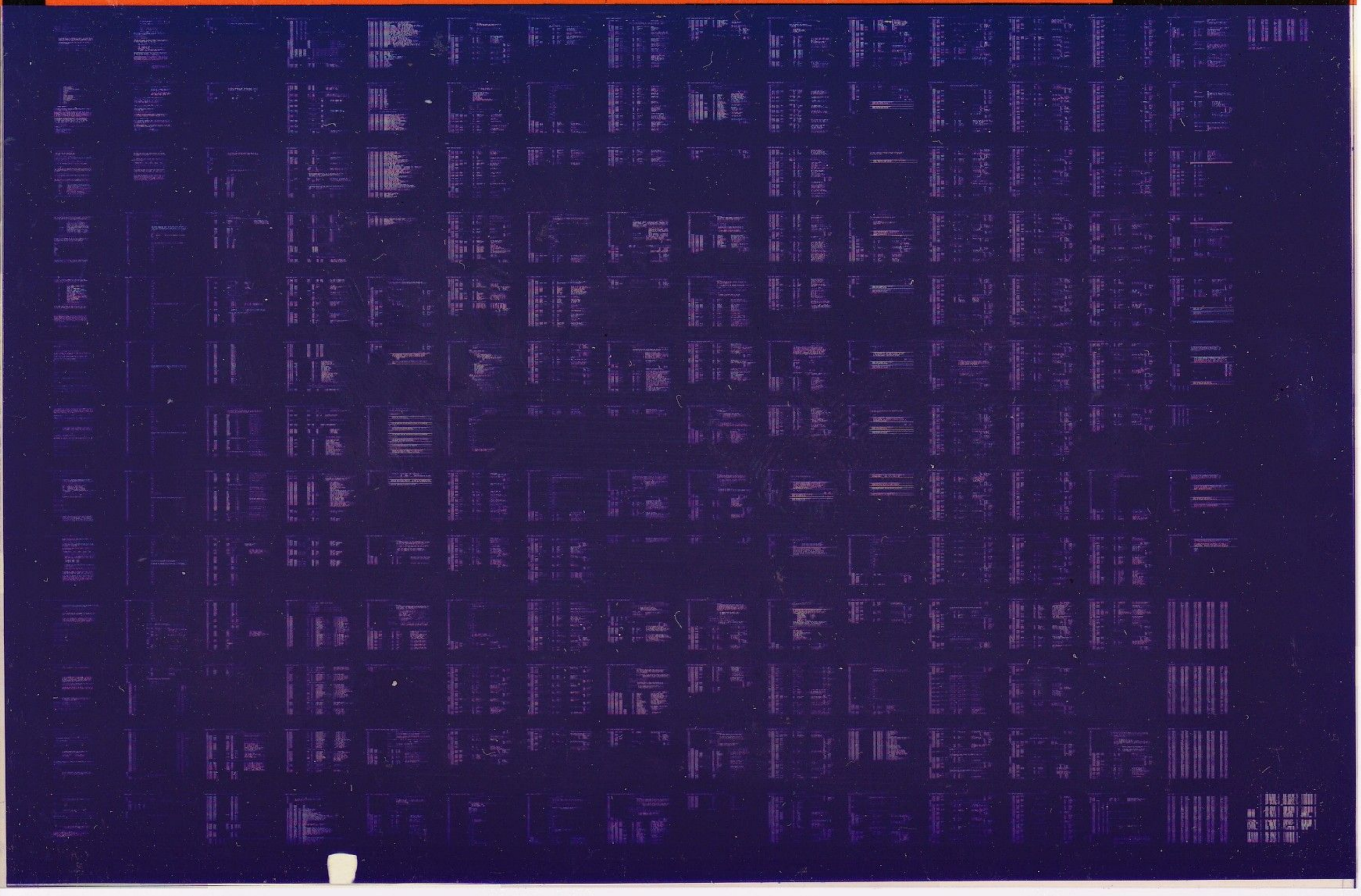
DEQNA DEQNA NI EXER DIAG
CVNIAA0

AH-T586A-MC
FICHE 01 OF 01

JUL 1984
digital
Made In USA



1

.REM &

IDENTIFICATION

PRODUCT CODE: AC T585A MC

PRODUCT NAME: CVNIAAO DEQNA NI EXERCISER DIAGNOSTIC

PRODUCT DATE: 09 MAR 84

MAINTAINER: MERRIMACK DIAGNOSTIC ENGINEERING

AUTHOR: STEVE SKONETSKI

TABLE OF CONTENTS

1.0 GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

THE NETWORK INTERCONNECT EXERCISER (NIE) PROGRAM IS MEANT TO PROVIDE FIELD SERVICE WITH A TOOL FOR DETERMINING THE CONNECTIVITY OF NODES ON THE NETWORK INTERCONNECT (NI).

THE NIE PROGRAM WILL DETERMINE THE ABILITY OF NODES ON THE NI TO COMMUNICATE WITH EACH OTHER AND PROVIDE NODE INSTALLATION VERIFICATION AND PROBLEM ISOLATION. THE NIE USES THE LOW LEVEL MAINTENANCE FEATURES OF THE DEQNA TO PROVIDE TESTING WITHOUT INTERRUPTING NORMAL OPERATION OF THE NI. THE VAX VERSION OF THE NIE CAN ALSO BE RUN CONCURRENTLY ON ANOTHER NODE, WITH EACH VERSION RUNNING INDEPENDENTLY OF EACH OTHER.

THIS DIAGNOSTIC HAS BEEN WRITTEN FOR USE WITH THE DIAGNOSTIC RUNTIME SERVICES SOFTWARE (SUPERVISOR). THESE SERVICES PROVIDE THE INTERFACE TO THE OPERATOR AND TO THE SOFTWARE ENVIRONMENT. THIS PROGRAM CAN BE USED WITH XXDP+, ACT, APT, SLIDE AND PAPER TAPE. FOR A COMPLETE DESCRIPTION OF THE RUNTIME SERVICES, REFER TO THE XXDP+ USER'S MANUAL. THERE IS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES IN SECTION 2 OF THIS DOCUMENT.

1.2 SYSTEM REQUIREMENTS

IN ORDER TO RUN THE CVNIA NIE PROGRAM, THE FOLLOWING MINIMUM HARDWARE IS REQUIRED:

A PDP 11 CPU WITH A Q-BUS (11/23, 11/23+, 11/03)
MINIMUM OF 24K WORDS OF MEMORY
A WORKING, LINE OR REAL TIME CLOCK
A CONSOLE TERMINAL
ANY XXDP+ SUPPORTED LOAD MEDIA
DEQNA QBUS TO ETHERNET ADAPTER

1.3 RELATED DOCUMENTS AND STANDARDS

DEQNA FUNCTIONAL AND PROGRAMMING SPEC
XXDP+ USER'S MANUAL (CHQUS?.SEQ WHERE ? IS THE REV. LEVEL OF THE MANUAL "C" IS THE CURRENT REV.)

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

THE GOAL OF THE NIE IS TO TEST THE COMMUNICATIONS LINK AND THEREFORE ASSUMES THAT THE CPU'S, CLOCKS, AND DEQNA'S OR DEQNA'S AT EACH END OF THE LINK HAVE ALREADY BEEN TESTED.

IF NO LINE OR REAL TIME CLOCK IS FOUND, THE PROGRAM WILL NOT RUN.

IT IS NOT THE INTENTION OF THE NIE TO TEST THE DEVICE (DEQNA), BUT TO TEST THE COMMUNICATIONS LINK TO WHICH IT IS CONNECTED.

1.5 ASSUMPTIONS

IT IS ASSUMED THAT THE COMMUNICATIONS DEVICE (DEQNA) HAS BEEN TESTED USING THE PREREQUISITE DIAGNOSTICS. THE OPERATOR SHOULD HAVE READ THE USER DOCUMENTATION PORTION OF THE LISTING TO FAMILIARIZE HIMSELF WITH THE COMMANDS AND CAPABILITIES AVAILABLE UNDER THE DIAGNOSTIC SUPERVISOR AND NIE.

2.0 OPERATING INSTRUCTIONS

THIS SECTION CONTAINS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES. FOR DETAILED INFORMATION, REFER TO THE XXDP+ USER'S MANUAL (CHQUS).

2.1 COMMANDS

THERE ARE ELEVEN LEGAL COMMANDS FOR THE DIAGNOSTIC RUNTIME SERVICES (SUPERVISOR). THIS SECTION LISTS THE COMMANDS AND GIVES A VERY BRIEF DESCRIPTION OF THEM. THE XXDP+ USER'S MANUAL HAS MORE DETAILS.

COMMAND	EFFECT
START	START THE DIAGNOSTIC FROM AN INITIAL STATE
RESTART	START THE DIAGNOSTIC WITHOUT INITIALIZING
CONTINUE	CONTINUE AT TEST THAT WAS INTERRUPTED (AFTER ^C)
PROCEED	CONTINUE FROM AN ERROR HALT
EXIT	RETURN TO XXDP+ MONITOR (XXDP+ OPERATION ONLY!)
ADD	ACTIVATE A UNIT FOR TESTING (ALL UNITS ARE CONSIDERED TO BE ACTIVE AT START TIME
DROP	DEACTIVATE A UNIT
PRINT	PRINT STATISTICAL INFORMATION (IF IMPLEMENTED BY THE DIAGNOSTIC - SECTION 4.0)
DISPLAY	TYPE A LIST OF ALL DEVICE INFORMATION
FLAGS	TYPE THE STATE OF ALL FLAGS (SEE SECTION 2.3)
ZFLAGS	CLEAR ALL FLAGS (SEE SECTION 2.3)

A COMMAND CAN BE RECOGNIZED BY THE FIRST THREE CHARACTERS. SO YOU MAY, FOR EXAMPLE, TYPE "STA' INSTEAD OF 'START'.

2.2 SWITCHES

THERE ARE SEVERAL SWITCHES WHICH ARE USED TO MODIFY SUPERVISOR OPERATION. THESE SWITCHES ARE APPENDED TO THE LEGAL COMMANDS. ALL OF THE LEGAL SWITCHES ARE TABULATED BELOW WITH A BRIEF DESCRIPTION OF EACH. IN THE DESCRIPTIONS BELOW, A DECIMAL NUMBER IS DESIGNATED BY "DDDDD".

SWITCH	EFFECT
/TESTS:LIST	EXECUTE ONLY THOSE TESTS SPECIFIED IN THE LIST. LIST IS A STRING OF TEST NUMBERS, FOR EXAMPLE /TESTS:1:5:7-10. THIS LIST WILL CAUSE TESTS 1,5,7,8,9,10 TO BE RUN. ALL OTHER TESTS WILL NOT BE RUN.
/PASS:DDDDD	EXECUTE DDDDD PASSES (DDDDD = 1 TO 64000)
/FLAGS:FLGS	SET SPECIFIED FLAGS. FLAGS ARE DESCRIBED IN SECTION 2.3.
/EOP:DDDDD	REPORT END OF PASS MESSAGE AFTER EVERY DDDDD PASSES ONLY. (DDDDD = 1 TO 64000)
/UNITS:LIST	TEST/ADD/DROP ONLY THOSE UNITS SPECIFIED IN THE LIST. LIST EXAMPLE - /UNITS:0:5:10-12 USE UNITS 0,5,10,11,12 (UNIT NUMBERS = 0-63)

EXAMPLE OF SWITCH USAGE:

START/TESTS:1-5/PASS:1000/EOP:100

THE EFFECT OF THIS COMMAND WILL BE: 1) TESTS 1 THROUGH 5 WILL BE EXECUTED, 2) ALL UNITS WILL TESTED 1000 TIMES AND 3) THE END OF PASS MESSAGES WILL BE PRINTED AFTER EACH 100 PASSES ONLY. A SWITCH CAN BE RECOGNIZED BY THE FIRST THREE CHARACTERS. YOU MAY, FOR EXAMPLE, TYPE "/TES:1-5" INSTEAD OF "/TESTS:1-5".

BELOW IS A TABLE THAT SPECIFIES WHICH SWITCHES CAN BE USED BY EACH COMMAND.

	TESTS	PASS	FLAGS	EOP	UNITS
START	X	X	X	X	X
RESTART	X	X	X	X	X
CONTINUE		X	X	X	
PROCEED			X		
DROP					X
ADD					X
PRINT					
DISPLAY					X
FLAGS					
ZFLAGS					
EXIT					

2.3 FLAGS

FLAGS ARE USED TO SET UP CERTAIN OPERATIONAL PARAMETERS SUCH AS LOOPING ON ERROR. ALL FLAGS ARE CLEARED AT STARTUP AND REMAIN CLEARED UNTIL EXPLICITLY SET USING THE FLAGS SWITCH. FLAGS ARE ALSO CLEARED AFTER A START COMMAND UNLESS SET USING THE

FLAG SWITCH. THE ZFLAGS COMMAND MAY ALSO BE USED TO CLEAR ALL FLAGS. WITH THE EXCEPTION OF THE START AND ZFLAGS COMMANDS, NO COMMANDS AFFECT THE STATE OF THE FLAGS; THEY REMAIN SET OR CLEARED AS SPECIFIED BY THE LAST FLAG SWITCH.

FLAG	EFFECT
HOE	HALT ON ERROR - CONTROL IS RETURNED TO RUNTIME SERVICES COMMAND MODE
LOE	LOOP ON ERROR
IER*	INHIBIT ALL ERROR REPORTS
IBE*	INHIBIT ALL ERROR REPORTS EXCEPT FIRST LEVEL (FIRST LEVEL CONTAINS ERROR TYPE, NUMBER, PC, TEST AND UNIT)
IXE*	INHIBIT EXTENDED ERROR REPORTS (THOSE CALLED BY PRINTX MACRO'S)
PRI	DIRECT MESSAGES TO LINE PRINTER
PNT	PRINT TEST NUMBER AS TEST EXECUTES
BOE	"BELL" ON ERROR
UAM	UNATTENDED MODE (NO MANUAL INTERVENTION)
ISR	INHIBIT STATISTICAL REPORTS (DOES NOT APPLY TO DIAGNOSTICS WHICH DO NOT SUPPORT STATISTICAL REPORTING)
IDR	INHIBIT PROGRAM DROPPING OF UNITS
ADR	EXECUTE AUTODROP CODE
LOT	LOOP ON TEST
EVL	EXECUTE EVALUATION (ON DIAGNOSTICS WHICH HAVE EVALUATION SUPPORT)

*ERROR MESSAGES ARE DESCRIBED IN SECTION 3.1

SEE THE XXDP+ USER'S MANUAL FOR MORE DETAILS ON FLAGS. YOU MAY SPECIFY MORE THAN ONE FLAG WITH THE FLAG SWITCH. FOR EXAMPLE, TO CAUSE THE PROGRAM TO LOOP ON ERROR, INHIBIT ERROR REPORTS AND TYPE A "BELL" ON ERROR, YOU MAY USE THE FOLLOWING STRING:

```
/FLAGS:LOE:IER:BOE
```

2.4 HARDWARE QUESTIONS

WHEN A DIAGNOSTIC IS STARTED, THE RUNTIME SERVICES WILL PROMPT THE USER FOR HARDWARE INFORMATION BY TYPING "CHANGE HW (L) ?' YOU MUST ANSWER "Y" AFTER A START COMMAND UNLESS THE HARDWARE INFORMATION HAS BEEN "PRELOADED" USING THE SETUP UTILITY (SEE CHAPTER 6 OF THE XXDP+ USER'S MANUAL). WHEN YOU ANSWER THIS QUESTION WITH A "Y", THE RUNTIME SERVICES WILL ASK FOR THE NUMBER OF UNITS (IN DECIMAL). YOU WILL THEN BE ASKED THE FOLLOWING QUESTIONS FOR EACH UNIT.

```
# UNITS (D) ? 1<CR>

UNIT 0
DEVICE CSR ADDRESS : (O) 174440 ?<CR>
INTERRUPT VECTOR ADDRESS : (O) 300 ?<CR>
```

```
        INTERRUPT PRIORITY : (0) 5 ?<CR>
```

WHEN YOU COMPLETE THE ABOVE SEQUENCE YOU WILL BE AT THE NIE>
COMMAND LEVEL.

```
        NIE> (A) ?
```

2.5 NETWORK INTERRCONNECT EXERCISER COMMANDS

THE "NIE>" COMMAND LEVEL FOLLOWS THE ATTACHING OF THE DEVICE AND ISSUING THE START TO THE SUPERVISOR. THESE COMMANDS CAN BE TYPED WHEN THE 'NIE>" PROMPT IS PRINTED.

YOU ONLY HAVE TO TYPE ENOUGH CHARACTERS TO UNIQUELY SPECIFY A COMMAND. THE COMMAND LINE IS INTERPRETED FROM LEFT TO RIGHT.

UNDERLINED ITEMS IN THE EXAMPLES REPRESENT THE USER'S RESPONSE TO THE PROMPT TO EXECUTE THAT PARTICULAR COMMAND.

HELP OR ? PRINTS OUT A BRIEF DESCRIPTION OF NIE COMMANDS.

EXAMPLE:

```
NIE> (A) ? H
```

OR

```
NIE> (A) ? ?
```

SHOW NODES PRINTS OUT THE CONTENTS OF THE NODE TABLE.

EXAMPLE:

```
NIE> (A) ? SH N
```

SHOW MESSAGE PRINTS OUT THE CURRENT MESSAGE PARAMETERS FOR SIZE, TYPE AND COPIES.

EXAMPLE:

```
NIE> (A) ? SH M
```

SHOW COUNTERS PRINTS OUT THE CONTENTS OF THE HOST NODE DEUNA INTERNAL COUNTERS.

****

NOTE:

BIT 3 OF THE RECEIVE ERROR BIT MASK IS THE "RUNT PACKET" BIT. THIS IS NOT PART

OF THE COUNTERS INFO. THIS BIT INDICATES THAT A RUNT PACKET WAS FOUND IN THE RECEIVE BUFFER. THE SOFTWARE WILL DISCARD THE RUNT PACKET. IT IS NOT CONSIDERED AN ERROR AND WILL NOT BE RECORDED AS ONE. THE HARDWARE DESIGN WILL ALLOW A RUNT PACKET CONDITION TO OCCUR AND REQUIRES THAT THE SOFTWARE HANDLE SUCH A CONDITION. SEE THE QNA DESIGN SPECIFICATIONS FOR FURTHER INFORMATION ON THE NATURE OF RUNT PACKETS.

THE MEANING OF THE COUNTERS IS EXPLAINED IN THE DEUNA USER'S GUIDE NUMBER EK DEUNA UG.

THE ERROR BIT MASKS FOR XMIT AND RECEIVE ARE IN BINARY FORMAT, NOT OCTAL OR DECIMAL (I.E. 11 = BIT1 AND BIT0, NOT AN OCTAL OR DECIMAL ELEVEN.

****

EXAMPLE:

```
NIE> (A) ? SH C
           --
```

NODE ADR/TYPE THE NODE COMMAND ALLOWS THE OPERATOR TO ENTER NODES INTO THE NODE TABLE. NODES ARE SPECIFIED USING THEIR 12 HEX DIGIT ETHERNET PHYSICAL ADDRESS AND CAN BE SPECIFIED AS EITHER TARGET OR ASSIST (A DEFAULT OF TARGET IS ASSUMED).

****

NOTE

IF THE NODE IS TO BE CHANGED FROM A TARGET TO AN ASSIST OR VICE VERSA, IT MUST BE CLEARED FROM THE TABLE FIRST!

WHEN RUNNING THE LOOPPAIR TEST, BE VERY CAREFUL WHEN DELETING NODES. IT IS BETTER TO DELETE NODES IN PAIRS (ONE TARGET NODE AND ONE ASSIST NODE).

****

EXAMPLE OF TARGET NODE ENTRY:

```
NIE> (A) ? N AA-00-04 FF-FF FO/T
             ----  -- --    -
```

EXAMPLE OF ASSIST NODE ENTRY:

```
NIE> (A) ? N AA 00-04 FF FF FO/A
             -   ---
```

EXAMPLE:

```
NIE> (A) ? N AA 00 04 FF-FF FO
             -              --
```

EXAMPLE OF CHANGING A TARGET NODE TO AN ASSIST NODE:

```
NIE> (A) ? CL N/AA 00-04-FF FF FO   (SEE CLEAR COMMAND)
           -  -  --

NIE> (A) ? N AA-00 04 FF FF FO/A
           -  ---  --  -  -  --


MESSAGE/TYPE=/SIZE=N/COPIES=M   THE MESSAGE COMMAND ALLOWS THE OPERATOR
                  TO SELECT THE CURRENT MESSAGE PARAMETERS AS FOLLOWS.
                  ANY OR ALL OF THE PARAMETER CAN BE CHANGED WITH THE COMMAND.
                  THE DEFAULT PARAMETERS ARE TYPE=ALPHA,SIZE=512,COPIES=1.

TYPE              ONE OF THE FOLLOWING MESSAGE TYPES:

                  ALPHA    !"#$%'()*+,-./0123456789;:=?ABCDEFG ETC.
                  ONES     MESSAGE OF ALL ONES (11111111....)
                  ZEROS -- MESSAGE OF ALL ZEROS (00000000....)
                  1ALT  -- ALTERNATING 1'S AND 0'S (10101010...)
                  0ALT  -- ALTERNATING 0'S AND 1'S (01010101...)
                  CCITT -  "CCITT" PSEUDO-RANDOM TEST PATTERN
                  OPERATOR SELECTED -- OPERATOR CHOSEN PATTERN OF LESS THAN
                    72 CHARACTERS USING 0-9, A-Z AND SPACES. (NOT USED IN
                    PATTERN TEST)

SIZE              THE SIZE OF THE MESSAGE BUFFER MAY BE BETWEEN 46 AND 512
                  BYTES.

COPIES            THE NUMBER OF COPIES OF EACH MESSAGE SENT TO EACH NODE
                  DURING A TEST MAY BE BETWEEN 1 AND 255.


EXAMPLE OF CHANGING TYPE:

NIE> (A) ? M/T=ZERO
           ----  --

EXAMPLE OF CHANGING SIZE:

NIE> (A) ? M/S=256
           ----   -

EXAMPLE OF CHANGING BOTH TYPE AND SIZE:

NIE> (A) ? M/S=512/T=ALPHA
             -  ------  -  --


RUN TEST/PASS=NN  CAUSES EXECUTION OF THE SPECIFIED TEST FOR NN NUMBER OF
                  PASSES.  A DEFAULT VALUE OF 1 IS ASSUMED IF /PASS=NN IS
                  NOT INCLUDED IN THE COMMAND LINE.  A VALUE OF NN= 1 WILL
                  CAUSE THE TEST TO BE RUN INDEFINITELY.  NODE ADDRESSES
                  FOR THE TESTS ARE TAKEN FROM THE NODE TABLE AND SHOULD BE
                  ENTERED PRIOR TO RUNNING THE TEST USING THE NODE COMMAND.
                  IN THE CASE OF THE LOOPPAIR TEST, NODE PAIRS ARE REQUIRED
```

AND MUST BE SPECIFIED AS TARGET AND ASSIST NODES. THE CURRENTLY SELECTED VALUES FOR MESSAGE TYPE, SIZE AND COPIES ARE USED BY EACH TEST.

THERE ARE FOUR TEST TO CHOSE FROM:

DIRECT — THE DIRECT TEST SENDS A LOOP DIRECT MESSAGE TO ALL OF THE NODES CONTAINED IN THE NODE TABLE AND WAITS FOR A RESPONSE. THE INTEGRITY OF THE RETURNED DATA IS CHECKED AND ANY ERRORS ARE REPORTED TO THE OPERATOR.

LOOPPAIR — THE LOOPPAIR TEST SENDS ASSISTED LOOPBACK MESSAGES TO THE NODE PAIRS CONTAINED IN THE NODE TABLE. THREE TYPE OF ASSISTED MESSAGES ARE SENT:

1) RECEIVE ASSIST -- HOST -> TARGET > ASSIST -> HOST
2) TRANSMIT ASSIST -- HOST -> ASSIST > TARGET > HOST
3) FULL ASSIST -- HOST -> ASSIST -> TARGET -> ASSIST > HOST

IN EACH CASE A RESPONSE IS WAITED FOR AND THE DATA IS CHECKED.

*** IMPORTANT!! *** ---> THE LOOPAIR TEST EXPECTS THAT EACH TARGET NODE HAVE A CORRESPONDING ASSIST NODE, AND THAT THE ORDER OF THE NODES IN THE NODE TABLE BE THE FOLLOWING:

```
FIRST NODE IN TABLE:  TARGET
  2ND NODE IN TABLE:  ASSIST
  3RD NODE IN TABLE:  TARGET
  4TH NODE IN TABLE:  ASSIST
                        ETC.
```

IF THERE ARE TWO ASSIST NODES IN A ROW, THE PROGRAM WILL CHANGE THE FIRST NODE TO A TARGET NODE, AND A SOFT ERROR WILL BE INDICATED.

BE VERY CAREFUL WHEN DELETING NODES. IT IS BEST TO DELETE NODES IN PAIRS (ONE TARGET NODE AND ONE ASSIST NODE).

PATTERN — THE PATTERN TEST SENDS SIX DIFFERENT LOOP DIRECT MESSAGES TO EACH NODE CONTAINED IN THE NODE TABLE. EACH OF THE SIX PATTERN TYPES (ALPHA, ONES, ZEROS, 1ALT, 0ALT, CCITT) IS USED FOR EACH NODE. RETURNED DATA IS CHECKED FOR ERRORS.

ALL — THE ALL NODE TEST PERFORMS THE MOST EXTENSIVE CHECK OF THE NETWORK AND IS COMPOSED OF TWO PARTS. FIRST A LOOP DIRECT MESSAGE IS SENT TO EACH NODE IN THE TABLE. IF THIS IS SUCCESSFUL, THE EXERCISER BUILDS AN ARRAY OF NODE PAIRS FROM THE TABLE AND SENDS A FULL ASSISTED LOOPBACK MESSAGE TO EACH PAIR IN THE ARRAY. A SAMPLE ARRAY OF PAIRS FOR A TABLE WITH 7 NODES IS SHOWN BELOW.

```
1-2   2-3   3 4   4 5   5-6   6 7
1 3   2 4   3 5   4 6   5 7
1 4   2-5   3-6   4 7
1 5   2 6   3-7
```

1 6 2 7
1 7

RESP RUNS THE RESPONDER TEST, A SECTION OF CODE THAT PROVIDES LOOP-SERVER FUNCTIONS SUCH AS FORWARDING MESSAGES, ANSWERING CONSOLE ID REQUESTS, AND TRANSMITTING A SYSTEM ID EVERY 8 TO 9 MINUTES.

****

I M P O R T A N T

THIS SECTION MUST BE RUN IF THE DEQNA IS TO BE USED AS A LOOP ASSIST OR TARGET NODE ON THE ETHERNET. THE OTHER TESTS WILL SIMPLY IGNORE FORWARDING REQUESTS, AND WILL NOT XMIT CONSOLE IDS.

THE ONLY WAY TO ESCAPE FROM A LARGE OR INFINITE NUMBER OF PASSES IS TO TYPE CONTROL C. BE CAREFUL!! IF YOU TYPE "START" TO DSR> AFTER THE CONTROL C YOU WILL DESTROY ALL SUMMARY STATISTICS AND COUNTERS. USE THE "RESTART" COMMAND TO DSR> TO GET BACK TO THE NIE> PROMPT AND PRESERVE THE COUNTERS.

****

EXAMPLE OF RUNNING THE DIRECT TEST ONE PASS:

```
NIE> (A) ? R D
```

EXAMPLE OF RUNNING THE DIRECT TEST 5 PASSES:

```
NIE> (A) ? R D/P=5
```

EXAMPLE OF RUNNING THE DIRECT TEST INFINITE PASSES:

```
NIE> (A) ? R D/P=0
```

EXAMPLE OF RUNNING LOOP PAIR TEST:

```
NIE> (A) ? R L
```

EXAMPLE OF RUNNING RESPONDER TEST:

```
NIE> (A) ? R R
```

IDENTIFY ADR — A REQUEST ID MESSAGE IS SENT TO THE NODE SPECIFIED BY ADR AND THE RESPONDED SYSTEM ID PARAMETERS ARE PRINTED.

EXAMPLE:

```
NIE> (A) ? ID AA 00 04 FF-FF FO
```

BUILD — THE BUILD COMMAND CAUSES THE EXERCISERS TO LISTEN FOR SYSTEM ID MESSAGES WHICH ARE BROADCAST BY ALL DEUNA NODES ONCE EVERY 10 MINUTES. ALL NODES IDENTIFYING THEMSELVES ARE ADDED TO THE NODE TABLE. THE BUILD COMMAND STOPS WHEN NO NEW NODES HAVE BEEN ADDED FOR 10 MINUTES OR WHEN 40 MINUTES HAVE ELAPSED. THE AVERAGE TIME FOR THIS COMMAND SHOULD BE 15 25 MINUTES.

IT IS POSSIBLE TO MISS A TRANSMISSION WITHIN THE TEN MINUTE PERIOD. IF NO NODES SHOW UP AFTER A BUILD, TRY WAITING 4 OR 5 MINUTES AND DO THE BUILD AGAIN.

A FEATURE OF THIS PROGRAM IS THE ABILITY TO ^C OUT OF THE BUILD ROUTINE AT ANY TIME WITHOUT HAVING TO WAIT FOR THE BUILDING PROCESS TO COMPLETE. THIS IS NICE WHEN YOU WANT TO TEST A FEW NODES AND DON'T CARE ABOUT "ALL" THE NODES ON THE NET.

EXAMPLE:

```
NIE> (A) ? BU
```

CLEAR NODE/ADR — THE CLEAR NODE COMMAND CLEARS THE SPECIFIED NODE FROM THE NODE TABLE. THE NODE CAN BE SPECIFIED BY EITHER ITS 12 DIGIT PHYSICAL ADDRESS OR ITS LOGICAL NAME (AS ASSIGNED BY NODE TABLE).

CLEAR NODE/ALL — THIS COMMAND CLEARS THE NODE TABLE.

CLEAR MESSAGE — THIS COMMAND SETS THE MESSAGE PARAMETERS BACK TO THE DEFAULT VALUES.

CLEAR SUMMARY — THIS COMMAND CLEARS THE SUMMARY TABLE.

EXAMPLE OF CLEARING A NODE USING THE NI ADDRESS:

```
NIE> (A) ? CL N/AA-00 04 FF-FF FO
```

EXAMPLE OF CLEARING A NODE USING IT'S LOGICAL NAME:

```
NIE> (A) ? CL N/N3
           -- - --
```

****

NOTE:

TO FIND WHICH LOGICAL NAME IS ASSOCIATED WITH AN ADDRESS, EXECUTE THE "SHOW NODE" COMMAND.

ALSO, A CLEARED NODE CAN BE RESTORED TO THE NODE TABLE BY EXECUTING THE "UNSAVE" COMMAND.

****

EXAMPLE OF CLEARING ALL NODES:

```
NIE> (A) ? CL N/ALL
           -- -----
```

SUMMARY — THE SUMMARY COMMAND PRINTS OUT THE SUMMARY TABLE. THE NIE MAINTAINS THE FOLLOWING INFORMATION FOR NODES WHO HAVE BEEN SENT MESSAGES:

RECEIVES NOT COMPLETE	RECEIVES COMPLETE
LENGTH ERRORS	DATA COMPARE ERRORS
BYTES COMPARED	BYTES TRANSFERRED

****

NOTE:

BYTES COMPARED REPRESENTS DATA MINUS THE LOOP-SERVER PROTOCOL OVERHEAD, AND THUS WILL BE LESS THAN BYTES TRANSFERRED WHICH REPRESENTS DATA PLUS LOOP SERVER PROTOCOL OVERHEAD

****

EXAMPLE:

```
NIE> (A) ? SUMM
           ---
```

SAVE — THE SAVE COMMAND SAVES THE CONTENTS OF THE NODE TABLE. FOR THE VAX VERSION, THE TABLE IS SAVED IN A FILE CALLED NIE.TBL. THE PDP-11 VERSION CANNOT WRITE TO EXTERNAL MEDIA, SO THE CONTENTS ARE SAVED INTERNALLY.

EXAMPLE:

```
NIE> (A) ? SAv
           --
```

UNSAVE THE UNSAVE COMMAND RESTORES THE CONTENTS OF THE NODE TABLE. USED. THE PDP 11 VERSION USES THE CONTENTS OF ITS INTERNALLY SAVED TABLE.

EXAMPLE:

```
NIE> (A) ? UNS
           ---
```

EXIT RETURNS CONTROL TO THE DIAGNOSTIC SUPERVISOR (EITHER VDS OR DRS).

****

NOTE:

THE DSR> RESTART AND CONTINUE COMMAND CANNOT BE USED IF THE EXIT COMMAND HAS BEEN USED TO LEAVE NIE>

****

EXAMPLE:

```
NIE> (A) ? EXIT
```

NOTES: 1) ADR IS THE PHYSICAL ADDRESS OF A NODE ON THE NI.
2) PASS COUNT IS A DECIMAL NUMBER BETWEEN 1 AND 65534. A DEFAULT VALUE OF 1 IS ASSUMED. SPECIFYING 1 CAUSES THE TEST TO BE RUN INDEFINITELY.

2.5.1 SOFTWARE QUESTIONS

AFTER YOU HAVE ANSWERED THE HARDWARE QUESTIONS OR AFTER A RESTART OR CONTINUE COMMAND, THE RUNTIME SERVICES WILL ASK FOR SOFTWARE PARAMETERS. THESE PARAMETERS WILL GOVERN SOME DIAGNOSTIC SPECIFIC OPERATION MODES. YOU WILL BE PROMPTED BY CHANGE SW (L) ? IF YOU WISH TO CHANGE ANY PARAMETERS, ANSWER BY TYPING "Y". THE SOFTWARE QUESTIONS AND THE DEFAULT VALUES ARE DESCRIBED IN THE NEXT PARAGRAPH(S).

2.6 QUICK START-UP PROCEDURE (XXDP.)

TO START-UP THIS PROGRAM:

1. BOOT XXDP.

2. GIVE THE DATE AND ANSWER THE LSI AND 50HZ (IF THERE IS A CLOCK) QUESTIONS

3. TYPE "R NAME", WHERE NAME IS THE NAME OF THE BIN OR BIC FILE FOR THIS PROGRAM

4. TYPE "START"

5. ANSWER THE "CHANGE HW' QUESTION WITH 'Y"

6. ANSWER ALL THE HARDWARE QUESTIONS

7. ANSWER THE "CHANGE SW" QUESTION WITH 'N"

WHEN YOU FOLLOW THIS PROCEDURE YOU WILL BE USING ONLY THE DEFAULTS FOR FLAGS AND SOFTWARE PARAMETERS. THESE DEFAULTS ARE DESCRIBED IN SECTIONS 2.3 AND 2.5.

3.0 ERROR INFORMATION

3.1 TYPES OF ERROR MESSAGES

THERE ARE THREE LEVELS OF ERROR MESSAGES THAT MAY BE ISSUED BY A DIAGNOSTIC: GENERAL, BASIC AND EXTENDED. GENERAL ERROR MESSAGES ARE ALWAYS PRINTED UNLESS THE "IER" FLAG IS SET (SECTION 2.3). THE GENERAL ERROR MESSAGE IS OF THE FORM:

```
NAME  TYPE  NUMBER  ON UNIT NUMBER  TST NUMBER PC:XXXXXX
ERROR MESSAGE
```

.WHERE; NAME = DIAGNOSTIC NAME
TYPE = ERROR TYPE (SYS FATAL, DEV FATAL, HARD OR SOFT)
NUMBER = ERROR NUMBER
UNIT NUMBER = 0 - N (N IS LAST UNIT IN PTABLE)
TST NUMBER = TEST AND SUBTEST WHERE ERROR OCCURRED
PC:XXXXXX = ADDRESS OF ERROR MESSAGE CALL

BASIC ERROR MESSAGES ARE MESSAGES THAT CONTAIN SOME ADDITIONAL INFORMATION ABOUT THE ERROR. THESE ARE ALWAYS PRINTED UNLESS THE "IER" OR "IBR" FLAGS ARE SET (SECTION 2.3). THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL MESSAGE.

EXTENDED ERROR MESSAGES CONTAIN SUPPLEMENTARY ERROR INFORMATION SUCH AS REGISTER CONTENTS OR GOOD/BAD DATA. THESE ARE ALWAYS PRINTED UNLESS THE "IER", "IBR" OR "IXR" FLAGS ARE SET (SECTION 2.3). THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL ERROR MESSAGE AND ANY ASSOCIATED BASIC ERROR MESSAGES.

3.2 SPECIFIC ERROR MESSAGES

ERROR MESSAGE:	MEANING
?ILL CMD BAD SYNTAX	A COMMAND WITH AN ILLEGAL CHAR WAS TYPED RETYPE THE COMMAND. THE VALID COMMANDS AND THEIR SYNTAX ARE SHOWN IN SECTION 2.5

?INCOMPLETE	A REQUIRED PART OF A COMMAND WAS LEFT OUT.
?NUMBER TOO BIG	THE VALUE OF A NUMERIC STRING IN THE COMMAND LINE WAS LARGER THAN 65535 OR 177777 OCTAL. (>16 BITS).
?BAD RADIX	A "8" OR "9" WAS TYPED WHEN AN OCTAL STRING WAS EXPECTED. PROBABLY OCCURRED WHEN TYPING A "DUMP" COMMAND WHERE OCTAL ADDRESSES ARE EXPECTED.

EXAMPLE OF A LOST PACKET ERROR DURING LOOPPAIR TESTING

```
CVNIA HRD ERR  00028 ON UNIT 00 TST 001 SUB 000 PC:064442

TIMEOUT OCCURRED - LOOP MESSAGE TYPE   RECEIVE ASSIST
FAILING TARGET NODE ADDRESS: AA-00-03-00-00-00
FAILING ASSIST NODE ADDRESS: AA-00 03-00-00-02
```

EXAMPLE OF A LOST PACKET ERROR DURING PATTERN TESTING

```
CVNIA HRD ERR 00028 ON UNIT 00 TST 001 SUB 000 PC:63730

TIMEOUT OCCURRED BEFORE LOOPBACK REPLY
FAILING NODE ADDRESS: AA-00-03-00-00-00
DATA PATTERN: ONES
```

4.0 PERFORMANCE AND PROGRESS REPORTS

AT THE END OF EACH PASS, THE PASS COUNT IS GIVEN ALONG WITH THE TOTAL NUMBER OF ERRORS REPORTED SINCE THE DIAGNOSTIC WAS STARTED. THE "EOP" SWITCH CAN BE USED TO CONTROL HOW OFTEN THE END OF PASS MESSAGE IS PRINTED. SECTION 2.2 DESCRIBES SWITCHES.

5.0 DEVICE INFORMATION TABLES

THIS IS THE DEFAULT HARDWARE P-TABLE. THE VALUES AND SIZE ARE USED AS A "TEMPLATE" FOR CREATING ACTUAL P-TABLE ENTRIES AND THE DEFAULT VALUES PROVIDED FOR THE OPERATOR. SEE SECTION 2.4 FOR AN EXAMPLE OF THE HARDWARE QUESTIONS.

THE NUMBERS IN BRACKETS (I.E. [6]) INDICATES THE OFFSET OF THE WORD INTO THE HARDWARE P-TABLE. THE OFFSETS MUST MATCH THE P-TABLE OFFSETS USED IN THE HARDWARE PARAMETER CODING SECTION WHERE THE "GET PARAMETER" CALLS ARE USED TO FILL THE P-TABLE.

```
.WORD   174440          ;[0] CSR ADDRESS
.WORD   300             ;[2] INTERRUPT VECTOR
.WORD   240             ;[6] INTERRUPT PRIORITY (5)
```

6.0 TEST SUMMARIES

6.1 DIRECT

THE DIRECT TEST SENDS A LOOP DIRECT MESSAGE TO ALL OF THE NODES CONTAINED IN THE NODE TABLE AND WAITS FOR A RESPONSE. THE INTEGRITY OF THE RETURNED DATA IS CHECKED AND ANY ERRORS ARE REPORTED TO THE OPERATOR.

6.2 LOOPPAIR

THE LOOPPAIR TEST SENDS ASSISTED LOOPBACK MESSAGES TO THE NODE PAIRS CONTAINED IN THE NODE TABLE. THREE TYPE OF ASSISTED MESSAGES ARE SENT:

1) RECEIVE ASSIST - HOST -> TARGET -> ASSIST > HOST
2) TRANSMIT ASSIST HOST > ASSIST -> TARGET -> HOST
3) FULL ASSIST HOST > ASSIST -> TARGET > ASSIST -> HOST

IN EACH CASE A RESPONSE IS WAITED FOR AND THE DATA IS CHECKED.

6.3 PATTERN

THE PATTERN TEST SENDS SIX DIFFERENT LOOP DIRECT MESSAGES TO EACH NODE CONTAINED IN THE NODE TABLE. EACH OF THE SIX PATTERN TYPES (ALPHA, ONES, ZEROS, 1ALT, 0ALT, CCITT) IS USED FOR EACH NODE. RETURNED DATA IS CHECKED FOR ERRORS.

6.4 ALL

THE ALL NODE TEST PERFORMS THE MOST EXTENSIVE CHECK OF THE NETWORK AND IS COMPOSED OF TWO PARTS. FIRST A LOOP DIRECT MESSAGE IS SENT TO EACH NODE IN THE TABLE. IF THIS IS SUCCESSFUL, THE EXERCISER BUILDS AN ARRAY OF NODE PAIRS FROM THE TABLE AND SENDS A FULL ASSISTED LOOPBACK MESSAGE TO EACH PAIR IN THE ARRAY. A SAMPLE ARRAY OF PAIRS FOR A TABLE WITH 7 NODES IS SHOWN BELOW.

1-2	2-3	3-4	4 5	5-6	6 7
1-3	2-4	3-5	4-6	5 7	
1-4	2 5	3-6	4-7		
1-5	2-6	3-7			
1-6	2-7				
1-7					

```
                                        .SBTTL  PROGRAM HEADER
                                        .ENABL  ABS,AMA
000000  002000                          .       =       2000


                                .SBTTL  PROGRAM MACROS


                                        ;I$STACK MACRO
                                        ;     --  - -

                                        ;+++
                                        ;THE I$STACK MACRO FACILITATES INITIALIZING THE R6 (HARDWARE) STACK
                                        ;AND THE R5 (PARAMETER) STACK.  R5 IS SET TO THE STACK LOW LIMIT
                                        ;(STAKLO) AND THE PARAMETER STACK GROWS UPWARD.  R6 IS SET TO THE
                                        ;STACK HIGH LIMIT (STAKHI) AND THE HARDWARE STACK GROWS DOWNWARD.
                                        ;IF THERE IS A STACK OVER-RUN, IT WILL BE DETECTED BY THE PREG14
                                        ;ROUTINE.
                                        ;---

                                        .MACRO  I$STACK STAKLO,STAKHI
                                        .LIST
                                        MOV     STAKLO,R5       ;INITIALIZE THE PARAMETER STACK POINTER.
                                        MOV     STAKHI,SP       ;INITIALIZE THE HARDWARE STACK POINTER.
                                        .NLIST
                                        .ENDM   I$STACK

                                        ;PUSH MACRO
                                        ; - -------

                                        ;+++
                                        ;THE "PUSH" MACRO FACILITATES PUSHING ITEMS ON THE HARDWARE STACK.
                                        ;UP TO SEVEN ITEMS MAY BE PLACED ON THE STACK WITH ONE MACRO.
                                        ;---
                                        .MACRO  PUSH    A,B,C,D,E,F,G

                                        .IF NB G
                                        .LIST
                                                                        MOV     G, (SP)
                                        .NLIST
                                        .ENDC

                                        .IF NB F
                                        .LIST
                                                                        MOV     F, (SP)
                                        .NLIST
                                        .ENDC

                                        .IF NB E
                                        .LIST
                                                                        MOV     E, (SP)
                                        .NLIST
                                        .ENDC

                                        .IF NB D
                                        .LIST
```

```
1003                                                                    MOV     D, (SP)
1004                            .NLIST
1005                            .ENDC
1006
1007                            .IF NB C
1008                            .LIST
1009                                                                    MOV     C, (SP)
1010                            .NLIST
1011                            .ENDC
1012
1013                            .IF NB B
1014                            .LIST
1015                                                                    MOV     B, (SP)
1016                            .NLIST
1017                            .ENDC
1018
1019                            .IF NB A
1020                            .LIST
1021                                                                    MOV     A, (SP)
1022                            .NLIST
1023                            .ENDC
1024
1025                            .ENDM   PUSH
1026
1027                            ;POP MACRO
1028                            ;---------
1029
1030                            ;+++
1031                            ;THE "POP" MACRO FACILITATES RETRIEVING ITEMS FROM THE HARDWARE STACK.
1032                            ;UP TO SEVEN ITEMS MAY BE RETRIEVED WITH ONE MACRO.
1033                            ;- -
1034
1035                            .MACRO  POP     A,B,C,D,E,F,G
1036
1037                            .IF NB A
1038                            .LIST
1039                                                                    MOV     (SP)+,A
1040                            .NLIST
1041                            .ENDC
1042
1043                            .IF NB B
1044                            .LIST
1045                                                                    MOV     (SP)+,B
1046                            .NLIST
1047                            .ENDC
1048
1049                            .IF NB C
1050                            .LIST
1051                                                                    MOV     (SP)+,C
1052                            .NLIST
1053                            .ENDC
1054
1055                            .IF NB D
1056                            .LIST
1057                                                                    MOV     (SP)+,D
1058                            .NLIST
1059                            .ENDC
```

```
1060
1061                                    .IF NB E
1062                                    .LIST
1063                                                            MOV     (SP)+,E
1064                                    .NLIST
1065                                    .ENDC
1066
1067                                    .IF NB F
1068                                    .LIST
1069                                                            MOV     (SP)+,F
1070                                    .NLIST
1071                                    .ENDC
1072
1073                                    .IF NB G
1074                                    .LIST
1075                                                            MOV     (SP)+,G
1076                                    .NLIST
1077                                    .ENDC
1078
1079                                    .ENDM   POP
1080
1081                                    ;CALL MACRO
1082                                    ;----------
1083
1084                                    ;+++
1085                                    ;THE CALL MACRO FACILITATES CALLING A SUBROUTINE VIA THE REGISTER
1086                                    ;PRESERVE ROUTINE (PREG14).  IT PLACES THE PARAMETERS TO BE PASSED ON
1087                                    ;THE PARAMETER STACK.  UP TO 7 PARAMETERS MAY BE PASSED USING THIS
1088                                    ;MACRO.
1089                                    ;---
1090
1091                                    .MACRO  CALL    S  A,B,C,D,E,F,G
1092
1093                                    .IF NB G
1094                                    .LIST
1095                                                            MOV     G,(R5)+
1096                                    .NLIST
1097                                    .ENDC
1098
1099                                    .IF NB F
1100                                    .LIST
1101                                                            MOV     F,(R5)+
1102                                    .NLIST
1103                                    .ENDC
1104
1105                                    .IF NB E
1106                                    .LIST
1107                                                            MOV     E,(R5)+
1108                                    .NLIST
1109                                    .ENDC
1110
1111                                    .IF NB D
1112                                    .LIST
1113                                                            MOV     D,(R5)+
1114                                    .NLIST
1115                                    .ENDC
1116
```

```
1117                                    .IF NB C
1118                                    .LIST
1119                                                                    MOV     C,(R5)+
1120                                    .NLIST
1121                                    .ENDC
1122
1123                                    .IF NB B
1124                                    .LIST
1125                                                                    MOV     B,(R5)+
1126                                    .NLIST
1127                                    .ENDC
1128
1129                                    .IF NB A
1130                                    .LIST
1131                                                                    MOV     A,(R5)+
1132                                    .NLIST
1133                                    .ENDC
1134
1135                                    .LIST
1136                                                                    JSR     R4,PREG14
1137                                                                    .WORD   S_ANCHOR
1138                                    .NLIST
1139                                    .ENDM   CALL
1140
1141                                    ;RETURN MACRO
1142                                    ;---- -------
1143
1144                                    ;+++
1145                                    ;THE RETURN MACRO FACILITATES PASSING PARAMETERS BACK TO A CALLING
1146                                    ;ROUTINE.  UP TO 7 PARAMETERS MAY BE PASSED BACK ON THE PARAMETER
1147                                    ;STACK.
1148                                    ;---
1149
1150                                    .MACRO  RETURN  A,B,C,D,E,F,G
1151
1152                                    .IF NB G
1153                                    .LIST
1154                                                                    MOV     G,(R5)+
1155                                    .NLIST
1156                                    .ENDC
1157
1158                                    .IF NB F
1159                                    .LIST
1160                                                                    MOV     F,(R5)+
1161                                    .NLIST
1162                                    .ENDC
1163
1164                                    .IF NB E
1165                                    .LIST
1166                                                                    MOV     E,(R5)+
1167                                    .NLIST
1168                                    .ENDC
1169
1170                                    .IF NB D
1171                                    .LIST
1172                                                                    MOV     D,(R5)+
1173                                    .NLIST
```

```
1174                                    .ENDC
1175
1176                                    .IF NB C
1177                                    .LIST
1178                                                            MOV     C,(R5)+
1179                                    .NLIST
1180                                    .ENDC
1181
1182                                    .IF NB B
1183                                    .LIST
1184                                                            MOV     B,(R5)+
1185                                    .NLIST
1186                                    .ENDC
1187
1188                                    .IF NB A
1189                                    .LIST
1190                                                            MOV     A,(R5)+
1191                                    .NLIST
1192                                    .ENDC
1193                                    .LIST
1194                                                            RTS     PC
1195                                    .NLIST
1196                                    .ENDM   RETURN
1197
1198                                    ;P$PUSH MACRO
1199                                    ;------------
1200
1201                                    ;+++
1202                                    ;THE P$PUSH MACRO FACILITATES PUSHING PARAMETERS ON THE PARAMETER
1203                                    ;STACK.  UP TO SEVEN ITEMS MAY BE PUSHED WITH ONE MACRO.
1204                                    ;---
1205
1206                                    .MACRO  P$PUSH  A,B,C,D,E,F,G
1207
1208                                    .IF NB G
1209                                    .LIST
1210                                                            MOV     G,(R5)+
1211                                    .NLIST
1212                                    .ENDC
1213
1214                                    .IF NB F
1215                                    .LIST
1216                                                            MOV     F,(R5)+
1217                                    .NLIST
1218                                    .ENDC
1219
1220                                    .IF NB E
1221                                    .LIST
1222                                                            MOV     E,(R5)+
1223                                    .NLIST
1224                                    .ENDC
1225
1226                                    .IF NB D
1227                                    .LIST
1228                                                            MOV     D,(R5)+
1229                                    .NLIST
1230                                    .ENDC
```

```
1231
1232                                    .IF NB C
1233                                    .LIST
1234                                                            MOV     C,(R5)+
1235                                    .NLIST
1236                                    .ENDC
1237
1238                                    .IF NB B
1239                                    .LIST
1240                                                            MOV     B,(R5)+
1241                                    .NLIST
1242                                    .ENDC
1243
1244                                    .IF NB A
1245                                    .LIST
1246                                                            MOV     A,(R5)+
1247                                    .NLIST
1248                                    .ENDC
1249
1250                                    .ENDM   P$PUSH
1251
1252                                    ;P$POP MACRO
1253                                    ;  ---- ---
1254
1255
1256                                    ;+++
1257                                    ;THE P$POP MACRO FACILITATES RETRIEVING PARAMETERS FROM THE PARAMETER
1258                                    ;STACK.  UP TO 7 PARAMETERS MAY BE RETRIEVED.
1259                                    ;
1260                                    ;THE ROUTINE THAT RECEIVES THE PARAMETERS HAS THE RESPONSIBILITY OF
1261                                    ;CLEANING UP THE PARAMETER STACK.  THIS MACRO IS AN AID TO MAKING
1262                                    ;A LOCAL COPY OF PASSED PARAMETERS AND CLEANING UP THE PARAMETER STACK.
1263                                    ;- -
1264
1265                                    .MACRO  P$POP   A,B,C,D,E,F,G
1266
1267                                    .IF NB A
1268                                    .LIST
1269                                                            MOV     (R5),A
1270                                    .NLIST
1271                                    .ENDC
1272
1273                                    .IF NB B
1274                                    .LIST
1275                                                            MOV     -(R5),B
1276                                    .NLIST
1277                                    .ENDC
1278
1279                                    .IF NB C
1280                                    .LIST
1281                                                            MOV     -(R5),C
1282                                    .NLIST
1283                                    .ENDC
1284
1285                                    .IF NB D
1286                                    .LIST
1287                                                            MOV     (R5),D
```

```
1288                                    .NLIST
1289                                    .ENDC
1290
1291                                    .IF NB E
1292                                    .LIST
1293                                                                    MOV     -(R5),E
1294                                    .NLIST
1295                                    .ENDC
1296
1297                                    .IF NB F
1298                                    .LIST
1299                                                                    MOV     -(R5),F
1300                                    .NLIST
1301                                    .ENDC
1302
1303                                    .IF NB G
1304                                    .LIST
1305                                                                    MOV     -(R5),G
1306                                    .NLIST
1307                                    .ENDC
1308
1309                                    .ENDM   P$POP
1310
1311
1316
1317                            .MACRO  CLI     CHAR,HITVAL,MISADR,CMPSTR
1318                                    NODCL   CHAR,HITVAL,\X$,MISADR,CMPSTR   ;;0000 PARSE TREE 0000
1319                            .ENDM
1320
1321
1322                            .MACRO  NODCL   CHAR,HITVAL,XY,MISADR,CMPSTR
1323                            NOD'XY: .BYTE   CHAR,HITVAL                     ;SPECIAL CHAR. CODE OR COMPARE CHAR.
1324                                                                            ; AND ACTION (HIT) VALUE FOR ACTION
1325                                                                            ; ROUTINES.
1326                            .IF NB  MISADR
1327                                    .WORD   MISADR NOD'XY                   ;DISPLACEMENT TO "MISS" NODE (BYTES)
1328                            .ENDC
1329                            .IF NB  CMPSTR
1330                                    .WORD   1$-NOD'XY                       ;DISPLACEMENT TO GET TO NEXT NODE
1331                                    .ASCIZ  CMPSTR                          ;  (ONLY IF ITS A "CLISTR" NODE)
1332                                    .EVEN
1333                              .NLIST
1334                              1$:
1335                              .LIST
1336                            .ENDC
1337                            .NLIST
1338                            X$=X$+1
1339                            .LIST
1340                            .ENDM
1341                            .MACRO  NEXT A,B
1342                                    .LIST
1343                                    .WORD   A'B                             ; SEGMENT BUFFER ADDRESS
1344                                    .NLIST
1345                            .ENDM
1346                            .MACRO  RNGFRM A,B,C                            ; MACRO TO FORM TRANSMIT AND RECEIVE
1347                                                                            ; DESCRIPTOR RINGS.
1348
```

```
                                        .LIST
                                        .WORD   0                       ; FLAG WORD
                                        .WORD   100000                  ; SET VALID BIT (DSCR/HIGH ADDRESS WRD)
                                        .NLIST
                                        NEXT    A,\B                    ; BUFFER LOW ADDRESS WORD
                                B=B+1
                                        .LIST
                                        .WORD   C                       ; BUFFER SIZE
                                        .WORD   0                       ; STATUS WORD 1
                                        .WORD   0                       ; STATUS WORD 2
                                        .NLIST
                                .ENDM
                                ;++
                                ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
                                ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
                                ; -

002000                                  POINTER BGNRPT

                                        HEADER  CVNIA,A,0,0,1,PRIO7
002000                                                                  L$NAME::
002000  103                                                                     .ASCII /C/
002001  126                                                                     .ASCII /V/
002002  116                                                                     .ASCII /N/
002003  111                                                                     .ASCII /I/
002004  101                                                                     .ASCII /A/
002005  000                                                                     .BYTE   0
002006  000                                                                     .BYTE   0
002007  000                                                                     .BYTE   0
002010                                                                  L$REV::
002010  101                                                                     .ASCII  /A/
002011                                                                  L$DEPO::
002011  060                                                                     .ASCII  /0/
002012                                                                  L$UNIT::
002012  000000                                                                  .WORD   0
002014                                                                  L$TIML::
002014  000000                                                                  .WORD   0
002016                                                                  L$HPCP::
002016  117114                                                                  .WORD   L$HARD
002020                                                                  L$SPCP::
002020  000000                                                                  .WORD   0
002022                                                                  L$HPTP::
002022  002170                                                                  .WORD   L$HW
002024                                                                  L$SPTP::
002024  000000                                                                  .WORD   0
002026                                                                  L$LADP::
002026  117312                                                                  .WORD   L$LAST
002030                                                                  L$STA::
002030  000000                                                                  .WORD   0
002032                                                                  L$CO::
002032  000000                                                                  .WORD   0
002034                                                                  L$DTYP::
002034  000001                                                                  .WORD   1
002036                                                                  L$APT::
002036  000000                                                                  .WORD   0
002040                                                                  L$DTP::
002040  002164                                                                  .WORD   L$DISPATCH
```

```
002042
002042  000340
002044
002044  000000
002046
002046  000000
002050
002050     003
002051     003
002052
002052  000000
002054  000000
002056
002056  000000
002060
002060  002122
002062
002062  102106
002064
002064  000000
002066
002066  000000
002070
002070  000000
002072
002072  000000
002074
002074  000000
002076
002076  002130
002100
002100  104035
002102
002102  000000
002104
002104  102126
002106
002106  103372
002110
002110  103370
002112
002112  102120
002114
002114  000000
002116
002116  000000
002120
002120  000000
1385
1396
1397                                    ;
1398                                    ; NAMES OF DEVICES SUPPORTED BY PROGRAM
1399                                    ;
1400 002122                                     DEVTYP  <DEQNA>
     002122
     002122     104     105     121
     002125     116     101     000

L$PRIO::
                .WORD   PRIO7
L$ENVI::
                .WORD   0
L$EXP1::
                .WORD   0
L$MREV::
                .BYTE   C$REVISION
                .BYTE   C$EDIT
L$EF::
                .WORD   0
                .WORD   0
L$SPC::
                .WORD   0
L$DEVP::
                .WORD   L$DVTYP
L$REPP::
                .WORD   L$RPT
L$EXP4::
                .WORD   0
L$EXP5::
                .WORD   0
L$AUT::
                .WORD   0
L$DUT::
                .WORD   0
L$LUN::
                .WORD   0
L$DESP::
                .WORD   L$DESC
L$LOAD::
                EMT     E$LOAD
L$ETP::
                .WORD   0
L$ICP::
                .WORD   L$INIT
L$CCP::
                .WORD   L$CLEAN
L$ACP::
                .WORD   L$AUTO
L$PRT::
                .WORD   L$PROT
L$TEST::
                .WORD   0
L$DLY::
                .WORD   0
L$HIME::
                .WORD   0

L$DVTYP::
                .ASCIZ  /DEQNA
```

```
                                                                                                .EVEN

   1401
   1407
   1408                                 ; TEST DESCRIPTION
   1409                                 ;
   1410 002130                                 DESCRIPT        <CVNID DEQNA NI EXERCISER>
                                                                                        L$DESC::
        002130                                                                          .ASCIZ  /CVNID DEQNA NI EXER
        002130      103     126     116
CISER/
        002133      111     104     040
        002136      104     105     121
        002141      116     101     040
        002144      116     111     040
        002147      105     130     105
        002152      122     103     111
        002155      123     105     122
        002160      000
                                                                                                .EVEN
   1411                                      .EVEN
   1412
   1419
   1420                                 ;
   1421                                 ; FORMAT STATEMENTS USED IN PRINT CALLS
   1422                                 ;
   1423
   1434
   1435
```

DISPATCH TABLE

```
1444                                    .SBTTL  DISPATCH TABLE
1445
1446                                    ;**
1447                                    ; THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
1448                                    ; IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
1449                                    ; -
1450
1451 002162                                    DISPATCH 1
                                                                                .WORD   1
     002162  000001
     002164                                                                 L$DISPATCH::
                                                                                .WORD   T1
     002164  103544
1452
```

```
                        .SBTTL  DEFAULT HARDWARE P TABLE

                        ;**
                        ; THE DEFAULT HARDWARE P TABLE CONTAINS DEFAULT VALUES OF
                        ; THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
                        ; IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P TABLES,
                        ; AND IS USED AS A "TEMPLATE" FOR BUILDING THE P TABLES.
                        ;

002166                          BGNHW   DFPTBL
002166  000003                                                          .WORD   L10000-L$HW/2
002170                                                          L$HW::
002170                                                          DFPTBL::

002170  174440                  .WORD   174440                 ; CSR
002172  000300                  .WORD   300                    ; VECTOR
002174  000240                  .WORD   PRIO5                  ; PRIORITY

002176                          ENDHW
002176                                                          L10000:
```

```
                                .SBTTL  SOFTWARE P TABLE

                                ;**
                                ; THE SOFTWARE TABLE CONTAINS VARIOUS DATA USED BY THE
                                ; PROGRAM AS OPERATIONAL PARAMETERS.  THESE PARAMETERS ARE
                                ; SET UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR
                                ; AT RUN TIME.
                                ;-

002176                                  BGNSW   SFPTBL
002176  000000                                                                 .WORD   L10001-L$SW/2
002200                                                                  L$SW::
002200                                                                  SFPTBL::

                                ;
                                ;
002200                                  ENDSW
002200                                                                  L10001:

                                .SBTTL  GLOBAL EQUATES SECTION

                                ;**
                                ; THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
                                ; ARE USED IN MORE THAN ONE TEST.
                                ;

002200                                  EQUALS
                                ;
                                ; BIT DIFINITIONS
                                ;
100000                          BIT15=: 100000
040000                          BIT14=: 40000
020000                          BIT13=: 20000
010000                          BIT12=: 10000
004000                          BIT11=: 4000
002000                          BIT10=: 2000
001000                          BIT09=: 1000
000400                          BIT08=: 400
000200                          BIT07=: 200
000100                          BIT06=: 100
000040                          BIT05=: 40
000020                          BIT04=: 20
000010                          BIT03=: 10
000004                          BIT02=: 4
000002                          BIT01=: 2
000001                          BIT00=: 1
                                ;
001000                          BIT9=:  BIT09
000400                          BIT8=:  BIT08
000200                          BIT7=:  BIT07
000100                          BIT6=:  BIT06
000040                          BIT5=:  BIT05
000020                          BIT4=:  BIT04
000010                          BIT3=:  BIT03
000004                          BIT2=:  BIT02
```

```
000002                  BIT1==  BIT01
000001                  BIT0==  BIT00
                        ;
                        ; EVENT FLAG DEFINITIONS
                        ;   EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION
                        ;
000040                  EF.START==      32.                     ; START COMMAND WAS ISSUED
000037                  EF.RESTART==    31.                     ; RESTART COMMAND WAS ISSUED
000036                  EF.CONTINUE==   30.                     ; CONTINUE COMMAND WAS ISSUED
000035                  EF.NEW==        29.                     ; A NEW PASS HAS BEEN STARTED
000034                  EF.PWR==        28.                     ; A POWER FAIL/POWER-UP OCCURRED
                        ;
                        ;
                        ; PRIORITY LEVEL DEFINITIONS
                        ;
000340                  PRI07== 340
000300                  PRI06== 300
000240                  PRI05== 240
000200                  PRI04== 200
000140                  PRI03== 140
000100                  PRI02== 100
000040                  PRI01== 40
000000                  PRI00== 0
                        ;
                        ;OPERATOR FLAG BITS
                        ;
000004                  EVL==       4
000010                  LOT==      10
000020                  ADR==      20
000040                  IDU==      40
000100                  ISR==     100
000200                  UAM==     200
000400                  BOE==     400
001000                  PNT==    1000
002000                  PRI==    2000
004000                  IXE==    4000
010000                  IBE==   10000
020000                  IER==   20000
040000                  LOE==   40000
100000                  HOE==  100000
```

```
1543
1544                                    ;
1545                                      ;CSR DEFINITIONS
1546            000010                  BD      ==      BIT03           ;BOOT/DIAGNOSTIC ROM
1547            001000                  EL      ==      BIT09           ;ELOOP
1548            010000                  XC      ==      BIT12           ;TRANSCEIVER
1549                                    ;
1550                                      ;DESCRIPTOR FIELDS
1551                                    ;
1552            000000                  FLAGWORD==      0               ;FLAG WORD
1553            000010                  STAT3   ==      8.              ;STATUS WORD 1
1554            000012                  STAT4   ==      10.             ;STATUS WORD 2
1555                                    ;
1556                                      ;DESCRIPTOR BITS
1557                                    ;
1558            100000                  V       ==      BIT15           ;VALID BIT
1559                                    ;
1560            000004                  CITVECTOR==     4               ;OFFSET IN 4K ROM FOR VECTOR TO CITIZENSHIP TEST
1561                                    ;
1562                                    ; TERMINAL CONTROL CHARACTERS
1563                                    ;
1564            000015                  CR      == 15
1565            000012                  LF      == 12
1566
1567                                    ;;;EQUATES FOR FLAG WORD;;;;;
1568
1569            000000                          CTARGT==0
1570            000001                          CASIST==1
1571            000002                          CSHCTR==2               ;ARG TYPE FOR 'SHOW COUNTERS' CMD
1572            000004                          CCLNAD==4               ;ARG TYPE FOR 'CLEAR NODE/ADR  CMD
1573            000010                          CCLNAL==8.              ;ARG TYPE FOR 'CLEAR NODE/ALL' CMD
1574            000020                          CEXIT==16.
1575
1576                                    ;;;CLOCK ENABLE VALUES TO BE LOADED IN CLK'S CSR;;;
1577
1578            000100                          LCLKEN==100             ; L-CLOCK CSR VALUE TO ENABLE THE CLOCK
1579            000111                          PCLKEN==111             ; P-CLOCK CSR VALUE TO ENABLE THE CLOCK
1580            001600                          PCLKCT==1600            ; P-CLOCK COUNT SET REGISTER FOR COUNTER
1581
1582                                    ; SPECIAL CLI CODES FOR "CHAR" ARGUEMENT IN CLI CALLS
1583                                    ;   (COMMAND LINE INTERPRETER DEFINITIONS)
1584            000000                          CLIERR= 0
1585            000001                          CLIEXT= 1
1586            000002                          CLIBR = 2
1587            000003                          CLIBIF= 3
1588            000004                          CLISPA= 4
1589            000005                          CLINUM= 5
1590            000006                          CLIALP= 6
1591            000007                          CLIALN= 7
1592            000010                          CLIOCT= 8.
1593            000011                          CLIDEC= 9.
1594            000012                          CLISTR= 10.
1595
1596                                    ;DEFS FOR COMMAND LINE INTERPRETATION ACTION VALUES
1597
1598            000000                          NULL=0
1599            000001                          HELP=1
```

GLOBAL EQUATES SECTION

```
1600            000002                  NODE=2
1601            000003                  BUILD=3
1602            000004                  CRUN=4
1603            000005                  CPATRN=5
1604            000006                  CSAVE=6
1605            000007                  SUMMRY=7
1606            000010                  IDENT=10
1607            000011                  EXIT=11
1608            000012                  NOTNUF=12
1609            000013                  CEXADR=13
1610            000014                  CSAVR4=14
1611            000015                  CNODE=15
1612            000016                  CALPHA=16
1613            000017                  CONES=17
1614            000020                  CZEROS=20
1615            000021                  C1ALT=21
1616            000022                  COALT=22
1617            000023                  CCCITT=23
1618            000024                  COPRSL=24
1619            000025                  CTYPE=25
1620            000026                  CSIZE=26
1621            000027                  CCPYS=27
1622            000030                  CNDADR=30
1623            000031                  CNODAL=31
1624            000032                  CRNALL=32
1625            000033                  CLUPPR=33
1626            000034                  CSHMSG=34
1627            000035                  CCLMSG=35
1628            000036                  CCNTR=36
1629            000037                  CNDLOG=37
1630            000040                  CFUNCT=40
1631            000041                  CUNSAV=41
1632            000042                  CCLSUM=42
1633            000043                  CDIR=43
1634            000044                  CDEFLT=44
1635            000045                  CUNSVF=45
1636            000046                  RSPOND=46
1637                                    ;
1638                                    ;       MESSAGE TYPE VALUES
1639                                    ;
1640            000000                  ALPHA==0
1641            000001                  ONES==1
1642            000002                  ZEROS==2
1643            000003                  ONEALT==3
1644            000004                  ZROALT==4
1645            000005                  CCITT==5
1646            000006                  OPRSEL==6
1647                                    ;
1648                                    ;       GLOBAL EQUATES FOR THE DEQNA DRIVER
1649                                    ;
```

```
1651                                    .SBTTL  BIT DEFINITIONS
1652                                ;*
1653                                ; THIS SECTION CONTAINS BIT DEFINTIONS FOR THE QNA CSRS AND BUFFER
1654                                ; DESCRIPTORS.  ALL UNDEFINED CSR BITS ARE RESERVED AND/OR UNUSED.
1655                                ; ALL $V DEFINITIONS REPRESENT BIT NUMBERS IN THE CSR, ALL $M
1656                                ; DEFINITIONS ARE A MASK OF THAT CSR BIT
1657                                ;
1658    000000                      VRCVEN          == 00           ; RECEIVER ENABLE BIT
1659    000001                      MRCVEN          == 1
1660    000001                      VRESET          == 01           ; SOFTWARE RESET BIT
1661    000002                      MRESET          == 2
1662    000002                      VNXM            == 02           ; NON EXISTENT MEMORY BIT
1663    000004                      MNXM            == 4
1664    000003                      VBOOT           == 03           ; BOOT/DIAGNOSTIC ROM LOAD
1665    000010                      MBOOT           == 10
1666    000004                      VINVXM          == 04           ; XMIT LIST INVALID BIT
1667    000020                      MINVXM          == 20
1668    000005                      VINVRC          == 05           ; RECEIVE LIST INVALID BIT
1669    000040                      MINVRC          == 40
1670    000006                      VINTEN          == 06           ; ENABLE INTERRUPT ON BITS 2, 7, OR 15
1671    000100                      MINTEN          == 100
1672    000017                      VRCVIN          == 17           ; RECEIVE INTERRUPT REQUEST
1673    100000                      MRCVIN          == 100000
1674    000010                      VILOOP          == 10           ; INTERNAL LOOPBACK BIT
1675    000400                      MILOOP          == 400
1676    000011                      VELOOP          == 11           ; EXTERNAL LOOPBACK BIT
1677    001000                      MELOOP          == 1000
1678    000012                      VSANITY         == 12           ; SANITY TIMER ENABLE BIT
1679    002000                      MSANITY         == 2000
1680    000015                      VCARRIER        == 15           ; CARRIER SENSE BIT
1681    003000                      MCARRIER        == 3000
1682    000007                      VXMTIN          == 07           ; XMIT INTERRUPT REQUEST
1683    000200                      MXMTIN          == 200
1684                                ;
1685                                ; THE FOLLOWING DEFINTIONS ARE FOR THE DESCRIPTOR BITS WHICH ARE A PART
1686                                ;OF THE BUFFER DESCRIPTOR LIST ENTRIES
1687                                ;
1688    000006                      VHIBYT          == 6            ; XMIT BUFFER BEGINS ON HIGH BYTE
1689    000100                      MHIBYT          == 100
1690    000007                      VLOBYT          == 7            ; XMIT BUFFER BEGINS ON LOW BYTE
1691    000200                      MLOBYT          == 200
1692    000014                      VXSETP          == 14           ; PACKET IS A SETUP PACKET
1693    010000                      MXSETP          == 10000
1694    000015                      VEOM            == 15           ; END OF MESSAGE
1695    020000                      MEOM            == 20000
1696    000016                      VCHADR          == 16           ; ADRS IS ANOTHER LIST ADRS. NOT DATA
1697    040000                      MCHADR          == 40000
1698    000017                      VVALID          == 17           ; VALID LIST ENTRY
1699    100000                      MVALID          == 100000
1700                                ;
1701                                ; THE FOLLOWING DEFINITIONS ARE FOR THE FIRST STATUS WORD FROM THE QNA
1702                                ; FOUND IN THE BUFFER DESCRIPTOR LIST ENTRIES WHEN A XMIT COMPLETES
1703                                ;
1704    000004                      VCOLLO          == 4            ; XMIT COLLISION COUNTER BIT ZERO
1705    000004                      SCOLL           == 4            ; SIZE OF THE COLLISION COUNTER FIELD
1706    000010                      VFAIL           == 10           ; HEARTBEAT COLLISION CHECK FAILURE
1707    000400                      MFAIL           == 400
```

```
BIT DEFINITIONS

1708    000013          VABORT          == 13                   ; XMISSION ABORTED, TOO MANY COLLISIONS
1709    004000          MABORT          == 4000
1710    000013          VNOCAR          == 13                   ; NO CARRIER SEEN DURING XMISSION
1711    002000          MNOCAR          == 2000
1712    000014          VLOSS           == 14                   ; CARRIER LOST DURING TRANSMISSION
1713    010000          MLOSS           == 10000
1714    000016          VXUSED          == 16                   ; XMIT BUFFER IS USED/IN USE
1715    040000          MXUSED          == 40000
1716    000016          VXERRS          == 16                   ; XMIT ERROR OCCURRED
1717    040000          MXERRS          == 40000
1718    000015          VXLAST          == 15                   ; LAST XMIT BUFFER OF A MESSAGE
1719    100000          MXLAST          == 100000
1720                    ;
1721                    ; FOLLOWING ARE THE DEFINITIONS FOR STATUS WORD 2 WRITTEN BY THE QNA.  THESE
1722                    ; ARE XMIT STATUS DEFINTIONS
1723                    ;
1724    000000          VTDR            == 0                    ; TIME DOMAIN REFLECT. BIT 0
1725    000016          STDR            == 16                   ; SIZE OF TIME DOMAIN REFLECT. FIELD
1726                    ;
1727                    ; THE FOLLOWING DEFINITIONS ARE FOR STATUS WORD 1 ON A RECEIVE OPERATION
1728                    ;
1729    000000          VOVF            == 0                    ; RECEIVER OVERFLOW OCCURRED
1730    000001          MOVF            == 1
1731    000001          VCRC            == 1                    ; CRC ERROR, IGNORE IF DISCARD NOT SET
1732    000002          MCRC            == 2
1733    000002          VFRAM           == 2                    ; RECEIVE FRAMING ERROR OCCURRED
1734    000004          MFRAM           == 4
1735    000003          VSHORT          == 3                    ; PACKET OF LESS THAN TEN BYTES RCVD
1736    000010          MSHORT          == 10
1737    000010          VRBL            == 10                   ; RECEIVED BYTE LENGTH BIT 8
1738    000003          SRBL            == 3                    ; SIZE OF RECEIVED BYTE LENGTH FIELD
1739    000013          VRUNT           == 13                   ; RUNT PACKET RECEIVED, DISCARD MESSAGE
1740    004000          MRUNT           == 4000
1741    000014          VDISC           == 14                   ; DISCARD THE CURRENT MESSAGE
1742    010000          MDISC           == 10000
1743    000015          VRLONG          == 15                   ; PACKET RECEIVED WAS TOO LARGE
1744    020000          MRLONG          == 20000
1745    000015          VRSETP          == 15
1746    020000          MRSETP          == 20000
1747    000016          VRUSED          == 16                   ; RECEIVE BUFFER IS OR BEING USED
1748    040000          MRUSED          == 40000
1749    000016          VRERRS          == 16                   ; RECEIVE ERROR OCCURRED
1750    040000          MRERRS          == 40000
1751    000017          VRLAST          == 17                   ; LAST RECEIVE BUFFER OF A MESSAGE
1752    100000          MRLAST          == 100000
1753                    ;
1754                    ; THE FOLLOWING DEFINITIONS ARE FOR STATUS WORD 2 AFTER A RECEIVE OPERATION
1755                    ;
1756    000000          VRBL            == 0                    ; RECEIVED BYTE LENGTH
1757    000010          SRBL            == 10                   ; SIZE OF RECEIVED BYTE LENGTH FIELD
1758                    ;
1759                    ; THE FOLLOWING DEFINITIONS ARE FOR THE FLAG WORD IN THE BUFFER DESCRIPTOR
1760                    ; ENTRIES
1761                    ;
1762    000016          VFUSED          == 16                   ; THIS BUFFER IS OR BEING USED FLAG
1763    040000          MFUSED          == 40000
1764    000016          VFERRS          == 16                   ; ERROR OCCURRED FLAG
```

```
BIT DEFINITIONS

040000                  MFERRS          == 40000
000017                  VFLAST          == 17                   ; LAST BUFFER OF A MESSAGE
100000                  MFLAST          == 100000
                        ;
                        ;       ETHERNET PACKET  OFFSETS
                        ;
000016                  HEADER  ==      14.             ; OFFSET (SIZE) TO END OF  HEADER IN BYTES
000000                  DESTIN  ==      0               ; DESTINATION ADDRESS
000006                  SOURCC  ==      6               ; SOURCE ADDRESS
000014                  PROTOT  ==      12.             ; PROTOCOL TYPE FIELD
                        ;
                        ; MISCELLANEOUS DEFS
                        ;
001100                  XPKLEN  == 1100                 ; XMT PACKET LENGTH IN BYTES
001100                  RPKLEN  == 1100                 ; RCV PACKET LENTH IN BYTES
000050                  TBLLEN  == 40.                  ; NODE TABLE LENGTH (CHANGE STBLEN IF
                                                        ;  THIS IS CHANGED, OR ELSE!)
000132                  STBLEN  == 90.                  ; SUMMARY TABLE LENGTH (= 2.25 X TBLLEN)
000004                  FRDADR  == 4                    ; OFFSET FOR MESSAGE HEADERS
                        ;
                        ; SYSTEM ID REPLY MESSAGE OFFSETS
                        ;
                                RIFUNC == 20
000020                          SIRCPT == 22
000022                          SIVERS == 27
000027                          SIECO  == 30
000030                          SIUECO == 31
000031                          SIFNCT == 35
000035                          SIADDR == 42
000042                          SIDEV  == 53
000053                  ;
                        ; LOOP DIRECT OFFSETS
                        ;
                                LDSKIP == 16
000016                          LDFCT1 == 20
000020                          LDADR1 == 22
000022                          LDFCT2 == 30
000030                          LDADR2 == 32
000032                  ;
                        ; FULL ASSIST OFFSETS
                        ;
                                FASKIP == 16
000016                          FAFCT1 == 20
000020                          FAADR1 == 22
000022                          FAFCT2 == 30
000030                          FAADR2 == 32
000032                          FAFCT3 == 40
000040                          FAADR3 == 42
000042                          FAFCT4 == 50
000050                          FAADR4 == 52
000052                  ;
                        ; COUNTER OFFSETS
                        ;
                                C.SIZ  == 0
000000                          C.SECS == 2
000002                          C.PREC == 4
000004
```

```
1822            000010                          C.MREC == 10
1823            000014                          C.RERB == 14
1824            000016                          C.RERR == 16
1825            000020                          C.RDAT == 20
1826            000024                          C.RMDB == 24
1827            000030                          C.RLIN == 30
1828            000032                          C.RLEX == 32
1829            000034                          C.PXMT == 34
1830            000040                          C.MXMT == 40
1831            000044                          C.PXM3 == 44
1832            000050                          C.PXM2 == 50
1833            000054                          C.PXMD == 54
1834            000060                          C.XDAT == 60
1835            000064                          C.XMDB == 64
1836            000070                          C.XABB == 70
1837            000072                          C.XABT == 72
1838            000074                          C.COLL == 74
1839                                    ;
1840                                    ; XMIT ABORT REASON MASK BIT DEFINITIONS
1841                                    ;
1842            000001                          C.RTRY == BIT00
1843            000002                          C.LCAR == BIT01
1844            000004                          C.SHRT == BIT02
1845            000010                          C.OPEN == BIT03
1846                                    ;
1847                                    ; RECEIVE ERROR REASON MASK BIT DEFINITIONS
1848                                    ;
1849            000001                          C.CRC  == BIT00
1850            000002                          C.FRAM == BIT01
1851            000004                          C.MLEN == BIT02
1852            000010                          C.RUNT == BIT03
1853                                    ;
1854                                    ; ROUTINE FUNCTION CODES
1855                                    ;
1856            000000          BMPRCV  == 0                    ; UPDATE RECEIVE COUNTERS
1857            000001          BMPXMT  == 1                    ; UPDATE TRANSMIT COUNTERS
1858            000000          INIADR  == 0                    ; INITIALIZE ADDRESS SETUP TABLE
1859            000001          ADDMUL  == 1                    ; ADD MULTICAST ADDRESS TO SETUP TABLE
1860            000002          KILMUL  == 2                    ; KILL MULTICAST ADDRESS IN SETUP TABLE
1861            000000          XMTSET  == 0                    ; XMIT AN ADDRESS SETUP PACKET
1862            000001          XMTDAT  == 1                    ; XMIT A DATA PACKET
1863                                    ;
1864                                    ; FUNCTION CODES FOR LOOP-SERVER PACKETS
1865                                    ;
1866            000002          FORWRD  == 2
1867            000001          REPLY   == 1
1868            000005          IDFUNC  == 5
1869                                    ;
1870                                    ; QNA DEVICE REGISTER OFFSETS
1871                                    ;
1872            000000          NETADD  == 0
1873            000004          LORCV   == 4
1874            000006          HIRCV   == 6
1875            000010          LOXMT   == 10
1876            000012          HIXMT   == 12
1877            000014          VECTOR  == 14
1878            000016          CSR     == 16
```

```
1879                                ;
1880                                ;BUFFER DESCRIPTOR OFFSET DEFINITIONS
1881                                ;
1882            000000              FLAG    == 0
1883            000002              DESC    == 2
1884            000002              HIADD   == 2
1885            000004              LOADD   == 4
1886            000006              WRDCNT  == 6
1887            000010              STAT1   == 10
1888            000012              STAT2   == 12
1889                                ;-
```

```
1891                                    .SBTTL  GLOBAL DATA SECTION
1892                                    ;
1893                                    ;COMMAND LINE BUFFER, DATA LOCATIONS AND MESSAGES FOR ACTION ROUTINES
1894
1895 002200  000000   BLDFLG: .WORD   0               ; INDICATES THAT WE ARE IN TABLE BUILD MODE
1896 002202            CMDBUF: .BLKB   72.             ;BUFFER FOR OPERATOR COMMANDS
1897 002312  000000    KEYWD1: .WORD   0               ;
1898 002314  000000    KEYWD2: .WORD   0
1899 002316  000000    ADRBUF: .WORD   0               ;BUFFER FOR NODE ADDRESS
1900 002320  000000            .WORD   0
1901 002322  000000            .WORD   0
1902 002324            STRBUF: .BLKB   18.             ;BUFFER FOR ALPHANUM. ADDRESS STRING
1903 002346            STRBU1: .BLKB   18.
1904 002370  000000    CBOADR: .WORD   0               ;POINTER FOR BEGINING OF ADDRESS STRING
1905 002372  000000    P$TYPE: .WORD   0               ;LOC. TO HOLD MESSAGE TYPE
1906 002374  000000    P$SIZE: .WORD   0               ;LOC. TO HOLD MESSAGE SIZE
1907 002376  000000    P$CPYS: .WORD   0               ;LOC. TO HOLD NO. OF MESSAGE COPIES
1908 002400  000000    P$PASS: .WORD   0               ;LOC. TO HOLD NO. OF PASSES
1909 002402  000000    LSRTRY: .WORD   0               ; NUMBER OF RETRIES FOR LOSS OF CARRIER
1910 002404  000000    NCRTRY: .WORD   0               ; NUMBER OF RETRIES FOR NO CARRIER
1911 002406  000000    NODTY:  .WORD   0               ;LOC. TO HOLD NODE TYPE FOR NODE TABLE SETUP
1912 002410  000000    SLOT::  .WORD   0               ;USED BY NODE TABLE SUBROUTINES
1913 002412            NODTBL: .BLKW   TBLLEN          ; SPACE FOR NODE TABLE
1914 002532  177777            .WORD   177777          ;FILL LAST FOUR BYTES OF TABLE WITH ONES
1915 002534  177777            .WORD   177777
1916 002536            SAVTBL: .BLKW   TBLLEN          ;SPACE FOR SAVE TABLE
1917 002656  177777    ILLADR: .WORD   177777          ;ILLEGAL ADDRESS FOR COMPARISON
1918 002660  177777            .WORD   177777          ; (MUST NOT BE PHYSICALLY SEPARATED FROM
1919 002662  177777            .WORD   177777          ; END OF SAVTBL)
1920 002664            STATBF: .BLKB   30              ; BUFFERS SUMMARY TABLE FOR SUMMARY REPORT
1921 002714            STATBL: .BLKW   STBLEN          ;SPACE FOR SUMMARY TABLE
1922 003200  177777            .WORD   177777
1923 003202  000000    RSPFLG: .WORD   0               ; INDICATES IF WE ARE IN RESPONDER MODE
1924                                    ;
1925                                    ;COMMAND LINE TRAVERSE LOCATIONS (USED BY "P$TRV")
1926                                    ;
1927 003204  000000    P$BUFA: .WORD   0               ;LOC. TO HOLD ADDR. OF CMD LINE BUFFER
1928 003206  000000    P$TREE: .WORD   0               ;LOC. TO HOLD ADDR. OF PARSING TREE
1929 003210  000000    P$ACT:  .WORD   0               ;LOC. TO HOLD ADDR. OF ACTION ROUTINE
1930 003212  000000    P$CNT:  .WORD   0               ;LOC. TO BE A COUNTER LOCATION
1931 003214  000000    P$NUM:  .WORD   0               ;LOC. TO HOLD NUMERIC VALUE FROM PARSE
1932 003216  000000    P$RADX: .WORD   0               ;LOC. TO HOLD RADIX(LO) & +/-(HI BYTE)
1933 003220     000    P$NNUF: .BYTE   0               ;RETURN =0 IF ENOUGH OF COMMAND FOUND
1934 003221     000    P$GDBD: .BYTE   0               ;RETURN CODE 0 IF NO ERROR FOUND
1935 003222     000    P$AERR: .BYTE   0               ;RETURN 0 IF 12 DIGIT ADDRESS ENTERED
1936 003223     000    P$MERR: .BYTE   0               ;RETURN -1 IF ERROR IN OPERATOR SELECTED
1937                                                   ;MESSAGE INPUT OCCURED, 0 FOR GOOD INPUT
1938 003224  055120    HLPTAB: .WORD   HELP1
1939 003226  055221            .WORD   HELP2
1940 003230  055314            .WORD   HELP3
1941 003232  055365            .WORD   HELP4
1942 003234  055436            .WORD   HELP5
1943 003236  055536            .WORD   HELP6
1944 003240  055651            .WORD   HELP7
1945 003242  055762            .WORD   HELP8
1946 003244  056052            .WORD   HELP9
1947 003246  056141            .WORD   HELP10
```

```
GLOBAL DATA SECTION

1948 003250  056232                        .WORD   HELP11
1949 003252  056330                        .WORD   HLP115
1950 003254  056435                        .WORD   HELP12
1951 003256  056542                        .WORD   HELP13
1952 003260  056641                        .WORD   HELP14
1953 003262  056740                        .WORD   HELP15
1954 003264  057043                        .WORD   HELP16
1955 003266  057132                        .WORD   HELP17
1956 003270  057235                        .WORD   HELP18
1957 003272  057305                        .WORD   HELP19
1958 003274  057410                        .WORD   HELP20
1959 003276  057466                        .WORD   HELP21
1960 003300  057551                        .WORD   HELP22
1961 003302  057652                        .WORD   HELP23
1962 003304  057752                        .WORD   HELP24
1963 003306  060102                        .WORD   HELP25
1964 003310  060166                        .WORD   HELP26
1965 003312  060272                        .WORD   HELP27
1966 003314  060374                        .WORD   HELP28
1967 003316  060513                        .WORD   HELP29
1968 003320  060563                        .WORD   HELP30
1969 003322  000000                HLPEND: .WORD   0
1970 003324  061536                MSGTAB: .WORD   MSGTY0                  ;MESSAGE TYPE ASCII ADDRESS TABLE
1971 003326  061544                        .WORD   MSGTY1
1972 003330  061551                        .WORD   MSGTY2
1973 003332  061557                        .WORD   MSGTY3
1974 003334  061564                        .WORD   MSGTY4
1975 003336  061571                        .WORD   MSGTY5
1976 003340  061577                        .WORD   MSGTY6
1977
1978                                ; THIS SECTION DEFINES THE DATA PATTERNS USED BY THE EXERCISER
1979
1980 003342                         MSGCNT::
1981 003342  000130                MSG0C:  .WORD   EMSG0-MSG00             ; THE NUMBER OF BYTES IN EACH MESSAGE
1982 003344  000001                MSG1C:  .WORD   EMSG1-MSG01
1983 003346  000001                MSG2C:  .WORD   EMSG2-MSG02
1984 003350  000001                MSG3C:  .WORD   EMSG3-MSG03
1985 003352  000001                MSG4C:  .WORD   EMSG4-MSG04
1986 003354  000100                MSG5C:  .WORD   EMSG5-MSG05
1987 003356  000000                MSG6C:  .WORD   0
1988
1989 003360                         MSGAD::
1990 003360  003376                        .WORD   MSG00
1991 003362  003526                        .WORD   MSG01
1992 003364  003527                        .WORD   MSG02
1993 003366  003530                        .WORD   MSG03
1994 003370  003531                        .WORD   MSG04
1995 003372  003532                        .WORD   MSG05
1996 003374  003632                        .WORD   OPSLBF
1997
1998 003376     040     041     042  MSG00:: .ASCII  \ !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ\
     003401     043     044     045
     003404     046     047     050
     003407     051     052     053
     003412     054     055     057
     003415     060     061     062
     003420     063     064     065
```

```
                003423  066     067     070
                003426  071     072     073
                003431  074     075     076
                003434  077     100     101
                003437  102     103     104
                003442  105     106     107
                003445  110     111     112
                003450  113     114     115
                003453  116     117     120
                003456  121     122     123
                003461  124     125     126
                003464  127     130     131
                003467  132
1999            003470  133     135     136             .ASCII  \[ ]↑ ABCDEFGHIJKLMNOPQRSTUVWXYZ\        ; ALPHANUMERIC
                003473  055     101     102
                003476  103     104     105
                003501  106     107     110
                003504  111     112     113
                003507  114     115     116
                003512  117     120     121
                003515  122     123     124
                003520  125     126     127
                003523  130     131     132
2000            003526                          EMSG0::
2001            003526  377                     MSG01:: .BYTE   377                     ; MESSAGE OF ALL ONES
2002            003527                          EMSG1::
2003            003527  000                     MSG02:: .BYTE   0                       ; MESSAGE OF ALL ZEROS
2004            003530                          EMSG2::
2005            003530  252                     MSG03:: .BYTE   252                     ; MESSAGE OF ALTERNATING ONES
2006            003531                          EMSG3::
2007            003531  125                     MSG04:: .BYTE   125                     ; MESSAGE OF ALTERNATING ZEROS
2008            003532                          EMSG4::
2009            003532                          MSG05::                                 ; CCITT 511 BIT TEST PATTERN
2010            003532  177603  157427  031011          .WORD   177603,157427,031011,047321,163715,105221
                003540  047321  163715  105221
2011            003546  143325  142304  040041          .WORD   143325,142304,040041,104116,052606,172334
                003554  104116  052606  172334
2012            003562  105025  123754  111337          .WORD   105025,123754,111337,111523,030030,145064
                003570  111523  030030  145064
2013            003576  137642  143531  063617          .WORD   137642,143531,063617,135075,066730,026575
                003604  135075  066730  026575
2014            003612  052012  053627  070071          .WORD   052012,053627,070071,151172,165044,031605
                003620  151172  165044  031605
2015            003626  166632  016147                  .WORD   166632,016147
2016            003632                          EMSG5::
2017            003632                          OPSLBF: .BLKB   66.              ;BUFFER FOR OPERATOR SELECTED MESSAGE TYPE
2018
2019
2020            003734  000000                  CFLAG:  .WORD   0                       ;ACTION ROUTINE CMD ARGUMENT FLAG
2021
2022                                            ;;CLOCK TABLES, EVENT LOG AND POINTERS
2023            003736  000000                  CLKCSR: .WORD   0                       ; CLOCK CSR ADDRESS
2024            003740  000000                  CLKBR:  .WORD   0                       ; CLOCK INTERRUPT LEVEL
2025            003742  000000                  CLKVEC: .WORD   0                       ; CLOCK INTERRUPT VECTOR
2026            003744  000000                  CLKHZ:  .WORD   0                       ; CLOCK'S FREQUENCY IN HERTZ
2027            003746  000000                  CLKEN:  .WORD   0                       ; CLOCK'S CSR VALUE TO INTRPT. ENABLE IT
2028
```

```
2029 003750  000000                TIMMIN: .WORD   0               ; PLACE TO KEEP TIME SINCE-START
2030 003752  000000                TIMSEC: .WORD   0
2031 003754  000000                TIMTCK: .WORD   0               ; PLACE TO KEEP NO. OF TICKS/SEC.
2032 003756  000000                TIMER1: .WORD   0               ; EVENT TIMER #1 (TICKS)
2033 003760  000000                TIMER2: .WORD   0               ; EVENT TIMER #2 (TICKS)
2034 003762  000000                TIMERS: .WORD   0               ; EVENT TIMER #3 (SECONDS)
2035                                       .EVEN
2036                               ;
2037                                 ;BUS ADDRESSES OF QNA HARDWARE REGISTERS
2038
2039 003764  000000                ROMAD1::.WORD   0               ;ON READ, GET QNA ADDRESS.  LOWER BYTE OF 1ST
2040 003766  000000                        .WORD   0               ;  THREE REGISTERS.
2041 003770  000000                RLSTLO::.WORD   0               ;LOW BYTE OF STARTING ADDRESS OF RECEIVE LIST
2042 003772  000000                RLSTHI::.WORD   0               ;HIGH BYTE OF STARTING ADDRESS OF RECEIVE LIST
2043                               ;
2044                               ; DEFAULT PHYSICAL NODE ADRRES
2045                               ;
2046 003774  000000                       DEPADR::         .WORD   0       ; PHYSICAL ADDRESS WORD 0
2047 003776  000000                                        .WORD   0       ; PHYSICAL ADDRESS WORD 1
2048 004000  000000                                        .WORD   0       ; PHYSICAL ADDRESS WORD 2
2049                               ;
2050                               ; ADDRESS OF ALL ZEROES, USED TO INIT THE TARGET ADDRESS TABLE
2051                               ;
2052 004002  000000                       NOADR::          .WORD   0
2053 004004  000000                                        .WORD   0
2054 004006  000000                                        .WORD   0
2055                               ;
2056                               ; CURRENT PHYSICAL ADDRESS
2057                               ;
2058 004010  000000                       PHYADR::         .WORD   0       ; PHYSICAL ADDRESS WORD 0
2059 004012  000000                                        .WORD   0       ; PHYSICAL ADDRESS WORD 1
2060 004014  000000                                        .WORD   0       ; PHYSICAL ADDRESS WORD 2
2061                               ;
2062                               ; BROADCAST ADDRESS   FF-FF-FF FF-FF FF
2063                               ;
2064 004016  177777                       BRDADR:          .WORD   -1
2065 004020  177777                                        .WORD    1
2066 004022  177777                                        .WORD   -1
2067                               ;
2068                               ; LOOP-SERVER MULTICAST ADDRESS   CF 00 00 00 00-00
2069                               ;
2070 004024  000317                       LPADR::          .WORD   317
2071 004026  000000                                        .WORD   0
2072 004030  000000                                        .WORD   0
2073                               ;
2074                               ; DECNET MULTICAST ADDRESS FOR CONSOLE ID   AB 00 00 02 00 00
2075                               ;
2076 004032  000253                       MCSTAD:          .WORD   253
2077 004034  001000                                        .WORD   1000
2078 004036  000000                                        .WORD   0
2079                               ;
2080                               ; RCV RING DESCRIPTOR ENTRY FOR SETUP MODE
2081                               ;
2082 004040                        LORTGT:                                 ; LOW ADRS OF TARGET ADRS TBL DESCRPTR
2083 004040  000000                       .WORD   0                        ; AGAIN FOR LAST ENTRY
2084 004042  100000                       .WORD   100000                   ; SET VALID BIT
2085 004044  023772                       .WORD   RCVTGT                   ; ADDRESS OF RECEIVE BUFFER
```

```
                                        .WORD   0
                                        .WORD   0
                                        .WORD   0
                                        .WORD   0                       ; JUST ENOUGH OF THE NEXT DESCRIPTOR
                                        .WORD   0                       ; TO INVALID LIST
                                ;
                                ; XMIT RING DESCRIPTOR ENTRY FOR SETUP MODE
                                ;
004060                          LOXTGT:                                 ; LOW ADRS OF TARGET ADRS TBL DESCRPTR
004060  000000                          .WORD   0                       ; AGAIN FOR LAST ENTRY
004062  100000                          .WORD   100000                  ; SET VALID BIT
004064  023570                          .WORD   TGTADR                  ; ADDRESS OF TARGET FIELD
004066  000000                          .WORD   0
004070  000000                          .WORD   0
004072  000000                          .WORD   0
004074  000000                          .WORD   0                       ; JUST ENOUGH OF THE NEXT DESCRIPTOR
004076  000000                          .WORD   0                       ; TO INVALID LIST
                                ;
                                ; CITIZENSHIP TEST SETUPS

                                  ;RECEIVE BUFFERS
000100                          SMLSIZ  ==      64.                     ;SMALLEST SIZE IN BYTES OF ETHERNET PACKET
000151                          RBUFNR  ==      105.                    ;NUMBER OF RECEIVE BUFFERS
004100                          RBUFB:  .BLKB   RBUFNR*SMLSIZ           ;100 RECEIVE BUFFERS, 64 BYTES EACH.
                                ;
                                  ;RECEIVE DESCRIPTORS
                                ;
                                ;THERE ARE RBUFNR+1 DESCRIPTORS.  ALL OF THEM EXCEPT THE LAST ONE
                                ;  ARE DATA DESCRIPTORS, I.E. THE ONLY CHAIN DESCRIPTOR IS THE LAST
                                ;  ONE TO CREATE A RING.  COMMUNICATIONS WITH THE TESTER IS
                                ;  HALF-DUPLEX, I.E. PACKETS ARE RECEIVED ONE AT A TIME UNTIL THE
                                ;  COMPLETE MESSAGE IS RECEIVED.
                                ;
000014                          RDESSZ  ==      12.                     ;SIZE IN BYTES OF EACH RECEIVE DESCRIPTOR
002354                          RDESAL  ==      RDESSZ*RBUFNR           ;SIZE IN BYTES OF ALL DESCRIPTORS
021200                          RDESC:  .BLKB   RDESAL                  ;STORAGE FOR RBUFNR RECEIVE DESCRIPTORS
023554                                  .BLKB   RDESSZ                  ;  PLUS CHAIN DESCRIPTOR TO GIVE A RING
                                        .EVEN
                                ;
                                ;
                                  ;CITIZENSHIP WORK BUFFER
                                ;
                                ;THE BOOT/DIAG ROM LOADS INTO THE FIRST 4KB OF THE RECEIVE BUFFERS.  THE 4KW
                                ;  WORK BUFFER EXTENDS FROM THERE THROUGH THE TRANSMIT BUFFERS INTO THE
                                ;  STORAGE AREA THAT IS CREATED BELOW.
014100                          CITWORK ==      RBUFB+4096.             ;BEGINNING OF 4KW WORK BUFFER
024100                          CITWEND ==      CITWORK+4096.           ;END+1 OF WORK BUFFER
                                ;
                                ;
                                ; TABLE OF TARGET ADDRESSES USED IN SETUP MODE
                                ;
023570                          TGTADR:
023570                                  .BLKB   8.                      ; BYTE 0 OF ALL ADDRESSES 1
023600                                  .BLKB   8.                      ; BYTE 1   "    "
023610                                  .BLKB   8.                      ; BYTE 2   "
023620                                  .BLKB   8.                      ; BYTE 3   "
023630                                  .BLKB   8.                      ; BYTE 4
```

```
2143 023640                                .BLKB   8.                      ; BYTE 5 '    '
2144 023650                                .BLKB   8.                      ; BYTE 6      '       '
2145 023660                                .BLKB   8.                      ; BYTE 7 '    "    '   '
2146 023670                                .BLKB   8.                      ; BYTE 0 OF ALL ADDRESSES 8   14
2147 023700                                .BLKB   8.                      ; BYTE 1 '          "   '
2148 023710                                .BLKB   8.                      ; BYTE 2 '    "     "
2149 023720                                .BLKB   8.                      ; BYTE 3 "    "     "   '
2150 023730                                .BLKB   8.                      ; BYTE 4 '    '     '   '
2151 023740                                .BLKB   8.                      ; BYTE 5 "    "         '
2152 023750                                .BLKB   8.                      ; BYTE 6 "    '     '   "
2153 023760                                .BLKB   8.                      ; BYTE 7      "     '
2154 023770                        TGTSIZ:
2155 023770  000200                        .WORD   TGTSIZ - TGTADR         ; SIZE OF THE TARGET ADDRESS
2156                                ;
2157                                ; RECEIVE BUFFER FOR TARGET ADDRESSES LOOPED BACK IN SETUP MODE
2158                                ;
2159 023772                        RCVTGT:
2160 023772                                .BLKB   8.                      ; BYTE 0 OF ALL ADDRESSES 1 7
2161 024002                                .BLKB   8.                      ; BYTE 1 '    "     '   "
2162 024012                                .BLKB   8.                      ; BYTE 2            "
2163 024022                                .BLKB   8.                      ; BYTE 3 "    '     '   '
2164 024032                                .BLKB   8.                      ; BYTE 4 "    "     '   '
2165 024042                                .BLKB   8.                      ; BYTE 5 '    "     "   "
2166 024052                                .BLKB   8.                      ; BYTE 6 "    "     "   '
2167 024062                                .BLKB   8.                      ; BYTE 7 "    "     "   '
2168 024072                                .BLKB   8.                      ; BYTE 0 OF ALL ADDRESSES 8   14
2169 024102                                .BLKB   8.                      ; BYTE 1 "    "     "   '
2170 024112                                .BLKB   8.                      ; BYTE 2 "    "     "   "
2171 024122                                .BLKB   8.                      ; BYTE 3 '    "     '   "
2172 024132                                .BLKB   8.                      ; BYTE 4 "          '   '
2173 024142                                .BLKB   8.                      ; BYTE 5 "    '     '   '
2174 024152                                .BLKB   8.                      ; BYTE 6 "    '     "
2175 024162                                .BLKB   8.                      ; BYTE 7 "    '     '   "
2176                                ;
2177                                ; TABLE OF  START ADDRESS OF RECEIVE RING BUFFERS
2178                                ;
2179 024172                        RRNGTB::
2180 024172  025674                        .WORD RRG001
2181 024174  026774                        .WORD RRG002
2182 024176  030074                        .WORD RRG003
2183 024200  031174                        .WORD RRG004
2184 024202  032274                        .WORD RRG005
2185 024204  033374                        .WORD RRG006
2186 024206  034474                        .WORD RRG007
2187 024210  035574                        .WORD RRG010
2188                                ;
2189                                ;       TABLE OF START ADDRESS OF TRANSMIT RING BUFFERS
2190                                ;
2191 024212                        XRNGTB::
2192 024212  024574                        .WORD   XRG001
2193                                ;                       POINTERS TO DESCRIPTOR RING ENTRIES
2194                                ;
2195 024214  024236                XRGSRT::        .WORD   XRING           ; FIRST ENTRY IN TRANSMIT RING
2196 024216  024260                RRGSRT::        .WORD   RRING           ; FIRST ENTRY IN RECEIVE RING
2197 024220  024236                XRGCUR::        .WORD   XRING           ; CURRENT ENTRY IN TRANSMIT RING
2198 024222  024260                RRGCUR::        .WORD   RRING           ; CURRENT ENTRY IN RECEIVE RING
2199 024224  024236                XRGNXT::        .WORD   XRING           ; NEXT ENTRY IN TRANSMIT RING
```

```
                        ;GLOBAL DATA SECTION

2200 024226  024260    RRGNXT::        .WORD   RRING           ; NEXT ENTRY IN RECEIVE RING
2201 024230  024236    XRGLST::        .WORD   XRING           ; LAST ENTRY IN TRANSMIT RING
2202 024232  024544    RRGLST::        .WORD   RRINGH          ; LAST ENTRY IN RECEIVE RING
2203 024234  024544    RRGPRV::        .WORD   RRINGH          ; PREVIOUS DESCRIPOTR POINTER
2204                    ;
2208 024236             XRING::
2209 024236                     RNGFRM  XRG00,B,0
     024236  000000             .WORD   0                       ; FLAG WORD
     024240  100000             .WORD   100000                  ; SET VALID BIT (DSCR/HIGH ADDRESS WRD)
     024242  024574             .WORD   XRG001                  ; SEGMENT BUFFER ADDRESS
     024244  000000             .WORD   0                       ; BUFFER SIZE
     024246  000000             .WORD   0                       ; STATUS WORD 1
     024250  000000             .WORD   0                       ; STATUS WORD 2
2210 024252             XRINGH:
2211 024252             XRGINV:
2212 024252  000000             .WORD   0                       ; JUST ENOUGH OF THE NEXT DESCRIPTOR
2213 024254  000000             .WORD   0                       ; TO INVALIDATE LIST
2214 024256             RNGSIZ:
2215 024256  000004             .WORD   . - XRINGH              ; SIZE OF DESCRIPTORS
2219 024260             RRING::
2220         000007              .REPT 7
2221                             .NLIST
2222                             RNGFRM RRG00,B,-RPKLEN/2       ; ENTER BUFF ADD, BUF OFFSET, BUF SIZE
2223                             .LIST
2224                             .ENDR
2225                    ;
2226 024404  000000             .WORD   0                       ; AGAIN FOR LAST ENTRY (FLAG WRD)
2227 024406  100000             .WORD   100000                  ; DESCR/HIGH ADDRESS WORD
2228 024410  035574             .WORD   RRG010                  ; LOW ADDRESS WORD
2229 024412  177340             .WORD   -RPKLEN/2               ; BUFFER SIZE (TWO'S COMPLEMENT)
2230 024414  000000             .WORD   0                       ; STATUS WRD 1
2231 024416  000000             .WORD   0                       ; STATUS WRD 2
2232                    ;
2236                    ;
2237         000007              .REPT 7
2238                             .NLIST
2239                             RNGFRM RRG01,B,-RPKLEN/2               ; ENTER BUFF ADD, BUF OFFSET, BUF SIZE
2240                             .LIST
2241                             .ENDR
2242 024544             RRINGH::
2243 024544  000000             .WORD   0                       ; AGAIN FOR LAST ENTRY (FLAG WRD)
2244 024546  000000             .WORD   0                       ; LEAVE THIS ONE INVALID
2245 024550  046574             .WORD   RRG020                  ; LOW ADDRESS WORD
2246 024552  177340             .WORD   -RPKLEN/2               ; BUFFER SIZE (TWO'S COMPLEMENT)
2247 024554  000000             .WORD   0                       ; STATUS WRD 1
2248 024556  000000             .WORD   0                       ; STATUS WRD 2
2249 024560             RRGCHN::
2250 024560  000000             .WORD   0
2251 024562  140000             .WORD   140000                  ; DESCRIPTOR TO CHAIN TO BEGINNING OF RING
2252 024564  024260             .WORD   RRING                   ; ADDRESS OF START OF DESCRIPTOR RING
2253 024566  000000             .WORD   0                       ; BYTE COUNT (UNNECESSARY)
2254 024570  000000             .WORD   0                       ; STAT1
2255 024572  000000             .WORD   0                       ; STAT2
2256                    ;
2257                    ;
2258 024574             XRG001::        .BLKB   XPKLEN          ; XMIT RING BUFFER
2259                    ;
```

```
2260 025674                         RRG001::        .BLKB   RPKLEN          ; RECEIVE RING BUFFERS
2261 026774                         RRG002::        .BLKB   RPKLEN
2262 030074                         RRG003::        .BLKB   RPKLEN
2263 031174                         RRG004::        .BLKB   RPKLEN
2264 032274                         RRG005::        .BLKB   RPKLEN
2265 033374                         RRG006::        .BLKB   RPKLEN
2266 034474                         RRG007::        .BLKB   RPKLEN
2267 035574                         RRG010::        .BLKB   RPKLEN
2268 036674                         RRG011::        .BLKB   RPKLEN
2269 037774                         RRG012::        .BLKB   RPKLEN
2270 041074                         RRG013::        .BLKB   RPKLEN
2271 042174                         RRG014::        .BLKB   RPKLEN
2272 043274                         RRG015::        .BLKB   RPKLEN
2273 044374                         RRG016::        .BLKB   RPKLEN
2274 045474                         RRG017::        .BLKB   RPKLEN
2275 046574                         RRG020::        .BLKB   RPKLEN
2276                                ;
2277                                ; CONSOLE ID INFORMATION
2278                                ;
2279 047674                         IDTDAT:
2280 047674                                 .BLKB   6       ; DESTINATION ADDRESS FIELD
2281 047702                                 .BLKB   6       ; SOURCE ADDRESS FIELD
2282 047710  001140                         .WORD   1140    ; CONSOLE ID TYPE CODE
2283 047712  000034                         .WORD   28.     ; NUMBER OF VALID DATA BYTES TO FOLLOW
2284 047714     007                         .BYTE   7       ; SYSTEM ID FUNCTION CODE
2285 047715     000                         .BYTE   0       ; PAD BYTE
2286 047716  046507                         .WORD   46507   ; RECEIPT NUMBER  ;; N.M. (CHANGED FROM 51115 TO 46507)
2287 047720  000000                         .WORD   0       ; MOP VERSION TYPE
2288 047722     000                         .BYTE   0       ; MOP VERSION LENGTH
2289 047723     000                         .BYTE   0       ; MOP VERSION VERSION
2290 047724     000                         .BYTE   0       ; MOP VERSION ECO
2291 047725     000                         .BYTE   0       ; MOP VERSION USER ECO
2292 047726  000002                         .WORD   2       ; FUNCTION TYPE
2293 047730     001                         .BYTE   1       ; FUNCTION LENGTH
2294 047731     000                         .BYTE   0       ; MAINTENANCE FUNCTIONS SUPPORTED 1
2295 047732     000                         .BYTE   0       ; MAINTENANCE FUNCTIONS SUPPORTED 2
2296 047733     007                         .BYTE   7       ; HARDWARE ADDRESS TYPE
2297 047734     000                         .BYTE   0       ; SECOND BYTE OF ADDRESS TYPE
2298 047735     006                         .BYTE   6       ; HARDWARE ADDRESS LENGTH
2299 047736                                 .BLKB   6       ; HARDWARE ADDRESS VALUE
2300 047744     144                         .BYTE   100.    ; DEVICE TYPE FIELD
2301 047745     000                         .BYTE   0       ; DEVICE TYPE FIELD BYTE 2
2302 047746     001                         .BYTE   1       ; LENGTH OF DEV TYPE FIELD
2303 047747     005                         .BYTE   5       ; DEQNA DEVICE TYPE
2304 047750                                 .BLKB   65.-49. ; PAD THE REST OF THE PACKET ;; N.M. (CHANGED 43 TO 49)
2305                                                        ; ;; N.M. (USED TO BE .EVEN HERE)
2306 047770                         IDTSIZ:
2307 047770  000074                         .WORD   . - IDTDAT
2308                                ;
2309                                ;INFORMATION ABOUT THE CURRENT UNIT AS OBTAINED FROM THE HARDWARE P TABLE
2310                                ;
2311                                ;       PTABLE INFO STORAGE FOR UNIT 0
2312                                ;
2313 047772  000000                 QNAADO::        .WORD   0       ;DEVICE ADDRESS FROM PTABLE
2314 047774  000000                 QNAVCO::        .WORD   0       ;VECTOR ADDRESS FROM PTABLE
2315 047776  000000                 QNAPRO::        .WORD   0       ;PRIORITY FROM PTABLE
2316                                ;
```

```
                                ;       REGISTER CONTENT HOLDING LOCATIONS FOR UNIT NUMBER 0
                                ;
050000                          NETADO::        .BLKW   2               ; HOLDS CONTENTS OF STATION ADDRESS ROM
050004                          CSRO::          .BLKW   1               ; HOLDS CONTENTS OF CSR REGISTER
                                ;
                                ;       PTABLE INFO STORAGE FOR UNIT 1
                                ;
050006  000000                  QNAAD1::        .WORD   0       ;CSR ADDRESS FROM PTABLE
050010  000000                  QNAVC1::        .WORD   0       ;VECTOR ADDRESS FROM PTABLE
050012  000000                  QNAPR1::        .WORD   0       ;PRIORITY FROM PTABLE
                                ;
                                ;       CSR LOCATIONS FOR UNIT NUMBER 1
                                ;
050014                          NETAD1::        .BLKW   2               ; HOLDS CONTENTS OF STATION ADDRESS ROM
050020                          CSR1::          .BLKW   1               ; CONTENTS OF CSR REGISTER
                                ;
                                ;
                                ;
050022  000000                  FRESIZ::.WORD   0       ;POINTER TO WORD CONTAINING SIZE OF FREE MEMORY
050024  000000                  FREMEM::.WORD   0       ;POINTER TO FREE MEMORY SPACE
050026  000000                  UNIT::  .WORD   0       ;CURRENT UNIT NUMBER BEING TESTED
                                ;
                                ;       SUMMARY DATA COUNTERS
                                ;
050030  000000                  S.REC::         .WORD   0               ; MESSAGES RECEIVED
050032  000000                  S.NREC::        .WORD   0               ; MESSAGES NOT RECEIVED
050034  000000                  S.LEN::         .WORD   0               ; LENGTH ERRORS
050036  000000                  S.COMP::        .WORD   0               ; COMPARE ERRORS
050040  000000                  S.BYTE::        .WORD   0               ; BYTES COMPARED
050042  000000                  S.XFER::        .WORD   0               ; BYTES TRANSFERED
                                ;
                                ;       SUMMARY LINK STATISTICS
                                ;
050044                          UCB12:                                  ; THIS AREA BUFFERS THE COUNTERS WHEN PRINTED
050044                                  .BLKB   100                     ; SIZE OF BUFFER FOR COUNTER
050144                          CNTRS:
050144  000076                          .WORD   CNTEND - CNTRS  ; SIZE OF COUNTER DATA BASE
050146  000000                          .WORD   0               ; SECONDS SINCE LAST ZEROED
050150                                  .BLKW   2               ; PACKETS RECEIVED
050154                                  .BLKW   2               ; MULTICAST PACKETS RECEIVED
050160  000000                          .WORD   0               ; RECEIVE ERROR REASON MASK
050162  000000                          .WORD   0               ; NUMBER OF PACKETS RECEIVED WITH ERRS
050164                                  .BLKW   2               ; NUMBER OF DATA BYTES RECEIVED
050170                                  .BLKW   2               ; MULTICAST DATA BYTES RECEIVED
050174  000000                          .WORD   0               ; RCV PACKETS LOST - INTERNAL BUFF ERR
050176  000000                          .WORD   0               ; RCV PACKETS LOST - LOCAL BUFFER ERR
050200                                  .BLKW   2               ; PACKETS XMITTED
050204                                  .BLKW   2               ; MULTICAST PACKETS XMITTED
050210                                  .BLKW   2               ; XMIT SUCCESS AFTER 3 OR MORE TRIES
050214                                  .BLKW   2               ; XMIT SUCCESS AFTER 2 TRIES
050220                                  .BLKW   2               ; XMISSIONS THAT HAD TO BE DEFERRED
050224                                  .BLKW   2               ; BYTES TRANSMITTED
050230                                  .BLKW   2               ; NUMBER OF MULTICAST BYTES XMITTED
050234  000000                          .WORD   0               ; ABORTED XMISSIONS, REASON MASK
050236  000000                          .WORD   0               ; COUNT OF ABORTED TRANSMISSIONS
050240  000000                          .WORD   0               ; COUNT OF XMIT COLLISON CHECK FAILURES
050242                          CNTEND:
```

```
                                        ;
                                        ;       DEQNA DRIVER AND ASSOCIATED SUBROUTINES DATA
                                        ;
050242  000000                  CITTST: .WORD   0               ; HOLDS STARTING ADDRESS OF CITIZENSHIP TEST AFTER LOADING F
ROM ROM
050244  000000                  CITSUM: .WORD   0               ; HOLDS ADDRESS OF THE ROM CHECKSUM AFTER LOADING
050246  000000                  CSRBUF: .WORD   0               ; BUFFERS THE CSR, BIS AND BIC HAPPEN TO THIS
                                                                ; LOCATION WHICH IS THEN MOVED TO THE CSR.
                                                                ; PREVENTS LOSING BITS THAT ARE "WRITE A 1 TO
                                                                ; CLEAR"
050250  000000                  FLAG1::.WORD    0                       ; FLAG TO INDICATE IF CITIZ. TEST HAS RUN.
050252  000000                  FATFLG::.WORD   0                       ; FATAL ERROR FLAG
050254  000000                  PCEFLG::.WORD   0                       ; PORT COMMAND ERROR FLAG
050256  000000                  NIRCNT::.WORD   0                       ; QNA RECEIVE MESSAGE COUNTER
050260  000000                  XFLAG:: .WORD   0                       ; FRAME TRANSMITTED FLAG
050262  000000                  DNIFLG::.WORD   0                       ; DONE INTERRUPT FLAG
050264  000000                  RBFCNT::.WORD   0                       ; RECEIVE BUFFERS LOST COUNTER
050266  000000                  BCOUNT::.WORD   0                       ; UNEXPLAINED INTERRUPTS COUNTER
050270  000000                  ERRFLG::.WORD   0                       ; ERROR FLAG
050272  000000                  TIMOUT::.WORD   0                       ; TIME OUT COUNTER
050274  000000                  RETRYS::.WORD   0                       ; COUNTER FOR FRAMES FAILING DUE TO RTRY ERROR
050276  000000                  RCVERR::.WORD   0                       ; COUNTS NO. OF BUFFERS RECEIVED WITH  ERRORS
050300  000000                  RCVBUF::.WORD   0                       ; COUNTS NO. OF GOOD BUFFERS RECEIVED
050302  010000                  ROMSIZ::        .WORD   4096.           ; SIZE OF BOOT/DIAGNOSTIC ROM IN BYTES
050304  000000                  COUNT:: .WORD   0                       ; USED IN BLDBUF SUBROUTINE AS COUNTER
050306  000220                  PROTO0::.WORD   000220                  ; PROTOCOL TYPE FOR LOOPBACK MESSAGES
050310  001140                  PROTO2::.WORD   001140                  ; PROTOCOL TYPE FOR REMOTE CONSOLE
050312  000000                  SLFTST: .WORD   0                       ; EXECUTE SELF TEST FLAG
050314  000000                  TEMP::  .WORD   0                       ; USED IN XMIT AS TEMPORARY STORAGE
050316  000000                  TEMP1:: .WORD   0                       ; USED FOR TEMPORARY STORAGE
050320  000000                  TEMP2:: .WORD   0                       ; USED FOR TEMPORARY STORAGE
050322  000000                  TEMP3:: .WORD   0                       ; USED FOR TEMPORARY STORAGE
050324  000000                  XFER::  .WORD   0                       ; STORES 'BYTES TRANSFERED'
050326  000000                  CPYCNT::.WORD   0                       ; 'NO. OF COPIES' COUNTER FOR LOOPING
050330  000000                  PCCALL::.WORD   0                       ; STORES PC OF CALLING ROUTINE FOR ERROR REPORTS
050332  001100                  BUFLEN::.WORD   XPKLEN                  ; STORES TRANSMIT BUFFER LENGTH
050334  000000                  CMPBUF::.WORD   0                       ; STORES LOCATION OF DATA BUFFER TO BE COMPARED
050336  000000                  LAPCNT: .WORD   0                       ; COUNTS HOW MANY LAPS "CURRENT RECEIVE
                                                                        ; DESCRIPTOR" POINTER IS AHEAD OF THE
                                                                        ; "NEXT RECEIVE DESCRIPTOR" POINTER
                                        ;
                                        ; REQUEST ID MESSAGE FORMAT
                                        ;
050340                          REQID::
050340  000003                          .WORD   3                       ; BYTE COUNT (=3 FOR REQUEST ID)
050342  000005                          .WORD   IDFUNC                  ; FUNCTION CODE FOR REQUEST ID
050344  051115                          .WORD   "MR                     ; RECEIPT NUMBER
                                        ;
                                        ; LOOP DIRECT MESSAGE
                                        ;
                                                .EVEN
050346                          LOPDIR::
050346  000000                          .WORD   0                       ; SKIP COUNT
050350  000002                          .WORD   FORWRD                  ; FUNCTION = FORWARD DATA
050352  000000  000000  000000          .WORD   0,0,0                   ; LOCAL NODE ADDRESS
050360  000001                          .WORD   1                       ; FUNCTION = REPLY
050362  000000  000000  000000          .WORD   0,0,0                   ; LOCAL NODE ADDRESS
                                        ;
```

```
2431                                    ; TRANSMIT ASSIST MESSAGE
2432                                    ;
2433 050370                             TASIST::
2434 050370  000000                             .WORD   0                       ; SKIP COUNT
2435 050372  000002                             .WORD   FORWRD                  ; FUNCTION = FORWARD DATA
2436 050374  000000  000000  000000             .WORD   0,0,0                   ; TRANSMIT ASSIST ADDRESS
2437 050402  000002                             .WORD   FORWRD                  ; FUNCTION = FORWARD DATA
2438 050404  000000  000000  000000             .WORD   0,0,0                   ; LOCAL NODE ADDRESS
2439 050412  000001                             .WORD   REPLY                   ; FUNCTION = REPLY
2440 050414  000000  000000  000000             .WORD   0,0,0                   ; LOCAL NODE ADDRESS
2441                                    ;
2442                                    ; RECEIVE ASSIST MESSAGE
2443                                    ;
2444 050422                             RASIST::
2445 050422  000000                             .WORD   0                       ; SKIP COUNT
2446 050424  000002                             .WORD   FORWRD                  ; FUNCTION = FORWARD DATA
2447 050426  000000  000000  000000             .WORD   0,0,0                   ; TRANSMIT ASSIST ADDRESS
2448 050434  000002                             .WORD   FORWRD                  ; FUNCTION = FORWARD DATA
2449 050436  000000  000000  000000             .WORD   0,0,0                   ; LOCAL NODE ADDRESS
2450 050444  000001                             .WORD   REPLY                   ; FUNCTION = REPLY
2451 050446  000000  000000  000000             .WORD   0,0,0                   ; LOCAL NODE ADDRESS
2452                                    ;
2453                                    ; FULL ASSIST MESSAGE
2454                                    ;
2455 050454                             FASIST::
2456 050454  000000                             .WORD   0                       ; SKIP COUNT
2457 050456  000002                             .WORD   FORWRD                  ; FUNCTION = FORWARD DATA
2458 050460  000000  000000  000000             .WORD   0,0,0                   ; TARGET NODE ADDRESS
2459 050466  000002                             .WORD   FORWRD                  ; FUNCTION = FORWARD DATA
2460 050470  000000  000000  000000             .WORD   0,0,0                   ; ASSIST NODE ADDRESS
2461 050476  000002                             .WORD   FORWRD                  ; FUNCTION = FORWARD DATA
2462 050500  000000  000000  000000             .WORD   0,0,0                   ; LOCAL NODE ADDRESS
2463 050506  000001                             .WORD   REPLY                   ; FUNCTION = REPLY
2464 050510  000000  000000  000000             .WORD   0,0,0                   ; LOCAL NODE ADDRESS
```

```
                                .SBTTL  COMMAND LINE ACTION TREE
                        ;SAMPLE CLI TREE NODE  (ALWAYS AT LEAST 1 WORD)
                        ;
                        ; ------------------   --
                        ; ! ACTION  ! CHAR CODE !
                        ; ------------------ -----
                        ; !  MISS DISPLACEMENT  !       ONLY IF "MISS" ARGUMENT DEFINED
                        ; - ---------- ---- ----
                        ; !  NEXT MODE DISPLMNT !       ONLY IF "ASCII" ARGUMENT DEFINED
                        ; --------------------  -
                        ; !  ASCIZ MATCH STRING !       ONLY IF "ASCII" ARGUMENT DEFINED
                        ; !       (.EVEN)      !
                        ; ------------  ---  - -
                        ;
                        ; FIRST KEYWORD
                        ;
050516                  CLITRE:
050516                          CLI     CLISPA,0,N10$                   ;SKIP ANY LEADING SPACES
050522                  N10$:   CLI     <'?>,HELP,N12$                  ;IS THE FIRST NON-SP CHAR. A  ?
050526                          CLI     CLIEXI,0                        ;  IF YES DO "HLP" AND EXIT
050530                  N12$:   CLI     CLISTR,HELP,N14$,<'HELP'>       ;ELSE IS FIRST WORD A "HELP'
050544                          CLI     CLIEXI,0                        ;  IF YES DO "HLP" AND EXIT
050546                  N14$:   CLI     CLISTR,NOTNUF,N16$,<'NODE'>     ;ELSE IS FIRST WORD A "NODE"
050562                          CLI     CLIBR,0,N80$                    ;  IF YES, BR N80$
050566                  N16$:   CLI     CLISTR,BUILD,N18$,<'BUILD'>     ;ELSE IS FIRST WORD A "BUILD'
050602                          CLI     CLIEXI,0                        ;  IF YES DO "BUILD" AND EXIT
050604                  N18$:   CLI     CLISTR,NOTNUF,N20$,<'RUN'>      ;ELSE IS FIRST WORD A "RUN"
050616                          CLI     CLIBR,0,N180$                   ;  IF YES, BR N180$
050622                  N20$:   CLI     <'S>,NOTNUF,N25$                ;ELSE IS FIRST CHAR. A "S"
050626                          CLI     CLISTR,0,N22$,<'HOW'>           ;  IF YES IS REST OF WORD "HOW"
050640                          CLI     CLIBR,0,N100$                   ;    IF YES,BR N100$
050644                  N22$:   CLI     CLISTR,SUMMRY,N23$,<'UMMARY'>   ;  ELSE IS REST OF WORD "UMMARY"
050662                          CLI     CLIEXI,0                        ;    IF YES, DO "SUMM" AND EXIT
050664                  N23$:   CLI     CLISTR,CSAVE,N24$,<'AVE'>       ;  ELSE IS REST OF WORD "AVE"
050676                          CLI     CLIEXI,0                        ;    IF YES, DO "SAVE" AND EXIT
050700                  N24$:   CLI     CLIERR,0                        ;  ELSE "ILL COMMAND"
050702                          CLI     CLIEXI,0                        ;  EXIT
050704                  N25$:   CLI     CLISTR,NOTNUF,N26$,<'CLEAR'>    ;ELSE IS FIRST WORD A "CLEAR'
050720                          CLI     CLIBR,0,N120$                   ;  IF YES, BR N120$
050724                  N26$:   CLI     CLISTR,NOTNUF,N28$,<'IDENTIFY'> ;ELSE IS FIRST WORD "IDENTIFY'
050744                          CLI     CLIBR,0,N140$                   ;  IF YES, GET ADDRS, BR N140$
050750                  N28$:   CLI     CLISTR,NOTNUF,N29$,<'MESSAGE'>  ;ELSE IS FIRST WORD "MESSAGE
050766                          CLI     CLIBR,0,N160$                   ;  IF YES, BR N160$
050772                  N29$:   CLI     CLISTR,CUNSAV,N30$,<'UNSAVE'>   ;ELSE IS FIRST WORD "UNSAVE'
051010                          CLI     CLIBR,0,N210$                   ;  IF YES, BR TO N210$
051014                  N30$:   CLI     CLISTR,EXIT,N31$,<'EXIT'>       ;ELSE IS FIRST WORD 'EXIT'
051030                          CLI     CLIEXI,0                        ;  IF YES EXIT
051032                  N31$:   CLI     CLISTR,NOTNUF,N32$,<'FUNCTION'> ;ELSE IS FIRST WORD "FUNCTION
051052                          CLI     CLIBR,0,N200$                   ;  IF YES, BR N200$
051056                  N32$:   CLI     CLISTR,RSPOND,N33$,<'RESPOND'>  ;ELSE IS FIRST WORD "RESPOND
051074                          CLI     CLIEXI,0                        ;  IF YES LEAVE
051076                  N33$:   CLI     CLIERR,0                        ;OTHERWISE 'ILL CMD".
051100                          CLI     CLIEXI,0                        ;  EXIT
                        ;
                        ; SECOND KEYWORD (ADR/TYPE) FOR NODE COMMAND
                        ;
051102                  N80$:   CLI     CLISPA,0,N81$                   ;SKIP ANY LEADING SPACES
```

```
2523 051106                     N81$:   CLI     CLIBR,CSAVR4,N82$                ;SAVE STRING POINTER LOCATION
2524 051112                     N82$:   CLI     CLIBR,NODE,N90$                  ;PARSE THROUGH ADDRESS,CHECK
2525                                                                             ;FOR TARGET OR ASSIST, DO NODE
2526 051116                     N90$:   CLI     CLIBIF,0,N32$                    ;TAKE ERROR BRANCH IF ERROR EXISTS
2527 051122                     N95$:   CLI     CLIEXI,0                         ;EXIT
2528                                     ;
2529                                     ; SECOND KEYWORD FOR SHOW COMMAND
2530                                     ;
2531 051124                     N100$:  CLI     CLISPA,0,N101$                   ;SKIP LEADING SPACES
2532 051130                     N101$:  CLI     CLISTR,CNODE,N102$,<'NODES'>     ;IS NEXT WORD "NODES'
2533 051144                             CLI     CLIBR,0,N110$                    ;  IF YES, SET FLAG, BR N110$
2534 051150                     N102$:  CLI     CLISTR,CSHMSG,N104$,<'MESSAGE'>  ;ELSE IS NEXT WORD "MESSAGE"
2535 051166                             CLI     CLIBR,0,N110$                    ;  IF YES ,SET FLAG, BR N110$
2536 051172                     N104$:  CLI     CLISTR,CCNTR,N106$,<'COUNTERS'>  ;ELSE IS NEXT WORD "COUNTERS'
2537 051212                             CLI     CLIBR,0,N110$                    ; GO TO COUNTERS ROUTINE
2538 051216                     N106$:  CLI     CLIBR,0,N32$                     ;ELSE  ILL COMMAND"
2539 051222                     N110$:  CLI     CLIEXI,0                         ;EXIT
2540                                     ;
2541                                     ; SECOND KEYWORD FOR CLEAR COMMAND
2542                                     ;
2543 051224                     N120$:  CLI     CLISPA,0,N121$                   ;SKIP LEADING SPACES
2544 051230                     N121$:  CLI     CLISTR,0,N130$,<'NODE'>          ;IS NEXT WORD "NODE"
2545 051244                             CLI     CLISPA,0,N122$                   ;  IF YES SKIP SPACES
2546 051250                     N122$:  CLI     <'/>,CSAVR4,N32$                 ;  LOOK FOR DELIMETER, ELSE "ILL COM'
2547 051254                             CLI     <'A>,0,N123$                     ;  IS NEXT CHAR. AN "A"
2548 051260                             CLI     CLISTR,CNODAL,N124$,<'LL'>       ;    IF YES, IS WORD "ALL"
2549 051272                             CLI     CLIBR,0,N135$                    ;      IF YES, SET FLAG,BR N135$
2550 051276                     N123$:  CLI     <'N>,0,N124$                     ;  ELSE IS NEXT CHAR. AN "N'
2551 051302                             CLI     CLIDEC,0,N32$                    ;    IF YES, STORE NODE LOGICAL NAME
2552 051306                             CLI     CLIBR,CNDLOG,N135$               ;    BR TO CLR. NODE LOGICAL ROUTINE
2553 051312                     N124$:  CLI     CLIBR,CEXADR,N126$               ;  ELSE, EXTRACT ADDRESS
2554 051316                     N126$:  CLI     CLIBR,CNDADR,N135$               ;  SET FLAG, BR N135$
2555 051322                     N130$:  CLI     CLISTR,CCLMSG,N132$,<'MESSAGE'>  ;ELSE IS NEXT WORD "MESSAGE
2556 051340                             CLI     CLIBR,0,N135$                    ;  IF YES, SET FLAG, BR N135$
2557 051344                     N132$:  CLI     CLISTR,CCLSUM,N134$,<'SUMMARY'>  ;ELSE IS NEXT WORD "SUMMARY'
2558 051362                             CLI     CLIBR,0,N135$                    ;  IF YES, CLEAR TABLE AND EXIT
2559 051366                     N134$:  CLI     CLIERR,0                         ;ELSE, "ILL COMMAND".
2560 051370                     N135$:  CLI     CLIEXI,0                         ;EXIT
2561                                     ;
2562                                     ; ADDRESS FOR IDENTIFY COMMAND
2563                                     ;
2564 051372                     N140$:  CLI     CLISPA,0,N141$                   ;SKIP LEADING SPACES
2565 051376                     N141$:  CLI     CLIBR,CSAVR4,N142$               ;SAVE POINTER TO FIRST CHAR. OF ADDRESS
2566 051402                     N142$:  CLI     CLIALN,0,N32$                    ;CHECK THAT ADDRESS HAS LEGAL CHAR.S
2567 051406                             CLI     CLIBR,CEXADR,N143$               ;GET ADDRESS
2568 051412                     N143$:  CLI     CLIEXI,IDENT                     ;DO 'IDENTIFY", EXIT
2569                                     ;
2570                                     ; REMAINING COMMAND LINE FOR MESSAGE COMMAND
2571                                     ;
2572 051414                     N160$:  CLI     CLISPA,0,N161$                   ;SKIP LEADING SPACES
2573 051420                     N161$:  CLI     <'/>,0,N178$                     ;IF CHAR. "/", CONT., ELSE BR N178$
2574 051424                             CLI     CLISTR,0,N170$,<'TYPE'>          ;IS NEXT WORD "TYPE"
2575 051440                             CLI     <'=>,0,N32$                      ;  IF YES, FOLLOWED BY '='?
2576 051444                             CLI     CLISTR,CALPHA,N162$,<'ALPHA'>    ;    IF "ALPHA", SET FLAG
2577 051460                             CLI     CLIBR,0,N168$                    ;    CONTINUE AT N168$
2578 051464                     N162$:  CLI     CLISTR,CONES,N163$,<'ONES'>      ;    IF "ONES", SET FLAG
2579 051500                             CLI     CLIBR,0,N168$                    ;    CONTINUE AT N168$
```

COMMAND LINE ACTION TREE

```
2580 051504                     N163$:  CLI     CLISTR,CZEROS,N164$,<'ZEROS'>   ;    IF "ZEROS", SET FLAG
2581 051520                             CLI     CLIBR,0,N168$                   ;    CONTINUE AT N168$
2582 051524                     N164$:  CLI     CLISTR,C1ALT,N165$,<'1ALT'>     ;    IF "1ALT", SET FLAG
2583 051540                             CLI     CLIBR,0,N168$                   ;    CONTINUE AT N168$
2584 051544                     N165$:  CLI     CLISTR,COALT,N166$,<'OALT'>     ;    IF "OALT", SET FLAG
2585 051560                             CLI     CLIBR,0,N168$                   ;    CONTINUE AT N168$
2586 051564                     N166$:  CLI     CLISTR,CCCITT,N167$,<'CCITT'>   ;    IF "CCITT", SET FLAG
2587 051600                             CLI     CLIBR,0,N168$                   ;    CONTINUE AT N168$
2588 051604                     N167$:  CLI     <'">,CSAVR4,N32$                ;    IF "OPERATOR", SET FLAG
2589 051610                             CLI     CLIBR,COPRSL,N168$              ;    AND INPUT SPECIFIED STRING
2590 051614                     N168$:  CLI     CLIBR,CTYPE,N160$               ;  DO "TYPE", CHECK FOR MORE INPUT
2591 051620                     N170$:  CLI     CLISTR,0,N175$,<'SIZE'>         ;ELSE IS WORD "SIZE"
2592 051634                             CLI     <'=>,0,N32$                     ;  IF YES, FOLLOWED BY "="?
2593 051640                             CLI     CLIDEC,CSIZE,N32$               ;    STORE NUMBER IN M$SIZE
2594 051644                             CLI     CLIBR,0,N160$                   ;  CHECK FOR MORE INFO
2595 051650                     N175$:  CLI     CLISTR,0,N32$,<'COPIES'>        ;ELSE IS WORD "COPIES"
2596 051666                             CLI     <'=>,0,N32$                     ;  IF YES, FOLLOWED BY "="?
2597 051672                             CLI     CLIDEC,CCPYS,N32$               ;    STORE NUMBER IN M$CPYS
2598 051676                             CLI     CLIBR,0,N160$                   ;  CHECK FOR MORE INFO
2599 051702                     N178$:  CLI     CLIBR,0,N32$                    ;ELSE "ILL COMMAND'
2600                                     ;
2601                                     ; SECOND KEYWORD FOR RUN COMMAND
2602                                     ;
2603 051706                     N180$:  CLI     CLISPA,0,N181$                  ;SKIP LEADING SPACES
2604 051712                     N181$:  CLI     CLISTR,CLUPPR,N182$,<'LOOPPAIR'>        ;IS NEXT WORD "LOOPPAIR
2605 051732                             CLI     CLIBR,0,N185$                   ;  IF YES, SET "LOOPPAIR" FLAG
2606 051736                     N182$:  CLI     CLISTR,CRNALL,N183$,<'ALL'>     ;ELSE IS NEXT WORD "ALL"
2607 051750                             CLI     CLIBR,0,N185$                   ;  IF YES, SET "ALL" FLAG
2608 051754                     N183$:  CLI     CLISTR,CDIR,N184$,<'DIRECT'>    ;ELSE IS NEXT WORD "DIRECT"
2609 051772                             CLI     CLIBR,0,N185$                   ;  IF YES, SET "DIRECT" FLAG
2610 051776                     N184$:  CLI     CLISTR,CPATRN,N32$,<'PATTERN'>  ;ELSE IS NEXT WORD "PATTERN'
2611 052014                     N185$:  CLI     CLIBR,CDEFLT,N186$              ;SEE IF DEFAULT OF 1 PASS
2612 052020                     N186$:  CLI     CLISTR,0,N32$,<'/PASS'>         ;PARSE THROUGH SWITCH
2613 052034                             CLI     <'=>,0,N32$                     ;PARSE THROUGH "="
2614 052040                             CLI     CLIDEC,0,N32$                   ;GET PASS COUNT
2615 052044                     N190$:  CLI     CLIEXI,CRUN                     ;RUN TEST AND EXIT
2616                                     ;
2617                                     ; REMAINING COMMAND LINE F.R FUNCTION COMMAND
2618                                     ;
2619 052046                     N200$:  CLI     CLISPA,0,N201$                  ; SKIP SPACES
2620 052052                     N201$:  CLI     CLIOCT,CFUNCT,N32$              ; GET OCTAL NUMBER AND DO FUNCT
2621 052056                             CLI     CLIEXI,0                        ; EXIT
2622                                     ;
2623                                     ; REMAINING COMMAND LINE FOR UNSAVE COMMAND
2624                                     ;
2625 052060                     N210$:  CLI     CLISPA,0,N212$                  ;SKIP SPACES
2626 052064                     N212$:  CLI     <'/>,0,N215$                    ;PARSE THROUGH '/'
2627 052070                             CLI     CLIEXI,CUNSVF                   ;SAVE POINTER TO FILE NAME
2628 052072                     N215$:  CLI     CLIEXI,0
```

```
                                        .SBTTL  GLOBAL TEXT SECTION
                .NLIST  BIN
        ;**
        ; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
        ; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
        ; MORE THAN ONE TEST.
        ;
        ;--
                        ERRTBL
052074                                                          L$ERRTBL::
052074
052074  ERRTYP::        .WORD   0
052076  ERRNBR::        .WORD   0
052100  ERRMSG::        .WORD   0
052102  ERRBLK::        .WORD   0
052104  CLI$PM: .ASCIZ  <12><15>/NIE>/                  ;NIE PROMPT
052113  CLIERM: .ASCIZ  /%N%A?ILL CMD-BAD SYNTAX?/
052144  CLINUF: .ASCIZ  /%N%A?INCOMPLETE COMMAND?/
052175  CLINBG: .ASCIZ  /%N%A?NUMBER TOO BIG?/
052222  CLIBRX: .ASCIZ  /%N%A?BAD RADIX?/
052242  L5060:  .ASCIZ  <CR><LF>\IS YOUR WALL OUTLET POWER 60 HERTZ?\
052310  LDRESP: .ASCIZ  /%N%ANODE %T%A HAS RESPONDED./
052345  RECERR: .ASCIZ  /%N%APACKET RECEIVED WITH QNA ERROR./
052411  RTRYER: .ASCIZ  /%N%ATRANSMISSION ABORTED -- EXCESSIVE COLLISIONS./
052473  BLDMSG: .ASCIZ  /%N%D2%A NODE ADDRESSES ADDED, ELAPSED TIME: %D2%A MINUTE./
052565  ILADMS: .ASCII  /%N%ACANNOT USE BROADCAST ADDRESS (FF-FF-FF-FF-FF-FF)/
052651  ILADM1: .ASCIZ  /%N%AFOR LOOP TESTING. ADDRESS WAS NOT ADDED TO NODE TABLE.%N/
052746  CADRER: .ASCIZ  /%N%APLEASE ENTER TWELVE HEXADECIMAL DIGITS./
053022  CADERR: .ASCIZ  /%N%ATWELVE HEX-DIGITS REQUIRED FOR ADDRESS./
053076  NULSTR: .ASCIZ  /%N%AA ZERO LENGTH STRING WAS ENTERED./
053144  NODADR: .ASCIZ  /%N%S2%T/
053154  LOGNAM: .ASCIZ  /%S14%AN%D2/
053167  NODTYP: .ASCIZ  /%S14%T/
053176  NTBHDR: .ASCIZ  \%N%ANODE PHYSICAL ADDRESS     NODE LOGICAL NAME     TYPE(A/T)\
053274  TABFUL: .ASCIZ  /%N%ATHE %T%A TABLE IS FILLED TO CAPACITY!/
053346  TABEMT: .ASCIZ  /%N%ATHE %T%A TABLE IS CURRENTLY EMPTY!/
053415  NOD:    .ASCIZ  /NODE/
053422  SUMM:   .ASCIZ  /SUMMARY/
053432  CLRMSG: .ASCIZ  /%N%ATHE MESSAGE PARAMETERS HAVE BEEN RESET TO:/
053511  CPYLMT: .ASCIZ  /%N%ATHE NUMBER OF COPIES MUST BE BETWEEN 1 AND 255./
053575  SIZLMT: .ASCIZ  /%N%ATHE MESSAGE SIZE MUST BE BETWEEN 32 AND 512 BYTES./
053664  NOCMPR: .ASCIZ  /%N%ATHE ADDRESS MARKED FOR DELETION WAS NOT IN THE TABLE./
053756  UNBOND: .ASCIZ  /%N%AAN UNBOUNDED "OPERATOR INPUT" STRING WAS ENTERED./
054044  ADRDEL: .ASCIZ  /%N%ATHE ADDRESS HAS BEEN DELETED FROM THE NODE TABLE./
054132  LOGDEL: .ASCIZ  /%N%ANODE N%D1%A HAS BEEN DELETED FROM THE NODE TABLE./
054220  TABCLR: .ASCIZ  /%N%ATHE %T%A TABLE HAS BEEN CLEARED./
054265  UNSMSG: .ASCIZ  /%N%ATHE NODE TABLE HAS BEEN %T/
054324  SAVED:  .ASCIZ  /SAVED./
054333  RESTOR: .ASCIZ  /RESTORED./
054345  MSGPRM: .ASCIZ  /%N%ATHE CURRENT MESSAGE PARAMETERS ARE:/
054415  MSG1:   .ASCIZ  /%N%ATHE COLLECTION OF ALL NODE ADDRESSES COULD TAKE AS LONG AS 40 MINUTES,/
054530  MSG11:  .ASCIZ  /%N%AHOWEVER, IF NO NEW NODES ARE ADDED TO THE TABLE FOR A 10 MINUTE PERIOD/
054643  MSG12:  .ASCIZ  /%N%ATHE COLLECTION WILL STOP.%N/
054703  MSG2:   .ASCIZ  /%N%AYOU ENTERED %T%A NODE: %T/
054741  MSG3:   .ASCIZ  /%N%ATHE SPECIFIED ADDRESS IS: %T/
055002  MSG4:   .ASCIZ  /%N%ATYPE=%T%A,SIZE=%D4%A,COPIES=%D3/
                .EVEN
055046  HDMSG1: .ASCIZ  /%N%A ETHERNET DEFAULT ADDRESS (HEX):  %T/
```

```
2682                    .EVEN
2683 055120  HELP1:    .ASCIZ  \#N#ACOMMAND SUMMARY FOR THE NETWORK INTERCONNECT EXERCISER (NIE)\
2684 055221  HELP2:    .ASCIZ  \#N#A(IT IS ONLY NECESSARY TO TYPE THE LETTERS IN BRACKETS)\
2685 055314  HELP3:    .ASCIZ  \#N2#A[H]ELP OR ?          TYPES THIS HELP TEXT.\
2686 055365  HELP4:    .ASCIZ  \#N2#A[E]XIT               - RETURN TO THE SUPERVISOR.\
2687 055436  HELP5:    .ASCIZ  \#N2#A[SH]OW [N]ODES       - PRINTS INFORMATION IN CURRENT NODE TABLE.\
2688 055536  HELP6:    .ASCIZ  \#N2#A[SH]OW [M]ESSAGE     - PRINTS THE SELECTED MESSAGE TYPE, SIZE AND COPIES.\
2689 055651  HELP7:    .ASCIZ  \#N2#A[SH]OW [C]OUNTERS    - PRINTS THE LOW LEVEL COUNTERS OF THE HOST NODE.\
2690 055762  HELP8:    .ASCIZ  \#N2#A[R]UN [L]OOPPAIR/PASS=NN - RUNS THE LOOPPAIR TEST.\
2691 056052  HELP9:    .ASCIZ  \#N2#A[R]UN [A]LL/PASS=NN - RUNS THE NODE-TO-NODE TEST.\
2692 056141  HELP10:   .ASCIZ  \#N2#A[R]UN [D]IRECT/PASS=NN - RUNS THE LOOP DIRECT TEST.\
2693 056232  HELP11:   .ASCIZ  \#N2#A[R]UN [P]ATTERN/PASS=NN - RUNS THE MESSAGE PATTERN TEST.\
2694 056330  HLP115:   .ASCIZ  \#N2#A[R]UN [R]ESPONDER - RUNS LOOP-SERVER/CONSOLE-ID RESPONDER TEST.\
2695 056435  HELP12:   .ASCIZ  \#N2#A[M]ESSAGE/[T]YPE=A/[S]IZE=N/[C]OPIES=M - ALLOWS THE OPERATOR TO\
2696 056542  HELP13:   .ASCIZ  \#N#AMODIFY THE DEFAULT MESSAGE TYPE, SIZE AND COPY PARAMETERS.\
2697 056641  HELP14:   .ASCIZ  \#N2#A[N]ODE ADR/TYPE - ENTERS A PHYSICAL ADDRESS INTO THE NODE\
2698 056740  HELP15:   .ASCIZ  \#N#ATABLE.  THE TYPE CAN BE EITHER [T]ARGET (DEFAULT) OR [A]SSIST.\
2699 057043  HELP16:   .ASCIZ  \#N2#A[SU]MMARY - PRINTS A SUMMARY OF THE TEST RESULTS.\
2700 057132  HELP17:   .ASCIZ  \#N2#A[B]UILD - BUILDS A TABLE OF REMOTE NODE PHYSICAL ADDRESSES BY\
2701 057235  HELP18:   .ASCIZ  \#N#ALISTENING TO ID MESSAGES ON THE NI.\
2702 057305  HELP19:   .ASCIZ  \#N2#A[C]LEAR [N]ODE/ADR - REMOVES THE NODE SPECIFIED BY EITHER ADR\
2703 057410  HELP20:   .ASCIZ  \#N#AOR NODE LOGICAL NAME FROM THE NODE TABLE.\
2704 057466  HELP21:   .ASCIZ  \#N2#A[C]LEAR [N]ODE/[AL]L - CLEARS THE NODE TABLE.\
2705 057551  HELP22:   .ASCIZ  \#N2#A[C]LEAR [M]ESSAGE - SETS ALL MESSAGE PARAMETERS TO DEFAULT.\
2706 057652  HELP23:   .ASCIZ  \#N2#A[C]LEAR [S]UMMARY - CLEARS THE TABLE OF SUMMARY TEST DATA.\
2707 057752  HELP24:   .ASCIZ  \#N2#A[I]DENTIFY ADR - USES THE REQUEST ID FUNCTION TO IDENTIFY A REMOTE NODE ON THE NI.\
2708 060102  HELP25:   .ASCIZ  \#N2#A[SA]VE - SAVES THE CONTENTS OF THE NODE TABLE.\
2709 060166  HELP26:   .ASCIZ  \#N2#A[U]NSAVE - REPLACES THE CURRENT NODE TABLE WITH THE SAVED ONE.\
2710 060272  HELP27:   .ASCIZ  \#N#S8#ANOTES: 1) ADR IS THE PHYSICAL ADDRESS OF A NODE ON THE NI.\
2711 060374  HELP28:   .ASCIZ  \#N#S8#A         2) PASS COUNT IS A DECIMAL NUMBER BETWEEN 1 AND 65534. A DEFAULT\
2712 060513  HELP29:   .ASCIZ  \#N#S8#A         VALUE OF 1 IS ASSUMED.\
2713 060563  HELP30:   .ASCIZ  \#N#S8#A         SPECIFYING -1 CAUSES THE TEST TO BE RUN INDEFINATELY.\
2714                    .EVEN
2715                    ;
2716                    ;       TEST MESSAGES AND ARGUMENTS
2717                    ;
2718 060672  PASABT:   .ASCIZ  /#N#A PASS ABORTED!/
2719 060715  TSTMS1:   .ASCIZ  /#N#T#A TEST -- /
2720 060735  TSTMS2:   .ASCIZ  /#N#T#A NODE: #T/
2721 060755  TSTMS3:   .ASCIZ  /#T#A ERROR/
2722 060770  TSTMS4:   .ASCIZ  /#N#T#A NODE: #T#T#A NODE: #T/
2723 061025  OK:       .ASCIZ  /#A - RESPONSE OK/
2724 061046  OKRE:     .ASCIZ  /#N#A - RECEIVE ASSIST - RESPONSE OK/
2725 061112  OKTR:     .ASCIZ  /#N#A - TRANSMIT ASSIST - RESPONSE OK/
2726 061157  OKFU:     .ASCIZ  /#N#A - FULL ASSIST   RESPONSE OK/
2727 061220  MESPAT:   .ASCIZ  /#N#AERROR OCCURED WITH #T#A MESSAGE TYPE/
2728 061271  MESPA1:   .ASCIZ  /#A DATA PATTERN: #T/
2729 061315  ALLNOD:   .ASCIZ  /ALL NODE/
2730 061326  LUPAIR:   .ASCIZ  /LOOPPAIR/
2731 061337  DIRECT:   .ASCIZ  /LOOP DIRECT/
2732 061353  FULAST:   .ASCIZ  /FULL ASSIST/
2733 061367  TRAST:    .ASCIZ  /TRANSMIT ASSIST/
2734 061407  RECAST:   .ASCIZ  /RECEIVE ASSIST/
2735 061426  PATTRN:   .ASCIZ  /MESSAGE PATTERN/
2736 061446  NORESP:   .ASCIZ  /NO RESPONSE/
2737 061462  RETRY:    .ASCIZ  /EXCESSIVE COLLISION/
2738 061506  LENGTH:   .ASCIZ  /LENGTH/
```

```
2739 061515  COMPAR: .ASCIZ  /DATA COMPARISON/
2740                .EVEN
2741          ;
2742          ;
2743 061536  MSGTY0: .ASCIZ  /ALPHA/                    ;MESSAGE TYPES
2744 061544  MSGTY1: .ASCIZ  /ONES/
2745 061551  MSGTY2: .ASCIZ  /ZEROS/
2746 061557  MSGTY3: .ASCIZ  /1ALT/
2747 061564  MSGTY4: .ASCIZ  /0ALT/
2748 061571  MSGTY5: .ASCIZ  /CCITT/
2749 061577  MSGTY6: .ASCIZ  /OPER SEL/
2750 061610  CMDTY1: .ASCIZ  /EXIT/                    ;COMMAND TYPES
2751 061615  CMDTY2: .ASCIZ  /SUMMARY/
2752 061625  CMDTY3: .ASCIZ  /BUILD/
2753 061633  CMDTY4: .ASCIZ  /SHOW/
2754 061640  CMDTY5: .ASCIZ  /RUN/
2755 061644  CMDTY6: .ASCIZ  /MESSAGE/
2756 061654  CMDTY7: .ASCIZ  /NODE/
2757 061661  CMDTY8: .ASCIZ  /CLEAR/
2758 061667  CMDTY9: .ASCIZ  /REQUEST ID/
2759 061702  ARGTY1: .ASCIZ  /NODES/                   ;ARGUMENT TYPES
2760 061710  ARGTY2: .ASCIZ  /MESSAGES/
2761 061721  ARGTY3: .ASCIZ  /COUNTERS/
2762 061732  ARGTY4: .ASCIZ  /LOOPPAIR/
2763 061743  ARGTY5: .ASCIZ  /ALL/
2764 061747  ARGTY6: .ASCIZ  / ASSIST/
2765 061757  ARGTY7: .ASCIZ  /TARGET/
2766                .EVEN
2767 061766  NOCLK:  .ASCIZ  /%N%ABAD CLOCK - PROGRAM WILL HANG ON "TIMEOUT"!!/
2768
2769          ;
2770          ;       QNA COUNTER INFORMATION MESSAGES
2771          ;
2772
2773 062047  CNTR00: .ASCIZ  /%N%S5%ACONTENTS OF NODE %T%A INTERNAL COUNTERS:/
2774 062127  CNTR01: .ASCIZ  /%N2%ASECONDS SINCE LAST ZEROED:%S15%Z5/
2775 062176  CNTR02: .ASCIZ  /%N%APACKETS RECEIVED:%S19%T/
2776 062232  CNTR03: .ASCIZ  /%N%AMULTICAST PACKETS RECEIVED:%S9%T/
2777 062277  CNTR04: .ASCIZ  /%N%APACKETS REC'D WITH ERROR - BITMAP:%S8%B4/
2778 062354  CNTR05: .ASCIZ  /%N%APACKETS RECEIVED WITH ERROR:%S13%Z5/
2779 062424  CNTR06: .ASCIZ  /%N%ADATA BYTES RECEIVED:%S16%T/
2780 062463  CNTR07: .ASCIZ  /%N%AMULTICAST DATA BYTES RECEIVED:%S6%T/
2781 062533  CNTR08: .ASCIZ  /%N%ARECEIVED PACKETS LOST-INTERNAL:%S10%Z5/
2782 062606  CNTR09: .ASCIZ  /%N%ARECEIVED PACKETS LOST -LOCAL:%S12%Z5/
2783 062657  CNTR10: .ASCIZ  /%N%APACKETS TRANSMITTED:%S16%T/
2784 062716  CNTR11: .ASCIZ  /%N%AMULTICAST PACKETS TRANSMITTED:%S6%T/
2785 062766  CNTR12: .ASCIZ  /%N%APACKETS TRANSMITTED 3+ TRYS:%S8%T/
2786 063034  CNTR13: .ASCIZ  /%N%APACKETS TRANSMITTED 2 TRYS:%S9%T/
2787 063101  CNTR14: .ASCIZ  /%N%APACKETS DEFFERED:%S19%T/
2788 063135  CNTR15: .ASCIZ  /%N%ADATA BYTES TRANSMITTED:%S13%T/
2789 063177  CNTR16: .ASCIZ  /%N%AMULTICAST BYTES TRANSMITTED:%S8%T/
2790 063245  CNTR17: .ASCIZ  /%N%ATRANSMIT PACKETS ABORTED-BITMAP:%S8%B6/
2791 063320  CNTR18: .ASCIZ  /%N%ATRANSMIT PACKETS ABORTED:%S16%Z5/
2792 063365  CNTR19: .ASCIZ  /%N%AXMIT COLLISION CHECK FAILURE:%S12%Z5/
2793
2794          ;
2795          ;       ERROR MESSAGES FOR DEQNA DRIVER
```

```
2796                    ;
2797
2798 063436  EMSG01: .ASCIZ  /DEQNA PORT COMMAND ERROR/       ; SHOULD NEVER GET THIS ERROR
2799 063467  EMSG02: .ASCIZ  /DEQNA NON EXISTENT MEMORY ERROR/
2800 063527  EMSG03: .ASCIZ  /UNEXPLAINED DEQNA INTERRUPT/
2801 063563  EMSG04: .ASCIZ  /UNKNOWN QNA ERROR/
2802 063605  EMSG08: .ASCIZ  /TIMEOUT!--DEQNA HARDWARE FAILED TO TRANSMIT PACKET/
2803 063670  EMSG10: .ASCIZ  /DEQNA STILL OWNS TRANSMIT DESCRIPTOR/
2804 063735  EMSG11: .ASCIZ  /RECEIVE RING BOOKKEEPING ERROR/
2805 063774  EMSG14: .ASCIZ  /MESSAGE SIZE TOO BIG FOR MAX. PACKET LENGTH/
2806 064050  EMSG22: .ASCIZ  /%N%ANO RESPONSE FROM NODE./
2807 064103  EMSG24: .ASCIZ  /TRANSMIT ERROR, ALL PACKETS NOT TRANSMITTED/
2808 064157  EMSG25: .ASCIZ  /ERROR WHILE WRITING MULTICAST ADDRESS LIST INTO DEQNA/
2809 064245  EMSG32: .ASCIZ  \%N%AILLEGAL TARGET/ASSIST PAIR IN NODE TABLE\
2810 064322  EMSG34: .ASCIZ  <15><12>/TIMEOUT WAITING FOR LOOPBACK REPLY/
2811 064367  EMSG35: .ASCIZ  /%AFAILING NODE ADDRESS: %T%N/
2812 064424  EMSG36: .ASCIZ  /%ADATA PATTERN: %T%N/
2813 064451  EMSG37: .ASCIZ  /%AFAILING TARGET NODE ADDRESS: %T%N/
2814 064515  EMSG38: .ASCIZ  /%AFAILING ASSIST NODE ADDRESS: %T%N/
2815 064561  EMSG40: .ASCIZ  <15><12>/TIMEOUT WAITING FOR REPLY - LOOP MESSAGE TYPE - RECEIVE ASSIST/
2816 064662  EMSG41: .ASCIZ  <15><12>/TIMEOUT WAITING FOR REPLY - LOOP MESSAGE TYPE - TRANSMIT ASSIST/
2817 064764  EMSG42: .ASCIZ  <15><12>/TIMEOUT WAITING FOR REPLY - LOOP MESSAGE TYPE - FULL ASSIST/
2818 065062  EMSG43: .ASCIZ  \NON-EXISTENT MEMORY ERROR DURING ADDRESS SETUP\
2819 065141  EMSG44: .ASCIZ  \SETUP ADDRESS MISMATCH ERROR - DEQNA INTERNAL LOOPBACK FAILED\
2820 065237  EMSG45: .ASCIZ  \ADDRESS SETUP-PACKET TRANSMIT ERROR\
2821 065303  EMSG46: .ASCIZ  \ADDRESS SETUP-PACKET TRANSMITTED, BUT NOT RECEIVED\
2822 065366  EMSG47: .ASCIZ  \PACKET RECEIVED WAS NOT AN ADDRESS SETUP-PACKET\
2823 065446  EMSG48: .ASCIZ  \ADDRESS SETUP-PACKET TRANSMIT AND RECEIVE BYTE COUNT MISMATCH\
2824 065544  EMSG50: .ASCIZ  \CASE OFFSET RANGE UNDERFLOW\
2825 065600  EMSG51: .ASCIZ  \CASE OFFSET RANGE OVERFLOW\
2826 065633  EMSG52: .ASCIZ  \LOOP-SERVER FUNCTION CODE IS NEITHER FORWARD OR REPLY\
2827 065721  EMSG53: .ASCIZ  \RECEIVE DATA TOO LARGE FOR RECEIVE BUFFER\
2828 065773  EMSG54: .ASCIZ  \MESSAGE ADDRESS WAS GOOD, BUT RUNT PACKET WAS RECEIVED.\
2829 066063  EMSG55: .ASCIZ  \RECEIVER OVERFLOW OCCURRED\
2830 066116  EMSG56: .ASCIZ  \RECEIVER CRC ERROR OCCURRED\
2831 066152  EMSG57: .ASCIZ  \RECEIVER FRAMING ERROR OCURRED\
2832 066211  EMSG58: .ASCIZ  \RECEIVED PACKET WITH ERRORS, THROWING IT AWAY\
2833 066267  EMSG59: .ASCIZ  \UNEXPECTED LOOP-SERVER REPLY PACKET RECEIVED\
2834 066344  EMSG60: .ASCIZ  \RECEIVED PACKET OF LESS THAN 10 BYTES\
2835 066412  EMSG61: .ASCIZ  \UNKNOWN RECEIVE ERROR OCCURRED\
2836 066451  EMSG62: .ASCIZ  \ERROR OCCURRED WHILE TRANSMITTING CONSOLE ID\
2837 066526  EMSG63: .ASCIZ  \NEXT RCV DESCRIPTOR POINTER PASSED CURRENT RCV POINTER\
2838 066615  EMSG64: .ASCIZ  \OPERATOR ERROR - CANNOT USE THE DEVICE'S OWN NI ADDRESS\
2839 066705  EMSG65: .ASCIZ  \TIMEOUT WAITING FOR BOOT/DIAGNOSTIC CODE TO LOAD FROM QNA\
2840 066777  EMSG66: .ASCIZ  \LOADED BOOT/DIAGNOSTIC ROM CODE AND ROM CHEKCSUM MISMATCH\
2841 067071  EMSG67: .ASCIZ  \CITIZENSHIP TEST FAILED\
2842 067121  EMSG68: .ASCIZ  \QNA HARDWARE RECEIVED A PACKET NOT ADDRESSED TO IT\
2843 067204  EMSG69: .ASCIZ  \AN ASSIST NODE HAS BEEN CHANGED TO A TARGET NODE\
2844                    ;
2845                    .EVEN
2846 067266  SIMSG1: .ASCIZ  /%N%A  NODE DEFAULT ADDRESS: %T/
2847 067325  SIMSG2: .ASCIZ  /%N%S8%ARECEIPT NUMBER: %06/
2848 067360  SIMSG3: .ASCIZ  /%N%A   MAINTENANCE VERSION: %Z2/
2849 067420  SIMSG4: .ASCIZ  /%N%S19%AECO: %Z2/
2850 067441  SIMSG5: .ASCIZ  /%N%S14%AUSER ECO: %Z2/
2851 067467  SIMSG6: .ASCIZ  /%N%S14%AFUNCTION: %02/
2852 067515  SIMSG7: .ASCIZ  /%N%S16%ADEVICE: %02/
```

```
2853                    ;
2854                    .EVEN
2855 067542  PCMSG::  .ASCIZ  /%N%APC OF CALLING ROUTINE = %O6/
2856                    .EVEN
2857 067602  CMPER1:  .ASCIZ  /%N%ABYTE NUMBER:%D4%A(D) EXPECTED=%O6%A(O) RECEIVED=%O6%A(O)/
2858 067677  CMPER2:  .ASCIZ  /%N%ATOTAL MISMATCHES IN MESSAGE = %D4/
2859 067745  LGERMS:  .ASCIZ  /%N%ALENGTH ERROR -- BYTES EXPECTED: %O6%A BYTES RECEIVED: %O6/
2860 070043  SUMMS1:  .ASCIZ  /%N2%S8%ANODE: %T/
2861 070064  SUMMS2:  .ASCIZ  /%N%ARX NOT COMPLETE    RX COMPLETE    LENGTH ERRORS/
2862 070150  SUMMS3:  .ASCIZ  /%N%S6%Z5%S12%Z5%S10%Z5/
2863 070177  SUMMS4:  .ASCIZ  /%N%ACOMPARE ERRORS    BYTES COMPARED    BYTES XFER/
2864 070262  SUMMS5:  .ASCIZ  /%N%S6%Z5%S8%T/
2865 070300  SUMMS6:  .ASCIZ  /%S5%T/
2866                    .EVEN
2867
2868                                    .LIST   BIN
```

```
                                .SBTTL  GLOBAL ERROR REPORT SECTION

                                ;++
                                ; THE GLOBAL ERROR REPORT SECTION CONTAINS MESSAGE PRINTING AREAS
                                ; USED BY MORE THAN TEST TO OUTPUT ADDITIONAL ERROR INFORMATION.  PRINTB
                                ; (BASIC) AND PRINTX (EXTENDED) CALLS ARE USED TO CALL PRINT SERVICES.
                                ;--


                                BGNMSG  ERR1
070306                                                                  ERR1::
070306
2880 070306  012737  000020  003734        MOV     #CEXIT,CFLAG
2881 070314                               PRINTX  #PCMSG,PCCALL
     070314  013746  050330                                                     MOV     PCCALL,-(SP)
     070320  012746  067542                                                     MOV     #PCMSG, (SP)
     070324  012746  000002                                                     MOV     #2,-(SP)
     070330  010600                                                             MOV     SP,R0
     070332  104415                                                             TRAP    C$PNTX
     070334  062706  000006                                                     ADD     #6,SP
2882 070340                               DOCLN
     070340  104444                                                             TRAP    C$DCLN
2883 070342                               ENDMSG
     070342                                                             L10002:
     070342  104423                                                             TRAP    C$MSG
2884 070344                               BGNMSG  ERR2
     070344                                                             ERR2::
2885 070344  010146                              MOV     R1,-(SP)
2886 070346  013701  002372                      MOV     P$TYPE,R1
2887 070352  006301                            ASL     R1
2888 070354  062701  003324                      ADD     #MSGTAB,R1
2889 070360                               PRINTX  #EMSG35,#STRBUF
     070360  012746  002324                                                     MOV     #STRBUF, (SP)
     070364  012746  064367                                                     MOV     #EMSG35,-(SP)
     070370  012746  000002                                                     MOV     #2, (SP)
     070374  010600                                                             MOV     SP,R0
     070376  104415                                                             TRAP    C$PNTx
     070400  062706  000006                                                     ADD     #6,SP
2890 070404                               PRINTX  #EMSG36,(R1)
     070404  011146                                                             MOV     (R1), (SP)
     070406  012746  064424                                                     MOV     #EMSG36, (SP)
     070412  012746  000002                                                     MOV     #2,-(SP)
     070416  010600                                                             MOV     SP,R0
     070420  104415                                                             TRAP    C$PNTx
     070422  062706  000006                                                     ADD     #6,SP
2891 070426  012601                              MOV     (SP)+,R1
2892 070430                               ENDMSG
     070430                                                             L10003:
     070430  104423                                                             TRAP    C$MSG
2893
2894 070432                               BGNMSG  ERR3
     070432                                                             ERR3::
2895 070432                               PRINTX  #EMSG37,#STRBUF
     070432  012746  002324                                                     MOV     #STRBUF, (SP)
     070436  012746  064451                                                     MOV     #EMSG37, (SP)
     070442  012746  000002                                                     MOV     #2, (SP)
     070446  010600                                                             MOV     SP,R0
     070450  104415                                                             TRAP    C$PNTx
```

```
GLOBAL ERROR REPORT SECTION

                070452  062706  000006                                                  ADD     #6,SP
          2896  070456                        PRINTX  #EMSG38,#STRBU1
                070456  012746  002346                                                  MOV     #STRBU1,-(SP)
                070462  012746  064515                                                  MOV     #EMSG38,-(SP)
                070466  012746  000002                                                  MOV     #2,-(SP)
                070472  010600                                                          MOV     SP,R0
                070474  104415                                                          TRAP    C$PNTX
                070476  062706  000006                                                  ADD     #6,SP
          2897  070502                  ENDMSG
                070502                                                          L10004:
                070502  104423                                                          TRAP    C$MSG
          2899                          ;*****************************************************************************
          2900                          ;       THESE MESSAGE AREAS ARE USED TO OUTPUT SUPPLEMENTARY INFORMATION
          2901                          ;       AFTER AN ERROR CALL.  THEY ARE INVOKED BY APPENDING THE NAME
          2902                          ;       OF THE AREA TO AN ERROR CALL:  ERRXXX 1,ERRORMESSAGE,AREANAME.
          2903                          ;       THE CORRESPONDING MESSAGE AREA IS SET UP IN THIS SECTION:
          2904                          ;               BGNMSG  AREANAME
          2905                          ;               [CODE]
          2906                          ;               ENDMSG
          2907                          ;
          2908                          ;       THE AREAS IN THIS SECTION ARE FOR MESSAGES USED IN MORE THAN ONE
          2909                          ;       TEST.  USE THE PRINTB (PRINT BASIC) AND PRINTX (PRINT EXTENDED)
          2910                          ;       MACROS.
          2911                          ;*****************************************************************************
```

```
2914                                    .SBTTL  GLOBAL SUBROUTINES SECTION
2915                                ;++
2916                                ; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
2917                                ; THAT ARE USED IN MORE THAN ONE TEST.
2918                                ;
2919                                ;++
2920                                ; FUNCTIONAL DESCRIPTION:
2921                                ;     SUBROUTINE TO....
2922                                ;
2924                                ;*************************************************************************
2925                                ;       COMPLETE THE "SUBROUTINE TO...." STATEMENT WITH A FUNCTIONAL
2926                                ;       DESCRIPTION OF THIS SUBROUTINE.
2927                                ;*************************************************************************
2929                                ;
2930                                ; INPUTS:
2931                                ;
2933                                ;*************************************************************************
2934                                ;       LIST THE INPUT DATA THAT ARE EXPLICITLY PASSED TO THIS SUBROUTINE.
2935                                ;*************************************************************************
2937                                ;
2938                                ; IMPLICIT INPUTS:
2939                                ;
2941                                ;*************************************************************************
2942                                ;       LIST THE INPUT DATA THAT ARE IMPLICITLY USED BY THIS SUBROUTINE;
2943                                ;       FOR EXAMPLE, DATA READ FROM COMMON AREAS.
2944                                ;*************************************************************************
2946                                ;
2947                                ; OUTPUTS:
2948                                ;
2950                                ;*************************************************************************
2951                                ;       LIST THE OUTPUT DATA THAT ARE EXPLICITLY GIVEN BY THIS SUBROUTINE
2952                                ;*************************************************************************
2954                                ;
2955                                ; IMPLICIT OUTPUTS:
2956                                ;
2958                                ;*************************************************************************
2959                                ;       LIST THE OUTPUT DATA THAT ARE IMPLICITLY GIVEN BY THIS SUBROUTINE;
2960                                ;       FOR EXAMPLE, DATA STORED IN COMMON AREAS.
2961                                ;*************************************************************************
2963                                ;
2964                                ; SUBORDINATE ROUTINES USED:
2965                                ;
2967                                ;*************************************************************************
2968                                ;       LIST THE SUBROUTINES CALLED BY THIS SUBROUTINE.
2969                                ;*************************************************************************
2971                                ;
2972                                ; FUNCTIONAL SIDE EFFECTS:
2973                                ;
2975                                ;*************************************************************************
2976                                ;       DESCRIBE ANY EFFECTS THIS SUBROUTINE MAY HAVE UPON OTHER
2977                                ;       MODULES OF THE DIAGNOSTIC PROGRAM.  AN EXAMPLE OF THIS IS
2978                                ;       THE SUBROUTINE INHIBITS ALL INTERRUPTS WITH PRIORITY 7.
2979                                ;*************************************************************************
2981                                ;
2982                                ; CALLING SEQUENCE:
2983                                ;
2985                                ;*************************************************************************
```

```
2986                    ;       GIVE THE EXACT CALLING SEQUENCE USED TO ACCESS THIS SUBROUTINE.
2987                    ;       FOR EXAMPLE:    MOV COUNT,R1    ;MOVE INPUT TO R1
2988                    ;                       JSR     PC,ROUTINE      ;GO TO ROUTINE
2989                    ;                       BCS     ERROR           ;CARRY SET IF ROUTINE HAD ERROR
2990                    ;*****************************************************************************
2992                    ;
2993                    ;
2995                    ;*****************************************************************************
2996                    ;       INSERT THE CODE FOR THIS SUBROUTINE.   THE NAME OF THE SUBROUTINE SHOULD
2997                    ;       BE DEFINED WITH A DOUBLE-COLON (::);  THIS WILL MAKE THE SUBROUTINE GLOBAL.
2998                    ;*****************************************************************************
3000                    ;
3002                    ;*****************************************************************************
3003                    ;       BEGIN EACH SUBROUTINE AT THE TOP OF A NEW PAGE.
3004                    ;*****************************************************************************
3006                    ;
```

```
                                .SBTTL          CLKSET  CLOCK SETUP SUBROUTINE

                        ;+
                        ; FUNCTIONAL DESCRIPTION:
                        ;               THIS SUBROUTINE SETS UP THE CLOCK INFORMATION TABLE FOLLOWING
                        ;               A "CLOCK" CALL EXECUTED IN THE INITIALIZATION CODE.  BUT SINCE
                        ;               THE 'CLOCK" CALL SAYS NOTHING ABOUT AN LSI 11'S CLOCK, THE
                        ;               ROUTINE IS ONLY USED IF A LINE OR P CLOCK IS FOUND.
                        ;
                        ; INPUTS        R1 - POINTS TO SUPERVISOR SPACE WHERE CLOCK INFO WAS RETURNED
                        ;               R2 - POINTS TO "CLK" TABLE WHERE CLOCK INFO WILL BE KEPT
                        ;
                        ; OUTPUTS -     "CLKCSR" GETS LOADED WITH THE CLOCK'D CSR ADDRESS
                        ;               "CLKBR" GETS LOADED WITH THE CLOCK'S INTERRUPT LEVEL
                        ;               "CLKVEC" GETS LOADED WITH THE CLOCK'S INTERRUPT VECTOR
                        ;               "CLKHZ" GETS LOADED WITH THE LINE FREQ. (IN HERTZ)
                        ;
                        ; CALLING PROCEDURE:
                        ;               JSR     PC,CLKSET       ; CALL CLOCK SETUP WITH R1 AND R2 SETUP
                        ;
                        ;-
070504                  CLKSET:
070504  012122                 MOV     (R1)+,(R2)+             ; LOAD CLOCK'S CSR ADDR. INTO "CLKCSR"
070506  012112                 MOV     (R1)+,(R2)              ; LOAD CLOCK'S INTR. LEVEL INTO "CLKBR"
070510  006312                 ASL     (R2)                    ; ADJUST THE INTR. LEVEL FOR LOADING
070512  006312                 ASL     (R2)                    ;   INTO THE PSW WITH A "SETVEC" CALL
070514  006312                 ASL     (R2)
070516  006312                 ASL     (R2)
070520  006322                 ASL     (R2)+
070522  012122                 MOV     (R1)+,(R2)+             ; LOAD CLOCK'S INTR. VEC INTO "CLKVEC"
070524  012122                 MOV     (R1)+,(R2)+             ; LOAD CLOCK'S FREQ. INTO "CLKHZ'
070526  000207                 RTS     PC
```

```
3041                                    .SBTTL          CLKINT  CLOCK INTERRUPT SERVICE ROUTINE
3042
3043                                ;--+
3044                                ; FUNCTIONAL DESCRIPTION:
3045                                ;               THIS IS THE CLOCK INTERRUPT SERVICE ROUTINE WHICH TAKES CARE
3046                                ;               OF KEEPING THE "TIME-SINCE-START" AND COUNTING DOWN ANY OF THE
3047                                ;               "EVENT" TIMERS.  THE TIMERS ARE USED TO TIME COMPLETION OF
3048                                ;               DEVICE REQUESTS.  THE "TIME-SINCE-START" IS USED TO BE LOGGED
3049                                ;               WITH EACH ENTRY INTO THE EVENT LOG.
3050                                ;
3051                                ; IMPLICIT INPUTS - TIMTCK - THE CURRENT NO. OF TICKS LEFT TO BE COUNTED UNTIL
3052                                ;                     A SECOND HAS BEEN COUNTED OFF
3053                                ;                   CLKHZ - THE NO. OF TICKS IN A SECOND, DETERMINED BY THE
3054                                ;                     SYS. LINE FREQ.
3055                                ;                   TIMMIN & TIMSEC - CURRENT VALUE OF "TIME-SINCE START" IN
3056                                ;                     MINUTES AND SECONDS
3057                                ;                   TIMER 1,2 AND S - CURRENT VALUES OF "EVENT TIMERS"
3058                                ;
3059                                ; IMPLICIT OUTPUTS - NEW VALUE OF EVENT TIMER "1" & "2" DECREMENTED BY 1 TICK
3060                                ;                     IF IT WAS NON-ZERO
3061                                ;                    NEW VALUE OF EVENT TIMER "S" DECREMENTED BY 1 SECOND IF IT
3062                                ;                     WAS NON-ZERO
3063                                ;
3064                                ; SIDE EFFECTS -          THE CLOCK IS DISABLED UPON ENTRY AND REENABLED WHEN LEAVING
3065                                ;
3066                                ; CALLING PROCEDURE - THIS ROUTINE IS CALLED WHEN THE CLODK INTERRUPTS THRU
3067                                ;                     "CLKVEC".  THE ADDRESS OF THIS ROUTINE WAS LOADED
3068                                ;                     INTO THE CLOCK'S INTERRUPT VECTOR WITH A "SETVEC" CALL
3069                                ;
3070                                ;--+
3071                                ;
3072 070530                             BGNSRV  CLKINT
     070530                                                                  CLKINT::
3073                                ;
3074 070530  005077  113202             CLR     @CLKCSR                 ; DISABLE THE CLOCK FROM INTERRUPTING
3075 070534  005337  003754             DEC     TIMTCK                  ; DECREMENT THE NO. OF TICKS/SEC
3076 070540  001017                     BNE     10$                     ; GO CHECK TIMERS
3077 070542  013737  003744  003754     MOV     CLKHZ,TIMTCK            ; RESET THE NO. OF TICKS/SEC.
3078 070550  005237  050146             INC     CNTRS+C.SECS            ; INCREMENT QNA'S NUMBER OF SECS
3079 070554  005237  003752             INC     TIMSEC                  ; INC. NO OF SECS-SINCE-START
3080 070560  022737  000074  003752     CMP     #60.,TIMSEC             ; SEE IF WE'VE COUNTED 60 SEC.S YET
3081 070566  001004                     BNE     10$                     ; IF NOT, GO CHECK TIMERS
3082 070570  005237  003750             INC     TIMMIN                  ;  ELSE, INC. MINUTES-SINCE-START
3083 070574  005037  003752             CLR     TIMSEC                  ;  AND RESTART SECOND COUNTER
3084 070600                         10$:
3085 070600  005737  003756             TST     TIMER1                  ; SEE IF TIMER1 TIMING ANYTHING
3086 070604  001402                     BEQ     20$                     ;  IF=0, NO, CHECK NEXT TIMER
3087 070606  005337  003756             DEC     TIMER1                  ;  ELSE DECREMENT THE TIMER VALUE (BY 1 TICK)
3088 070612                         20$:
3089 070612  005737  003760             TST     TIMER2                  ; SEE IF TIMER2 TIMING ANYTHING
3090 070616  001402                     BEQ     30$                     ;  IF=0, NO, CHECK NEXT TIMER
3091 070620  005337  003760             DEC     TIMER2                  ;  ELSE DECREMENT TIMER VALUE (BY 1 TICK)
3092 070624                         30$:
3093 070624  005737  003762             TST     TIMERS                  ; SEE IF TIMERS TIMING ANYTHING
3094 070630  001406                     BEQ     40$                     ;  IF=0, NOTHING BE TIMED, LEAVE
3095 070632  023737  003744  003754     CMP     CLKHZ,TIMTCK            ; SEE IF A SECOND HAS BEEN COUNTED OFF
3096 070640  001002                     BNE     40$                     ;  BR IF NO
```

```
3097 070642  005337  003762                DEC     TIMERS                  ;  ELSE, DECREMENT TIMER VALUE (BY 1 SEC.)
3098 070646                          40$:
3099 070646  013777  003746  113062        MOV     CLKEN,@CLKCSR           ; REENABLE THE CLOCK TO INTERRUPT
3100 070654                                ENDSRV
     070654                                                          L10005:
     070654  000002                                                        RTI
```

```
3102                                            .SBTTL  PREG14  PRESERVE REGISTERS 1 THROUGH 4 ACROSS SUBROUTINE CALLS
3103                                    ;+
3104                                    ;INPUTS:        THE RELATIVE ADDRESS OF THE CALLED ROUTINE MUST FOLLOW THE
3105                                    ;               CALL TO THIS ROUTINE (SEE CALLING SEQUENCE).
3106                                    ;
3107                                    ;OUTPUTS:       REGISTERS 1 THROUGH 4 ARE PRESERVED ACROSS THE CALLED ROUTINE.
3108                                    ;               A CHECK IS MADE TO ENSURE THE HARDWARE STACK HASN T OVER RUN
3109                                    ;               THE PARAMETER STACK.
3110                                    ;
3111                                    ;CALLING SEQUENCE:      THIS IS BEST HANDLED BY THE "CALL' MACRO.  THE ACTUAL
3112                                    ;                       CALL IS:
3113                                    ;                                       JSR     R4,PREG14
3114                                    ;                                       .WORD   [SUBROUTINE NAME] ANCHOR
3115                                    ;
3116                                    ;COMMENTS:      THIS ROUTINE IS RE-ENTRANT AND RELOCATABLE.
3117                                    ;               THIS ROUTINE IS DRS COMPATIBLE.
3118                                    ;
3119                                    ;SUBORDINATE ROUTINES CALLED:   THE ROUTINE SPECIFIED IN THE CALL.
3120                                    ;-
3121                                                                    ;R4 IS ALREADY ON THE R6 STACK.
3122 070656                            PREG14:
3123 070656  010346                            MOV     R3,-(SP)        ;PUSH R3, R2, R1
3124 070660  010246                            MOV     R2,-(SP)        ;
3125 070662  010146                            MOV     R1,-(SP)        ;
3126 070664  010437  050330                    MOV     R4, PCCALL
3127 070670  012401                            MOV     (R4)+,R1        ;GET THE RELATIVE ADDRESS OF THE CALLED
3128                                                                    ;ROUTINE.
3129 070672  060701                            ADD     PC,R1           ;MAKE IT AN ABSOLUTE ADDRESS.
3130 070674                            ANCHOR:
3131 070674  010446                            MOV     R4,-(SP)        ;SAVE THE RETURN TO THE CALLING ROUTINE.
3132 070676  020506                            CMP     R5,SP           ;CHECK FOR STACK OVER-RUN.
3133 070700  103401                            BLO     10$             ;
3134 070702  000000                            HALT                    ;HANDLE STACK OVER RUN CONDITION.
3135 070704                            10$:
3136 070704  004711                            JSR     PC,(R1)         ;CALL THE SPECIFIED ROUTINE.
3137 070706  012604                            MOV     (SP)+,R4        ;RESTORE THE RETURN TO THE CALLING ROUTINE.
3138 070710  012601                            MOV     (SP)+,R1        ;RESTORE THE REGISTERS.
3139 070712  012602                            MOV     (SP)+,R2        ;
3140 070714  012603                            MOV     (SP)+,R3        ;
3141 070716  000204                            RTS     R4              ;BACK TO THE CALLING ROUTINE.
```

```
                                .SBTTL          WAIT    WAIT FOR DEQNA INTERRUPT WITH TIMEOUT

                              ;++
                              ; FUNCTIONAL DESCRIPTION:
                              ;               THIS SUBROUTINE WAITS FOR INTERRUPTS FROM THE DEQNA
                              ;               OR REPORTS A TIMEOUT.   IF A QNA INTERRUPT HAS NOT
                              ;               OCCURED WITHIN THE TIMEOUT PERIOD THE SUBROUTINE ERROR
                              ;               IS CALLED TO HANDLE IT.
                              ;
                              ;               SUCCESS OR FAILURE IS REPORTED VIA P1.
                              ;
                              ; INPUTS
                              ;               NONE
                              ;
                              ; OUTPUTS-
                              ;               P1:     SUCCESS/FAILURE      0=SUCCESS/ 1=FAILURE
                              ;
                              ; CALLING SEQUENCE:
                              ;               CALL    WAIT
                              ;               P$POP   P1
                              ;--
070720                        WAIT::
070720  012703  000012                MOV     #10.,R3                 ; MOVE NO. OF COUNTS TO R3
070724  012704  003756                MOV     #TIMER1,R4              ;  AND TIMER TO BE USED TO R4
070730  005002                        CLR     R2                      ;LOCAL STATUS PARAMETER
070732  010314                        MOV     R3,(R4)                 ;SET NUMBER OF TICKS. (GLOBAL)
070734                        10$:
070734  005737  050270                TST     ERRFLG                  ;CHECK IF ERROR OCCURED
070740  001011                        BNE     30$                     ; BR IF YES
070742  005737  050262                TST     DNIFLG                  ;CHECK FOR INTERRUPT
070746  001403                        BEQ     20$                     ; BR IF INTERRUPT RECEIVED
070750  005037  050262                CLR     DNIFLG
070754  000410                        BR      50$
070756                        20$:
070756  005714                        TST     (R4)                    ;HAS TIMER EXPIRED?
070760  001365                        BNE     10$                     ; BR IF NO TO WAIT FOR INTERRUPT
070762  000403                        BR      40$                     ;BR TO 40$
070764                        30$:
070764                                CALL    ERROR                   ;CALL ERROR ROUTINE
070764  004437  070656                                                        JSR     R4,PREG14
070770  000106                                                                .WORD   ERROR ANCHOR
070772                        40$:
070772  012702  177777                MOV     # 1,R2                  ;INDICATE FAILURE
070776                        50$:
070776                                RETURN  R2                      ;RETURN WITH SUCCESS/FAILURE INDICATION
071000  010225                                                                MOV     R2,(R5)+
071000  000207                                                                RTS     PC
```

```
                        .SBTTL  ERROR   HANDLE QNA INTERRUPT ERRORS

                ;+
                ; FUNCTIONAL DESCRIPTION:
                ;               THIS SUBROUTINE CHECKS THE ERROR FLAGS SET BY
                ;               QNAISR THE INTERRUPT SERVICE ROUTINE AND PRINTS
                ;               OUT THE APPRORIATE ERROR MESSSAGES.
                ;
                ; INPUTS
                ;   IMPLICIT:   ERROR FLAGS SHOULD BE SET BY UNAISR ROUTINES.
                ; OUTPUTS
                ;   IMPLICIT:   ERROR MESSAGES ARE PRINTED OUT TO THE OPERATOR CONSOLE.
                ;
                ; CALLING SEQUENCE:
                ;               CALL ERROR
                ;
                ;-
071002          ERROR::
071002  005337  050270          DEC     ERRFLG                  ;DECREMENT ERROR COUNTER TO SHOW
                                                                ;THAT IT HAS BEEN HANDLED
071006  005737  050254          TST     PCEFLG                  ;SEE IF PORT COMMAND ERROR
071012  001013                  BNE     10$                     ; IF YES, BRANCH
071014  005737  050252          TST     FATFLG                  ;SEE IF QNA FATAL ERROR
071020  001017                  BNE     20$                     ; IF YES, BRANCH
071022  005737  050266          TST     BCOUNT                  ;SEE IF UNEXPLAINED INTERRUPT
071026  001023                  BNE     30$                     ; IF YES, BRANCH
071030                          ERRDF   4,EMSG04,ERR1           ;ELSE UNKNOWN ERROR
                                                                        TRAP    C$ERDF
071030  104455                                                    .WORD   4
071032  000004                                                    .WORD   EMSG04
071034  063563                                                    .WORD   ERR1
071036  070306
071040  000424                  BR      40$                     ; EXIT
071042                  10$:    ERRDF   1,EMSG01,ERR1           ;PORT COMMAND ERROR
071042  104455                                                        TRAP    C$ERDF
071044  000001                                                    .WORD   1
071046  063436                                                    .WORD   EMSG01
071050  070306                                                    .WORD   ERR1
071052  005337  050254          DEC     PCEFLG                  ; INDICATE THAT IT WAS HANDLED
071056  000415                  BR      40$                     ; EXIT
071060                  20$:    ERRDF   2,EMSG02,ERR1           ;QNA FATAL ERROR
071060  104455                                                        TRAP    C$ERDF
071062  000002                                                    .WORD   2
071064  063467                                                    .WORD   EMSG02
071066  070306                                                    .WORD   ERR1
071070  005337  050252          DEC     FATFLG                  ; KEEP UP ON BOOK KEEPING
071074  000406                  BR      40$                     ; EXIT
071076                  30$:    ERRDF   3,EMSG03,ERR1           ;UNEXPLAINED INTERRUPT
071076  104455                                                        TRAP    C$ERDF
071100  000003                                                    .WORD   3
071102  063527                                                    .WORD   EMSG03
071104  070306                                                    .WORD   ERR1
071106  005337  050266          DEC     BCOUNT                  ; BOOK KEEPING
071112                  40$:    RETURN                          ;RETURN
071112  000207                                                        RTS     PC

```

```
                        .SBTTL          QNAINI  INITIALIZE THE QNA
                ;+
                ; SUBROUTINE TO
                ;               1) SETS QNA IN THE READY STATE
                ;               2) INITIALIZES ALL QNA GLOBAL DATA LOCATIONS
                ;                       TO DEFAULT VALUES.
                ; ROUTINE STEPS:
                ;
                ;       SET RESET BIT RESET THE QNA
                ;       SET TIMER TO ALLOW 100 USECONDS FOR INIT
                ;       INVALIDATE THE TRANSMIT LIST(S)
                ;       VALIDATE THE RECEIVE LIST(S)
                ;       ZERO RECEIVED BUFFER COUNT
                ;       WAIT FOR TIMER
                ;       CLEAR THE RESET BIT
                ;       SET VECTOR ADDRESS
                ;       READ PHYSICAL ADDRESS ROM AND STORE IT
                ;       WRITE THE RECEIVE RING LOWER AND UPPER ADDRESS(S)
                ;       ENABLE INTERRUPTS AND DATA RECEPTION
                ;       EXIT
                ;
                ; CALLED BY:
                ;               CALL QNAINI
                ;
                ; INPUTS:       NONE
                ;
                ; OUTPUTS:      NONE
                ;
                ; SIDEEFFECTS:  ALL GLOBAL LOCATIONS ARE ZEROED
                ;-
                ;
                ; RESET AND STOP QNA HARDWARE
                ;
071114          QNAINI::
071114  013702  047772          MOV     QNAADO,R2               ; GET THE DEVICE ADDRESS
071120  052737  000002  050246  BIS     #MRESET,CSRBUF          ; SET SOFTWARE INIT BIT
071126  013762  050246  000016  MOV     CSRBUF,CSR(R2)          ; WRITE IT TO THE QNA
071134  012737  000001  003756  MOV     #1,TIMER1               ; TIMER FOR INITITIALIZE
071142          10$:
071142  005737  003756          TST     TIMER1                  ; INIT DONE?
071146  001375                  BNE     10$                     ; BRANCH IF NOT
                ;
071150  012762  000002  000016  MOV     #MRESET,CSR(R2)         ; CLEAR THE RESET BIT
071156  005062  000016          CLR     CSR(R2)                 ; WRITE IT TO THE QNA
                ;
                ;LOAD AND RUN CITIZENSHIP TEST
                ;
071162  005737  050250          TST     FLAG1                   ; HAVE WE GONE THRU THE CITIZENSHIP
071166  001162                  BNE     QNAIN1                  ; TEST BEFORE?  IF YES, BRANCH.
071170  012737  000001  050250  MOV     #1,FLAG1
071176  012701  000002          MOV     #2,R1                   ;BUILD TWO RECEIVE DESCRIPTORS OF
                                                                ;  2K BYTES EACH
071202  012704  021200          MOV     #RDESC,R4               ;GET POINTER TO DESCRIPTOR
071206  012703  004100          MOV     #RBUFB,R3               ;GET POINTER TO 1ST BUFFER
071212          40$:
071212  012724  177777          MOV     #-1,(R4)+               ;FLAGWORD
071216  012724  100000          MOV     #V,(R4)+                ;DESCRIPTOR BITS
```

QNAINT INITIALIZE THE QNA

```
3283 071222  010324                        MOV     R3,(R4)+                ;POINTER TO BUFFER
3284
3285 071224  062703  004000                ADD     #2048.,R3               ;GET POINTER TO 2ND BUFFER
3286 071230  012724  176000                MOV     #-1024.,(R4)+           ;SIZE IN WORDS
3287 071234  012724  177777                MOV     # 1,(R4)+              ;STATUS WORD 1
3288 071240  012724  177777                MOV     #-1,(R4)+              ;STATUS WORD 2
3289
3290 071244  005301                        DEC     R1
3291 071246  003361                        BGT     40$
3292 071250  005024                        CLR     (R4)+                   ;FLAGWORD OF INVALID DESCRIPTOR
3293 071252  005024                        CLR     (R4)+                   ;DESCRIPTOR BITS OF INVALID DESCRIPTOR
3294
3295 071254  012762  000002  000016        MOV     #MRESET,CSR(R2)         ; CLEAR THE RESET BIT
3296 071262  005062  000016                CLR     CSR(R2)                 ; WRITE IT TO THE QNA
3297
3298 071266  012762  021200  000004        MOV     #RDESC,LORCV(R2)        ;SET POINTER TO DESCRIPTOR
3299 071274  012762  001010  000016        MOV     #BD!EL,CSR(R2)          ;SET LOAD ROM BIT
3300 071302  005062  000006                CLR     HIRCV(R2)               ;BEGIN THE LOAD
3301 071306  005004                        CLR     R4                      ;WAIT FOR BOOT PHASE 1
3302 071310                          45$:
3303 071310  077401                        SOB     R4,45$
3304 071312  042762  000010  000016        BIC     #BD,CSR(R2)             ;CLEAR BOOT BIT
3305 071320  005004                        CLR     R4                      ;WAIT FOR BOOT PHASE 2
3306 071322                          47$:
3307 071322  077401                        SOB     R4,47$
3308
3309 071324  012762  000002  000016        MOV     #MRESET,CSR(R2)         ; CLEAR THE RESET BIT
3310 071332  005062  000016                CLR     CSR(R2)                 ; WRITE IT TO THE QNA
3311
3312 071336  012704  021200                MOV     #RDESC,R4
3313 071342  026427  000000  177777        CMP     FLAGWORD(R4),# 1        ;DID CITIZENSHIP TEST LOAD O.K.?
3314 071350  001016                        BNE     50$                     ;NO, ERROR
3315
3316 071352  026427  000010  177777        CMP     STAT3(R4),# 1
3317 071360  001012                        BNE     50$                     ;NO, ERROR
3318
3319 071362  026427  000012  177777        CMP     STAT4(R4),#-1
3320 071370  001006                        BNE     50$                     ;NO, ERROR
3321
3322 071372  012704  021214                MOV     #RDESC+RDESSZ,R4        ;GET POINTER TO 2ND DESCRIPTOR
3323 071376  026427  000000  177777        CMP     FLAGWORD(R4),# 1        ;DID CITIZENSHIP TEST LOAD O.K.?
3324 071404  001410                        BEQ     70$                     ;YES
3325
3326 071406                          50$:
3327 071406                              ERRHRD  65,EMSG65               ;NO, ERROR.
     071406  104456                                                              TRAP    C$ERHRD
     071410  000101                                                              .WORD   65
     071412  066705                                                              .WORD   EMSG65
     071414  000000                                                              .WORD   0
3328 071416  012700  177777                MOV     # 1,R0                  ;SET FAILURE STATUS
3329 071422  000137  072122                JMP     QNAEXI                  ;AND EXIT
3330
3331 071426                          70$:
3332 071426  013701  047772                MOV     QNAADO,R1               ;GET BASE ADDRESS
3333
3334 071432  010246                        MOV     R2,-(SP)
3335 071434  010546                        MOV     R5, (SP)                ;SAVE R5 STACKPOINTER
```

```
3336
3337 071436  012702  014100                  MOV     #CITWORK,R2             ;GET WORK AREA
3338 071442  004737  004104                  JSR     PC,RBUFB+CITVECTOR      ;GO TO THE CITIZENSHIP TEST VIA VECTOR
3339
3340 071446  012605                          MOV     (SP)+,R5                ;RESTORE R5 STACKPOINTER
3341 071450  012602                          MOV     (SP)+,R2
3342
3343 071452  005700                          TST     R0                      ;DID ERRORS OCCUR DURING CITIZENSHIP?
3344 071454  001413                          BEQ     80$                     ;NO
3345 071456  020027  100000                  CMP     R0,#100000
3346 071462  001410                          BEQ     80$                     ;NO
3347 071464                                  ERRHRD  67,EMSG67               ;NO, ERROR.
     071464  104456                                                                  TRAP    C$ERHRD
     071466  000103                                                                  .WORD   67
     071470  067071                                                                  .WORD   EMSG67
     071472  000000                                                                  .WORD   0
3348 071474  012700  177777                  MOV     #-1,R0                  ;SET FAILURE STATUS
3349 071500  000137  072122                  JMP     QNAEXI                  ;AND EXIT
3350
3351 071504  032762  010000  000016  80$:    BIT     #XC,CSR(R2)             ;IS TRANSCEIVER OK?
3352 071512  001410                          BEQ     QNAIN1                  ;YES
3353 071514                                  ERRHRD  66,EMSG66               ;NO, ERROR.
     071514  104456                                                                  TRAP    C$ERHRD
     071516  000102                                                                  .WORD   66
     071520  066777                                                                  .WORD   EMSG66
     071522  000000                                                                  .WORD   0
3354 071524  012700  177777                  MOV     #-1,R0                  ;SET FAILURE STATUS
3355 071530  000137  072122                  JMP     QNAEXI                  ;AND EXIT
3356
3357 071534                          QNAIN1:
3358 071534  013702  047772                  MOV     QNAADO,R2               ; RESTORE DEVICE ADDRESS TO R2
3359 071540  052737  000002  050246          BIS     #MRESET,CSRBUF          ; SET SOFTWARE INIT BIT
3360 071546  013762  050246  000016          MOV     CSRBUF,CSR(R2)          ; WRITE IT TO THE QNA
3361 071554  012737  000001  003756          MOV     #1,TIMER1               ; TIMER FOR INITITIALIZE
3362 071562                          5$:
3363 071562  005737  003756                  TST     TIMER1                  ; INIT DONE?
3364 071566  001375                          BNE     5$                      ; BRANCH IF NOT
3365 071570  042737  000002  050246          BIC     #MRESET,CSRBUF          ; CLEAR THE RESET BIT
3366 071576  013762  050246  000016          MOV     CSRBUF,CSR(R2)          ; WRITE IT TO THE QNA
3367 071604  013762  047774  000014          MOV     QNAVCO,VECTOR(R2)       ; SET THE INTERRUPT VECTOR
3368 071612  012737  024236  024220          MOV     #XRING,XRGCUR           ; SET XMIT POINTER TO BEGINNING OF RING
3369 071620  012737  024236  024224          MOV     #XRING,XRGNXT           ; AGAIN
3370 071626  012737  024260  024222          MOV     #RRING,RRGCUR           ; SET RCV POINTER TO BEGINNING OF RING
3371 071634  012737  024544  024234          MOV     #RRINGH,RRGPRV          ; SET PREVIOUS POINTER TO LAST DESCRIPTOR
3372 071642  012737  024260  024226          MOV     #RRING,RRGNXT           ; SET NEXT DESCRIPTOR POINTER
3373 071650  012703  000001                  MOV     #1,R3                   ; RESET OWNERSHIP/STATUS OF XMIT RING
3374 071654                          10$:
3375 071654  013701  024220                  MOV     XRGCUR,R1               ; RING ENTRIES
3376 071660  005061  000000                  CLR     FLAG(R1)                ; RESET FLAG
3377 071664  005061  000010                  CLR     STAT1(R1)               ; RESET STATUS
3378 071670  005061  000012                  CLR     STAT2(R1)               ; RESET STATUS 2
3379 071674  012761  024574  000004          MOV     #XRG001,LOADD(R1)
3380 071702                                  CALL    GETXNX  #0,#XRGCUR
     071702  012725  024220                                                        MOV     #XRGCUR,(R5)+
     071706  012725  000000                                                        MOV     #0,(R5)+
     071712  004437  070656                                                        JSR     R4,PREG14
     071716  006534                                                                .WORD   GETXNX ANCHOR
```

```
3381 071720  005303                        DEC     R3
3382 071722  001354                        BNE     10$
3383 071724  012703  000017                MOV     #15.,R3                 ; ...AND RECEIVE RING
3384 071730                          20$:
3385 071730  013701  024222                MOV     RRGCUR,R1
3386 071734  005061  000000                CLR     FLAG(R1)                ; CLEAR FLAG WORD
3387 071740  005061  000010                CLR     STAT1(R1)               ; CLEAR STATUS 1
3388 071744  005061  000012                CLR     STAT2(R1)               ; CLEAR STATUS 2
3389 071750  012761  100000  000002        MOV     #MVALID,DESC(R1)        ; SET THE VALID BIT FOR THIS DESCRIPTOR
3390 071756                                CALL    GETRNX  #0,#RRGCUR      ; UPDATE, BUT DON'T VALDIATE PREVIOUS
     071756  012725  024222                                                      MOV     #RRGCUR,(R5)+
     071762  012725  000000                                                      MOV     #0,(R5)+
     071766  004437  070656                                                      JSR     R4,PREG14
     071772  006450                                                              .WORD   GETRNX ANCHOR
3391 071774  005303                        DEC     R3
3392 071776  001354                        BNE     20$
3393 072000  012737  024236  024220        MOV     #XRING,XRGCUR           ; SET POINTERS TO BEGINING OF RING
3394 072006  012737  024260  024222        MOV     #RRING,RRGCUR
3395 072014  005037  024546                CLR     RRINGH+DESC             ; INVALIDATE LAST DESCRIPTOR
3396 072020  005037  050256                CLR     NIRCNT                  ; CLEAR BUFFERS RECEIVED COUNTER
3397 072024  012762  024260  000004        MOV     #RRING,LORCV(R2)        ; SET LOW RECEIVE LIST ADDRESS
3398 072032  005062  000006                CLR     HIRCV(R2)               ; SET HIGH ADDRESS OF RCV LIST
3399 072036  052737  000501  050246        BIS     #MILOOP!MINTEN!MRCVEN,CSRBUF    ; SET BITS IN BUFFERED CSR
3400 072044  013762  050246  000016        MOV     CSRBUF,CSR(R2)          ; WRITE TO THE QNA
3401 072052  012703  004010                MOV     #PHYADR,R3              ; GET PHYSICAL NODE ADDRESS BUF
3402                                ;
3403                                ; NOW SETUP ADDRESSES IN THE DEQNA
3404                                ;
3405 072056  116223  000000                MOVB    NETADD(R2),(R3)+        ; GET BYTE OF NODE ADDRESS
3406 072062  116223  000002                MOVB    NETADD+2(R2),(R3)+      ; GET NEXT BYTE OF NODE ADDRESS
3407 072066  116223  000004                MOVB    NETADD+4(R2),(R3)+      ; GET NEXT BYTE OF NODE ADDRESS
3408 072072  116223  000006                MOVB    NETADD+6(R2),(R3)+      ; GET NEXT BYTE OF NODE ADDRESS
3409 072076  116223  000010                MOVB    NETADD+10(R2),(R3)+     ; GET NEXT BYTE OF NODE ADDRESS
3410 072102  116213  000012                MOVB    NETADD+12(R2),(R3)      ; GET NEXT BYTE OF NODE ADDRESS
3411 072106                                CALL    SETUP   #INIADR         ; CALL SETUP WITH INIT ADRESS FUNC CODE
     072106  012725  000000                                                      MOV     #INIADR,(R5)+
     072112  004437  070656                                                      JSR     R4,PREG14
     072116  001232                                                              .WORD   SETUP-ANCHOR
3412 072120                                P$POP   R0                      ; GET RETURN STATUS
     072120  014500                                                              MOV     (R5),R0
3413 072122                          QNAEXI:
3414 072122                                RETURN  R0                      ; FINISHED, LEAVE, VAMOOSE, ETC.
     072122  010025                                                              MOV     R0,(R5)+
     072124  000207                                                              RTS     PC
```

```
                        .SBTTL  SETUP - SETUP UP NI ADDRESSES IN QNA
                ;**
                ; FUNCTIONAL DESCRIPTION:
                ;
                ;       THIS ROUTINE LETS THE QNA KNOW WHICH ADDRESSES IT IS SUPPOSED TO
                ;       RECOGNIZE.
                ;
                ; BEGIN
                ; POINT TO TARGET ADDRESS TABLE
                ; GET SETUP TYPE FROM PARAMETER STACK
                ; CASE SETUP TYPE
                ; .     INITIALIZE TABLE
                ; .     .       INITIALIZE ADDRESS GROUP POINTER
                ; .     .       DO WHILE ALL ADDRESS GROUPS HAVE NOT BEEN WRITTEN
                ; .     .       .       INITIALIZE COLUMN POINTER
                ; .     .       .       POINT TO BASIC ADDRESSES TABLE (MBZ, OUR OWN)
                ; .     .       .       DO WHILE ALL ADDRESS COLUMNS FOR THIS GROUP ARE DONE
                ; .     .       .       .       IF    BASIC ADDRESSES HAVE BEEN WRITTEN
                ; .     .       .       .       .       THEN
                ; .     .       .       .       .       POINT TO OUR OWN NI PHYSICAL ADDRESS
                ; .     .       .       .       ENDIF
                ; .     .       .       .       EXECUTE SUBROUTINE TO FILL IN THIS ADDRESS
                ; .     .       .       ENDDO
                ; .     .       ENDDO
                ; .     ADD DECNET CONSOLE ID MULTICAST ADDRESS
                ; .     .       POINT TO MULTICAST ADDRESS
                ; .     .       INIT GROUP POINTER
                ; .     .       INIT COLUMN POINTER TO MULTICAST ADDRESS COLUMN
                ; .     .       INIT ROW POINTER
                ; .     .       EXECUTE ROUTINE TO FILL IN THIS ADDRESS
                ; .     KILL DECNET CONSOLE ID MULTICAST ADDRESS
                ; .     .       POINT TO MULTICAST ADDRESS
                ; .     .       INIT GROUP POINTER
                ; .     .       INIT COLUMN POINTER TO MULTICAST ADDRESS COLUMN
                ; .     .       INIT ROW POINTER
                ; .     .       EXECUTE ROUTINE TO FILL IN THIS ADDRESS
                ; ENDCASE
                ; VALIDATE THE XMIT DESCRIPTOR FOR A SETUP
                ; EXECUTE ROUTINE TO XMIT THE SETUP PACKET
                ; IF    STATUS = SUCCESS
                ; .     THEN
                ; .     SET STATUS = ERROR
                ; .     EXECUTE ROUTINE TO FIND A RECEIVED PACKET
                ; .     IF    ERROR WAS DETECTED
                ; .     .       THEN
                ; .     .       .       REPORT RECEIVE ERROR
                ; .     ELSE
                ; .     .       IF    NO PACKET HAS BEEN FOUND
                ; .     .       .       THEN
                ; .     .       .       REPORT PACKET NOT RECEIVED
                ; .     .       ELSE
                ; .     .       .       IF    PACKET IS NOT A SETUP PACKET
                ; .     .       .       .       THEN
                ; .     .       .       .       REPORT UNEXPECTED PACKET
                ; .     .       .       ELSE
                ; .     .       .       .       IF    THE XMITTED PACKET SIZE NOT = RECEIVED PACKET SIZE
                ; .     .       .       .       .       THEN
```

```
3473                                    ;  .    .    .    .    .    REPORT BYTE COUNT ERROR
3474                                    ;  .    .    .    .  ELSE
3475                                    ;  .    .    .    .    .    IF   XMITTED SETUP DATA NOT = RECEIVED SETUP DATA
3476                                    ;  .    .    .    .    .    .    THEN
3477                                    ;  .    .    .    .    .    .    REPORT ADDRESS MISMATCH
3478                                    ;  .    .    .    .    .    ELSE
3479                                    ;  .    .    .    .    .    .    STATUS = SUCCESS
3480                                    ;  .    .    .    .    .    ENDIF
3481                                    ;  .    .    .    .  ENDIF
3482                                    ;  .    .    .  ENDIF
3483                                    ;  .    .  ENDIF
3484                                    ;  .  ENDIF
3485                                    ; ENDIF
3486                                    ; RETURN STATUS = STATUS
3487                                    ; RETURN WITH RETURN STATUS
3488                                    ; END
3489                                    ;
3490                                    ; CALLING SEQUENCE
3491                                    ;
3492                                    ;       TBD
3493                                    ;
3494                                    ; INPUT
3495                                    ;
3496                                    ;       NONE
3497                                    ;
3498                                    ; IMPLICIT INPUT
3499                                    ;
3500                                    ;       NONE
3501                                    ;
3502                                    ; OUTPUT
3503                                    ;
3504                                    ;       NONE
3505                                    ;
3506                                    ; IMPLICIT OUTPUT
3507                                    ;
3508                                    ;       NONE
3509                                    ;
3510                                    ; COMPLETION CODES
3511                                    ;
3512                                    ;       NONE
3513                                    ;
3514                                    ; SIDE EFFECTS
3515                                    ;
3516                                    ;       NONE
3517                                    ;
3518                                    ; REGISTERS USED
3519                                    ;
3520                                    ;       R0 - RETURN STATUS, SET BY CALLED ROUTINES
3521                                    ;       R1 - CASE OFFSET REG, ADDRESS GROUP POINTER
3522                                    ;       R2   ADDRESS COLUMN POINTER
3523                                    ;       R3 - POINTS TO TABLE OF NI ADDRESSES
3524                                    ;
3525                                    ; DEBUG
3526                                    ;
3527                                    ;       NONE
3528                                    ;
3529 072126                            SETUP:
```

```
SETUP   SETUP UP NI ADDRESSES IN QNA

3530 072126  012700  177777                MOV     #-1,R0          ; ASSUME ERROR
3531 072132  012704  023570                MOV     #TGTADR,R4      ; DEQNA ADDRESS TABLE (NOT NODE TABLE)
3532 072136                                P$POP   R1              ; GET PARAMETER, BUT DON'T LOSE IT
     072136  014501                                                                MOV     -(R5),R1
3533 072140  100005                        BPL     10$             ; BRANCH IF NOT UNDER RANGE
3534 072142                                ERRSF   50,EMSG50       ; REPORT RANGE UNDERFLOW
     072142  104454                                                                TRAP    C$ERSF
     072144  000062                                                                .WORD   50
     072146  065544                                                                .WORD   EMSG50
     072150  000000                                                                .WORD   0
3535 072152  000536                        BR      SETEXI
3536 072154                          10$:
3537 072154  022701  000002                CMP     #2,R1           ; CHECK UPPER LIMIT OF CASE OFFSET
3538 072160  002005                        BGE     20$             ; BRANCH IF WITHIN RANG
3539 072162                                ERRSF   51,EMSG51       ; REPORT RANGE OVERFLOW
     072162  104454                                                                TRAP    C$ERSF
     072164  000063                                                                .WORD   51
     072166  065600                                                                .WORD   EMSG51
     072170  000000                                                                .WORD   0
3540 072172  000526                        BR      SETEXI          ; LEAVE
3541 072174                          20$:
3542 072174  006301                        ASL     R1              ; TURN INTO WORD INDEX
3543 072176  062701  072204                ADD     #30$,R1         ; POINT TO WORD OFFSET
3544 072202  061107                        ADD     (R1),PC         ; GO DO IT
3545 072204                          30$:
3546 072204  000006                        .WORD   SETINI - 30$    ; OFFSET TO INITIALIZE
3547 072206  000142                        .WORD   SETADD - 30$    ; OFFSET TO SET DECNET MULTICAST
3548 072210  000204                        .WORD   SETKLD - 30$    ; OFFSET TO KILL DECNET MULTICAST
3549 072212                          SETINI:
3550 072212  012701  177700                MOV     #-100,R1        ; GROUP POINTER
3551 072216                          10$:
3552 072216  062701  000100                ADD     #100,R1         ; BUMP THE GROUP POINTER
3553 072222  001427                        BEQ     20$             ; IF ZERO, DO THE FIRST GROUP
3554 072224  022701  000200                CMP     #200,R1         ; HAVE THE TWO GROUPS BEEN WRITTEN?
3555 072230  001024                        BNE     20$             ; BRANCH IF NOT
3556                                       ;
3557 072232  013701  024220                MOV     XRGCUR,R1       ; GET XMIT PACKET ADDRESS
3558 072236  012761  177700  000006        MOV     #-100,WRDCNT(R1) ; WRITE ALL ADDRESSES
3559 072244                                CALL    SETWRT          ; WRITE THEM
     072244  004437  070656                                                        JSR     R4,PREG14
     072250  001560                                                                .WORD   SETWRT ANCHOR
3560 072252                                P$POP   R0              ; ANY ERROR
     072252  014500                                                                MOV     (R5),R0
3561 072254  001075                        BNE     SETEXI          ; IF YES, GO WRITE THE ADDRESSES
3562 072256  013701  024220                MOV     XRGCUR,R1       ; ELSE, GET NXT XMIT DESCRIPTOR
3563 072262  012761  177705  000006        MOV     #-73,WRDCNT(R1) ; WORD COUNT TO INIT THE MODE BITS
3564 072270                                CALL    SETWRT          ; WRITE THE PACKET
     072270  004437  070656                                                        JSR     R4,PREG14
     072274  001560                                                                .WORD   SETWRT ANCHOR
3565 072276                                P$POP   R0              ; GET STATUS
     072276  014500                                                                MOV     (R5),R0
3566 072300  000463                        BR      SETEXIT         ; LEAVE
3567 072302                          20$:
3568 072302  012702  177777                MOV     #-1,R2          ; INIT THE COLUMN POINTER
3569 072306  012703  004002                MOV     #NOADR,R3       ; POINT TO MBZ AND PHYSICAL ADDRESSES
3570 072312                          30$:
3571 072312  005202                        INC     R2              ; BUMP THE COLUMN POINTER
```

J6

SETUP SETUP UP NI ADDRESSES IN QNA

```
3572 072314  022702  000010                CMP     #8.,R2          ; ADDRESS COLUMNS 0   7 INITIALIZED?
3573 072320  001736                        BEQ     10$             ; DO NEXT GROUP IF YES
3574 072322  005701                        TST     R1              ; IS THIS THE FIRST GROUP:
3575 072324  001003                        BNE     40$             ; BRANCH IF NOT
3576 072326  022702  000002                CMP     #2,R2           ; HAVE BASIC ADDRESSES BEEN WRITTEN?
3577 072332  103002                        BHIS    50$             ; BRANCH IF NOT
3578 072334                        40$:
3579 072334  012703  004010                MOV     #PHYADR,R3      ; E SE, POINT TO OUR PHYSICAL ADDRESS
3580 072340                        50$:
3581 072340  004137  073006                JSR     R1,SETFIL       ; FILL IN THE ADDRESS
3582 072344  000762                        BR      30$             ; GO DO NEXT COLUMN
3583 072346                        SETADD:
3584 072346  012703  004032                MOV     #MCSTAD,R3      ; GET DECNET MULTICAST ADDRESS
3585 072352  005001                        CLR     R1              ; INIT GROUP POINTER
3586 072354  012702  000004                MOV     #4,R2           ; INIT COLUMN TO CORRECT SLOT
3587 072360  004137  073006                JSR     R1,SETFIL       ; FILL IN THIS ADDRESS
3588 072364  013701  024220                MOV     XRGCUR,R1       ; GET CURRENT XMIT DESCRIPTOR
3589 072370  012761  177706  000006        MOV     # 72,WRDCNT(R1) ; WRITE ADDRESSES ONLY
3590 072376                                CALL    SETWRT          ; WRITE THE NEW ADDRESS TABLE INTO QNA
     072376  004437  070656                                              JSR     R4,PREG14
     072402  001560                                                      .WORD   SETWRT-ANCHOR
3591 072404                                P$POP   R0              ; GET THE RETURNED STATUS
     072404  014500                                                      MOV     -(R5),R0
3592 072406  000420                        BR      SETEXI          ; LEAVE
3593 072410                        SETKLD:
3594 072410  012703  004010                MOV     #PHYADR,R3      ; GET OUR PHYSICAL ADDRESS
3595 072414  005001                        CLR     R1              ; INIT GROUP POINTER
3596 072416  012702  000004                MOV     #4,R2           ; INIT ADDRESS COLUMN POINTER
3597 072422  004137  073006                JSR     R1,SETFIL       ; FILL IN THIS ADDRESS
3598 072426  013701  024220                MOV     XRGCUR,R1       ; GET CURRENT XMIT DESCRIPTOR
3599 072432  012761  177706  000006        MOV     #-72,WRDCNT(R1) ; WRITE ADDRESSES ONLY
3600 072440                                CALL    SETWRT          ; WRITE THE NEW ADDRESS TABLE
     072440  004437  070656                                              JSR     R4,PREG14
     072444  001560                                                      .WORD   SETWRT-ANCHOR
3601 072446                                P$POP   R0              ; GET THE RETURNED STATUS
     072446  014500                                                      MOV     (R5),R0
3602 072450                        SETEXI:
3603 072450                                RETURN  R0              ; RETURN
     072450  010025                                                      MOV     R0,(R5)+
     072452  000207                                                      RTS     PC
```

```
                                .SBTTL  SETWRT   WRITE A SETUP PACKET
                        ;**
                        ; FUNCTIONAL DESCRIPTION:
                        ;
                        ;       THIS ROUTINE TAKES CARE OF WRITING SETUP PACKETS AND VERIFYING THAT
                        ;       ALL OF IT OCCURRED OKAY
                        ;
                        ; CALLING SEQUENCE
                        ;
                        ;       CALL    SETWRT
                        ;
                        ; INPUT
                        ;
                        ;       NONE
                        ;
                        ; IMPLICIT INPUT
                        ;
                        ;       NONE
                        ;
                        ; OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; IMPLICIT OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; COMPLETION CODES
                        ;
                        ;       NONE
                        ;
                        ; SIDE EFFECTS
                        ;
                        ;       NONE
                        ;
                        ; REGISTERS USED
                        ;
                        ;       R1 - HOLDS TRANSMITTED ADDRESS TABLE ADDRESS
                        ;       R2 - HOLDS ADDRESS OF PACKET
                        ;       R3 - AND ADDRESS OF RECEIVED ADDRESS TABLE
                        ;       R4   COUNT OF NUMBER OF BYTES TO CHECK
                        ;
                        ; DEBUG
                        ;
                        ;       NONE
                        ;-
072454                  SETWRT:                                 ; WRITE NEW ADDRESS TABLE INTO THE QNA
072454  005200                  INC     R0                      ; ASSUME ERROR
072456  013704  024224          MOV     XRGNXT,R4               ; GET DESCRIPTOR
072462  012764  023570  000004  MOV     @TGTADR,LOADD(R4)
072470                          CALL    XMIT    @XMTSET         ; TRANSMIT THE PACKET
072470  012725  000000                                                  MOV     @XMTSET,(R5)+
072474  004437  070656                                                  JSR     R4,PREG;4
072500  002326                                                          .WORD   XMIT ANCHOR
072502  005737  050252          TST     FATFLG                  ; NON EXISTENT MEMORY?
072506  001407                  BEQ     20$                     ; BRANCH IF NOT
072510                  10$:
```

```
3659 072510  005037  050252                CLR     FATFLG                  ; CLEAR FATAL ERROR FLAG
3660 072514                                ERRHRD  43,EMSG43               ; REPORT TIMEOUT
     072514  104456                                                            TRAP    C$ERHRD
     072516  000053                                                            .WORD   43
     072520  065062                                                            .WORD   EMSG43
     072522  000000                                                            .WORD   0
3661 072524  000504                        BR      WRTEXI                  ; LEAVE
3662 072526                          20$:
3663 072526                                P$POP   R1                      ; GET STATUS
     072526  014501                                                              MOV     (R5),R1
3664 072530  001405                        BEQ     30$                     ; CONTINUE IF OKAY
3665 072532                                ERRHRD  45,EMSG45               ; REPORT XMIT ERROR
     072532  104456                                                            TRAP    C$ERHRD
     072534  000055                                                            .WORD   45
     072536  065237                                                            .WORD   EMSG45
     072540  000000                                                            .WORD   0
3666 072542  000475                        BR      WRTEXI                  ; EXIT SETUP, FAILURE OCCURRED
3667 072544                          30$:
3668 072544                                CALL    RECEVE                  ; PACKET FOUND?
     072544  004437  070656                                                        JSR     R4,PREG14
     072550  003044                                                            .WORD   RECEVE ANCHOR
3669 072552  005737  050252                TST     FATFLG                  ; NONE EXISTENT MEMORY ERROR?
3670 072556  001354                        BNE     10$                     ; BRANCH IF YES
3671 072560  005700                        TST     R0                      ; DID AN ERROR OCCUR?
3672 072562  001401                        BEQ     40$                     ; BRANCH IF NOT
3673 072564  000455                        BR      SETVAL                  ; EXIT SETUP, ERROR ALREADY REPORTED
3674 072566                          40$:
3675 072566                                P$POP   R2                      ; WERE ANY PACKETS RECEIVED
     072566  014502                                                              MOV     (R5),R2
3676 072570  001005                        BNE     50$                     ; BRANCH IF YES
3677 072572                                ERRHRD  46,EMSG46               ; NO PACKET RECEIVED
     072572  104456                                                            TRAP    C$ERHRD
     072574  000056                                                            .WORD   46
     072576  065303                                                            .WORD   EMSG46
     072600  000000                                                            .WORD   0
3678 072602  000455                        BR      WRTEXI
3679 072604                          50$:
3680 072604  013702  024226                MOV     RRGNXT,R2               ; GET ADDRESS OF THE PACKET
3681 072610  032762  020000  000010        BIT     #MRSETP,STAT1(R2)       ; SETUP PACKET?
3682 072616  001005                        BNE     60$                     ; BRANCH IF IT IS
3683 072620                                ERRHRD  47,EMSG47               ; REPORT UNEXPECTED PACKET STATUS
     072620  104456                                                            TRAP    C$ERHRD
     072622  000057                                                            .WORD   47
     072624  065366                                                            .WORD   EMSG47
     072626  000000                                                            .WORD   0
3684 072630  000433                        BR      SETVAL                  ; EXIT
3685 072632                          60$:
3686 072632  016401  000004                MOV     LOADD(R4),R1            ; GET XMITTED ADDRESS TABLE
3687 072636  016404  000006                MOV     WRDCNT(R4),R4           ; GET XMIT SIZE
3688 072642  005404                        NEG     R4                      ; CONVERT FROM TWOS COMPLEMENT
3689 072644  006304                        ASL     R4                      ; TURN WORD COUNT TO BYTE COUNT
3690                                  ;; N.M. ADD     #12,R4                  ; COMPENSATE FOR GARBAGE BYTES
3691 072646  016203  000004                MOV     LOADD(R2),R3            ; GET RECEIVED PACKET
3692 072652  120462  000012                CMPB    R4,STAT2(R2)            ; IS THE SIZE THE SAME?
3693 072656  001405                        BEQ     70$                     ; BRANCH IF IT IS
3694 072660                                ERRHRD  48,EMSG48               ; BYTE COUNT ERROR
     072660  104456                                                            TRAP    C$ERHRD
```

```
                                                                                .WORD   48
                                                                                .WORD   EMSG48
                                                                                .WORD   0
072662  000060
072664  065446
072666  000000
3695  072670  000413                BR      SETVAL                  ; EXIT
3696  072672                    70$:
3697                            ;; N.M. SUB     #12,R4                  ; REMOVE GARBAGE BYTE COUNT
3698  072672  006204                    ASR     R4                      ; TURN BYTE COUNT INTO WORD COUNT
3699  072674                    80$:
3700  072674  022123                    CMP     (R1)+,(R3)+             ; DOES XMITTED = RECEIVED?
3701  072676  001405                    BEQ     90$                     ; BRANCH IF NOT
3702  072700                            ERRHRD  44,EMSG44               ; COMPARE ERROR
                                                                                TRAP    C$ERHRD
072700  104456                                                            .WORD   44
072702  000054                                                            .WORD   EMSG44
072704  065141                                                            .WORD   0
072706  000000
3703  072710  000403                    BR      SETVAL                  ; LEAVE
3704  072712                    90$:
3705  072712  005304                    DEC     R4                      ; DECREMENT THE BYTE COUNT
3706  072714  001367                    BNE     80$                     ; CHECK NEXT ADDRESS
3707  072716  005000                    CLR     R0                      ; SET SUCCESS CODE IN R0
3708  072720                    SETVAL:                                 ; REVALIDATE THE DESCRIPTOR
3709  072720                            CALL    GETRNX  #1,@RRGNXT      ; UPDATE NEXT AND PREVIOUS PACKET ADDRESS
                                                                                MOV     @RRGNXT,(R5)+
072720  012725  024226                                                        MOV     #1,(R5)+
072724  012725  000001                                                        JSR     R4,PREG14
072730  004437  070656                                                        .WORD   GETRNX-ANCHOR
072734  006450
3710  072736                    WRTEXI:
3711  072736  013704  024224            MOV     XRGNXT,R4               ; GET TRANSMIT PACKET ADDRESS AGAIN
3712  072742  012764  024574  000004    MOV     #XRG001,LOADD(R4)       ; RESET ADDRESS
3713  072750  005064  000000            CLR     FLAG(R4)                ; REINIT THE DESCRIPTOR
3714  072754  005064  000010            CLR     STAT1(R4)
3715  072760  005064  000012            CLR     STAT2(R4)
3716  072764                            CALL    GETXNX  #0,@XRGNXT      ; GET NEXT XMIT BUFFER
                                                                                MOV     @XRGNXT,(R5)+
072764  012725  024224                                                        MOV     #0,(R5)+
072770  012725  000000                                                        JSR     R4,PREG14
072774  004437  070656                                                        .WORD   GETXNX ANCHOR
073000  006534
3717  073002                    RETURN  R0                              ; RETURN WITH STATUS
                                                                                MOV     R0,(R5)+
073002  010025                                                                RTS     PC
073004  000207
```

```
                                .SBTTL  SETFIL - FILL AN ADDRESS SLOT IN SETUP TABLE
                        ;++
                        ; FUNCTIONAL DESCRIPTION:
                        ;
                        ;       THIS
                        ;
                        ; BEGIN
                        ; INIT THE ROW POINTER
                        ; DO UNTIL 6 ADDRESS BYTES OF THE ADDRESS ARE WRITTEN
                        ; .     ADD ROW NUMBER TO INDEX
                        ; .     ADD COLUMN NUMBER TO INDEX
                        ; .     ADD TABLE ADDRESS TO INDEX
                        ; .     STUFF THE ADDRESS BYTE IN
                        ; ENDDO
                        ; BUMP ROW POINTER
                        ; STUFF A ZERO INTO THIS BYTE (MBZ)
                        ; BUMP ROW POINTER
                        ; STUFF A ZERO INTO THIS BYTE (MBZ)
                        ; RETURN FROM SUB
                        ; END
                        ;
                        ; CALLING SEQUENCE
                        ;
                        ;       JSR     PC,SETFIL
                        ;
                        ; INPUT
                        ;
                        ;       R1 = GROUP POINTER
                        ;       R2 = COLUMN POINTER
                        ;       R3 = POINTER TO NI ADDRESS
                        ;       R4 = ROW POINTER
                        ;
                        ; IMPLICIT INPUT
                        ;
                        ;       NONE
                        ;
                        ; OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; IMPLICIT OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; COMPLETION CODES
                        ;
                        ;       NONE
                        ;
                        ; SIDE EFFECTS
                        ;
                        ;       NONE
                        ;
                        ; REGISTERS USED
                        ;
                        ;       R1    INDEX
                        ;       R2    ADDRESS COLUMN POINTER
                        ;       R3    POINTS TO GROUPS OF ADDRESSES
```

```
SETFIL   FILL AN ADDRESS SLOT IN SETUP TABLE

                                        ;       R4    ADDRESS ROW POINTER
                                        ;
                                        ; DEBUG
                                        ;
                                        ;       NONE
                                        ;
                                        ; INIT THE ROW POINTER
                                        ; DO WHILE 6 ADDRESS BYTES OF THE ADDRESS ARE TO BE WRITTEN
073006                                  SETFIL:
073006  012704  023570                          MOV     @TGTADR,R4      ; SET ROW POINTER TO DESTINATION TABLE
073012  060204                          ADD     R2,R4           ; ADD COLUMN TO IT
073014  061604                          ADD     (SP),R4         ; ADD SAVED GROUP TO IT
073016                                  PUSH    R1              ; SAVE RETURN ADDRESS
073016  010146                                                          MOV     R1,-(SP)
073020  010401                          MOV     R4,R1           ; INIT LIMIT TO CURRENT ADDRESS
073022  062701  000050                  ADD     @40.,R1         ; UPPER LIMIT (5 ADRS BYTES * 8 BYTE OFFSET TO
                                                                ; NEXT ROW)
073026                                  10$:
073026  112314                          MOVB    (R3)+,(R4)      ; STUFF THE ADDRESS BYTE IN
073030  062704  000010                  ADD     @8.,R4          ; ELSE, POINT TO NEXT ROW IN THE COLUMN
073034  020401                          CMP     R4,R1           ; HAVE ALL ADDRESSES BEEN WRITTEN
073036  003001                          BGT     20$             ; BRANCH IF YES
073040  000772                          BR      10$             ; AND STUFF IT IN
073042                                  20$:
073042  111314                          MOVB    (R3),(R4)       ; STASH LAST ADDRESS BYTE
073044  112714  000000                  MOVB    @0,(R4)         ; NEXT BYTE MBZ
073050  062704  000010                  ADD     @8.,R4          ; POINT TO NEXT BYTE
073054  112714  000000                  MOVB    @0,(R4)         ; NEXT BYTE MBZ
073060                                  POP     R1              ; RESTORE RETURN ADDRESS
073060  012601                                                          MOV     (SP)+,R1
073062  000201                          RTS     R1
```

```
                                .SBTTL  QNA INTERRUPT SERVICE ROUTINE
                        ;++
                        ; FUNCTIONAL DESCRIPTION:
                        ;
                        ;       THIS IS THE INTERRUPT SERVICE ROUTINE
                        ;
                        ; CALLING SEQUENCE
                        ;
                        ;       NONE
                        ;
                        ; INPUT
                        ;
                        ;       NONE
                        ;
                        ; IMPLICIT INPUT
                        ;
                        ;       NONE
                        ;
                        ; OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; IMPLICIT OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; COMPLETION CODES
                        ;
                        ;       NONE
                        ;
                        ; SIDE EFFECTS
                        ;
                        ;       NONE
                        ;
                        ; REGISTERS USED
                        ;
                        ;       NONE
                        ;
                        ; DEBUG
                        ;
                        ;       NONE
                        ;--
073064                  QNAISR:
073064  010146                  MOV     R1,-(SP)                ;SAVE R1
073066  010346                  MOV     R3,-(SP)                ;...
073070  013701  047772          MOV     QNAADO,R1               ; GET QNA ADDRESS
073074  016103  000016          MOV     CSR(R1),R3              ; AND ITS CSR CONTENTS
073100  032703  000004          BIT     #MNXM,R3                ; INTERRUPT RESULT OF MEMORY TIMEOUT?
073104  001403                  BEQ     10$
073106  005237  050252          INC     FATFLG                  ; SET FLAG
073112  000436                  BR      40$                     ; CONTINUE
073114                  10$:
073114  032703  100000          BIT     #MRCVIN,R3              ; RECV INTERRUPT ??
073120  001423                  BEQ     20$                     ; NO
073122  005237  050256          INC     NIRCNT                  ; YES, SET FLAG
073126  052737  100000  050246  BIS     #MRCVIN,CSRBUF          ; SET RECEIVER DONE BIT IN BUFFERED CSR
073134  013761  050246  000016  MOV     CSRBUF,CSR(R1)          ; WRITE IT TO REARM INTERRUPTS
```

QNA INTERRUPT SERVICE ROUTINE

```
3863 073142  042737  100000  050246          BIC     @#RCVIN,CSRBUF          ; CLEAR THE BIT
3864 073150  032703  000040                  BIT     @#INVRC,R3              ; RECEIVE LIST INVALID?
3865 073154  001405                          BEQ     20$                     ; NO
3866 073156  012761  024260  000004          MOV     @#RRING,LORCV(R1)       ; SET LOW ADDRESS OF RINGS
3867 073164  005061  000006                  CLR     HIRCV(R1)               ; CLEAR UPPER ADDRESS BITS
3868 073170                          20$:
3869 073170  032703  000200                  BIT     @#MXMTIN,R3             ; TRANSMIT INTERRUPT ??
3870 073174  001403                          BEQ     30$                     ; NO
3871 073176  005037  050260                  CLR     XFLAG                   ; YES, INDICATE THAT XMIT HAS OCCURRED
3872 073202  000404                          BR      50$                     ; LEAVE
3873 073204                          30$:
3874 073204  005237  050266                  INC     BCOUNT                  ; ELSE, NONSENSE INTERRUPT
3875 073210                          40$:
3876 073210  005237  050270                  INC     ERRFLG
3877 073214                          50$:
3878 073214  012603                          MOV     (SP)+,R3                ; RESTORE REGISTERS
3879 073216  012601                          MOV     (SP)+,R1                ; RESTORE REGISTERS
3880 073220  000002                          RTI                             ; AUF WIEDERSEHEN
3881
```

```
                                .SBTTL  XMIT   TRANSMIT QNA PACKETS
                        ;**
                        ; FUNCTIONAL DESCRIPTION:
                        ;
                        ;       THIS ROUTINE SETS UP THE TRANSMIT DESCRIPTOR RING ENTRIES AND WRITES
                        ;       THE RING ADDRESS INTO THE TRANSMIT LOW AND HIGH ADDRESS REGISTERS TO
                        ;       START TRANSMISSION.
                        ;
                        ; CALLING SEQUENCE
                        ;
                        ;       CALL    XMIT    @XMTSET OR @XMTDAT      ; XMTSET = SETUP, XMTDAT = DATA
                        ;               P$POP   P1
                        ;
                        ; INPUT
                        ;
                        ;       NONE
                        ;
                        ; IMPLICIT INPUT
                        ;
                        ;       NONE
                        ;
                        ; OUTPUT
                        ;
                        ;       P1 = STATUS - 0 = SUCCESS, -1 FOR FAILURE
                        ;
                        ; IMPLICIT OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; COMPLETION CODES
                        ;
                        ;       0 = SUCCESS
                        ;       -1 = FAILURE
                        ;
                        ; SIDE EFFECTS
                        ;
                        ;       NONE
                        ;
                        ; REGISTERS USED
                        ;
                        ;
                        ; DEBUG
                        ;
                        ;       NONE
                        ;--
073222                  XMIT::
073222  005037  050274          CLR     RETRYS
073226                  10$:
073226  012737  000020  002402  MOV     @16.,LSRTRY             ; SET CARRIER LOSS RETRY COUNT
073234  012737  000020  002404  MOV     @16.,NCRTRY             ; SET NO CARRIER RETRY COUNT
073242                  20$:
073242  013704  024220          MOV     XRGCUR,R4               ; MOVE RING ENTRY LOCATION INTO R4
073246  005714                  TST     (R4)                    ; MAKE SURE WE OWN THIS
073250  001405                  BEQ     30$                     ; BRANCH IF WE DO
073252  005764  000010          TST     STAT1(R4)               ; ELSE, SEE IF STATUS IS WRITTEN
073256  001002                  BNE     30$                     ; IF YES, CONTINUE
073260  000137  073650          JMP     160$                    ; ELSE REPORT THAT QNA STILLS OWNS IT
```

```
3940 073264                                  30$:
3941 073264                                          P$POP   R1                              ; GET XMIT TYPE
     073264  014501                                                                              MOV     -(R5),R1
3942 073266  001007                                  BNE     40$                             ; BRANCH IF NOT A SETUP PACKET
3943 073270  012764  130000  000002                 MOV     #MVALID!MEOM!MXSETP,DESC(R4) ; SET DESCRIPTOR BITS
3944 073276  012764  023570  000004                 MOV     #TGTADR,LOADD(R4)               ; SET BUFFER ADDRESS
3945 073304  000431                                  BR      50$
3946 073306                                  40$:
3947 073306  005737  003202                         TST     RSPFLG                          ; ARE WE IN LISTEN/RESPOND MODE?
3948 073312  001020                                 BNE     45$                             ; BRANCH IF YES, JUST VALIDATE AND XMIT
3949 073314  013764  050332  000006                 MOV     BUFLEN,WRDCNT(R4)               ; BUF LENGTH TO FIRST WORD OF NEXT RING
3950 073322  006264  000006                         ASR     WRDCNT(R4)                      ; TURN IT INTO A WORD COUNT
3951 073326  005464  000006                         NEG     WRDCNT(R4)                      ; TWO'S COMPLEMENT THE COUNT
3952 073332  016402  000004                         MOV     LOADD(R4),R2                    ; GET ADDRESS OF BUFFER
3953 073336  062702  000006                         ADD     #6,R2                           ; POINT TO SOURCE ADDRESS FIELD
3954 073342  012703  004010                         MOV     #PHYADR,R3                      ; POINT TO OUR PHYSICAL ADDRESS
3955 073346  012322                                 MOV     (R3)+,(R2)+                     ; FILL IN FIRST TWO ADDRESS BYTES
3956 073350  012322                                 MOV     (R3)+,(R2)+                     ; AND NEXT TWO ADDRESS BYTES
3957 073352  012322                                 MOV     (R3)+,(R2)+                     ; AND NEXT TWO ADDRESS BYTES
3958 073354                                  45$:
3959 073354  012764  120000  000002                 MOV     #MVALID!MEOM,DESC(R4)           ; SET VALID BIT
3960 073362  012764  024574  000004                 MOV     #XRG001,LOADD(R4)               ; SET BUFFER ADDRESS
3961 073370                                  50$:
3962 073370  005064  000000                         CLR     FLAG(R4)                        ; CLEAR FLAG WORD
3963 073374  005064  000010                         CLR     STAT1(R4)                       ; CLEAR STATUS WORD 1
3964 073400  005064  000012                         CLR     STAT2(R4)                       ; CLEAR STATUS WORD 2
3965 073404  012737  000001  050260                 MOV     #1,XFLAG                        ; SET TRANSMIT FLAG
3966 073412  013703  047772                         MOV     QNAADO,R3                       ; GET DEVICE ADDRESS
3967 073416  012737  001130  003760                 MOV     #1130,TIMER2                    ; SET TIMER
3968 073424                                  60$:
3969 073424  032763  000020  000016                 BIT     #MINVXM,CSR(R3)                 ; IS XMIT LIST INVALID?
3970 073432  001004                                 BNE     70$                             ; YES, CONTINUE
3971 073434  005737  003760                         TST     TIMER2                          ; TIMEOUT YET
3972 073440  001371                                 BNE     60$                             ; NO
3973 073442  000514                                 BR      180$                            ; TIMEOUT, REPORT ERROR AND LEAVE
3974 073444                                  70$:
3975 073444  010463  000010                         MOV     R4,LOXMT(R3)                    ; LOW ADDRESS OF XMIT BUFFER
3976 073450  005063  000012                         CLR     HIXMT(R3)                       ; START TRANSMISSION
3977 073454  012701  003760                         MOV     #TIMER2,R1                      ; SET UP TO WAIT FOR TRANSMIT TO COMPLETE
3978 073460  012711  001130                         MOV     #1130,(R1)
3979 073464                                  80$:
3980 073464  005737  050260                         TST     XFLAG                           ; SEE IF TRANSMIT DONE BIT SET
3981 073470  001407                                 BEQ     100$                            ;  IF SET, SKIP WAIT LOOP
3982 073472  005737  050252                         TST     FATFLG                          ; NON EXISTENT MEMORY?
3983 073476  001401                                 BEQ     90$                             ; BRANCH IF NOT
3984 073500  000503                                 BR      190$                            ; LEAVE ROUTINE
3985 073502                                  90$:
3986 073502  005711                                 TST     (R1)                            ;  ELSE, SEE IF TIMEOUT YET
3987 073504  001367                                 BNE     80$                             ;   NO, WAIT
3988 073506  000472                                 BR      180$                            ;   YES, EXIT
3989 073510                                  100$:
3990 073510  052737  000200  050246                 BIS     #MXMTIN,CSRBUF                  ; SET XMIT DONE BIT IN BUFFERED CSR
3991 073516  013763  050246  000016                 MOV     CSRBUF,CSR(R3)                  ; REARM XMIT INTERRUPTS
3992 073524  042737  000200  050246                 BIC     #MXMTIN,CSRBUF                  ; CLEAR THE BIT IN BUFFERED CSR
3993 073532  005764  000010                         TST     STAT1(R4)                       ; SEE WHO OWNS THIS ENTRY
3994 073536  001444                                 BEQ     160$                            ; IF QNA STILL OWNS THIS, SOMETHING IS WRONG
3995 073540  032764  010000  000002                 BIT     #MXSETP,DESC(R4)                ; SETUP PACKET?
```

```
3996 073546  001006                          BNE     105$                    ; BRANCH IF IT IS, WE DON'T COUNT SETUP STATISTICS
3997 073550                                  CALL    BMPCNT  R4,#BMPXMT      ; ELSE, GO UPDATE THE COUNTERS
     073550  012725  000001                                                        MOV     #BMPXMT,(R5)+
     073554  010425                                                              MOV     R4,(R5)+
     073556  004437  070656                                                      JSR     R4,PREG14
     073562  003736                                                              .WORD   BMPCNT-ANCHOR
3998 073564                          105$:
3999 073564  032764  040000  000010          BIT     #MXERRS,STAT1(R4)       ; SEE IF ANY ERRORS
4000 073572  001011                          BNE     120$                    ;  IF YES, BRANCH AND TAKE CARE OF THEM
4001 073574                          110$:
4002 073574                                  CALL    GETXNX  #0,#XRGCUR      ; UPDATE 'TRANSMIT RING CURRENT' POINTER
     073574  012725  024220                                                        MOV     #XRGCUR,(R5)+
     073600  012725  000000                                                        MOV     #0,(R5)+
     073604  004437  070656                                                      JSR     R4,PREG14
     073610  006534                                                              .WORD   GETXNX-ANCHOR
4003 073612  005003                          CLR     R3                      ;  INDICATE SUCCESS
4004 073614  000437                          BR      200$                    ; RETURN
4005 073616                          120$:
4006 073616  032764  016000  000010          BIT     #MABORT!MLOSS!MNOCAR,STAT1(R4) ; WAS MESSAGE STILL SENT?
4007 073624  001763                          BEQ     110$                    ; BRANCH IF YES, DO NEXT ONE
4008 073626                          140$:
4009 073626  005237  050274                  INC     RETRYS                  ; LET'S TRY IT AGAIN, KEEP COUNT OF THEM
4010 073632  022737  000003  050274          CMP     #3,RETRYS               ; HAVE WE RETRIED THREE TIMES YET?
4011 073640  100001                          BPL     150$                    ; IF NOT, THEN LET'S RETRANSMIT
4012 073642  000754                          BR      110$                    ; ELSE, CLEAN UP AND LEAVE
4013 073644                          150$:
4014 073644  000137  073370                  JMP     50$                     ;  TRY AGAIN
4015 073650                          160$:
4016 073650                                  ERRDF   10,EMSG10,ERR1          ; TRANSMIT RING BOOKKEEPING ERROR
     073650  104455                                                                  TRAP    C$ERDF
     073652  000012                                                                  .WORD   10
     073654  063670                                                                  .WORD   EMSG10
     073656  070306                                                                  .WORD   ERR1
4017 073660  000413                          BR      190$
4018 073662                          170$:
4019 073662                                  ERRDF   12,EMSG10,ERR1          ; BOOKKEEPING ERROR, XRGNXT SHOULD = XRGCUR
     073662  104455                                                                  TRAP    C$ERDF
     073664  000014                                                                  .WORD   12
     073666  063670                                                                  .WORD   EMSG10
     073670  070306                                                                  .WORD   ERR1
4020 073672  000406                          BR      190$
4021 073674                          180$:
4022 073674  005237  050272                  INC     TIMOUT
4023 073700                                  ERRHRD  8,EMSG08,ERR1           ; REPORT ERROR
     073700  104456                                                                  TRAP    C$ERHRD
     073702  000010                                                                  .WORD   8
     073704  063605                                                                  .WORD   EMSG08
     073706  070306                                                                  .WORD   ERR1
```

```
4024 073710
4025 073710  012703  177777          190$:   MOV     #-1,R3          ; ERROR INDICATOR
4026 073714                          200$:
4027 073714  005064  000000                  CLR     FLAG(R4)        ; CLEAR FLAG WORD
4028 073720  005064  000010                  CLR     STAT1(R4)       ; CLEAR STATUS WORD 1
4029 073724  005064  000012                  CLR     STAT2(R4)       ; CLEAR STATUS WORD 2
4030 073730  005064  000002                  CLR     DESC(R4)        ; INVALIDATE THE DESCRIPTOR
4031 073734                                  RETURN  R3              ; RETURN
     073734  010325                                          MOV     R3,(R5)+
     073736  000207                                          RTS     PC
```

```
                                        .SBTTL  RECEVE   RECEIVE QNA RING BUFFERS
                                ;**
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ;       THIS ROUTINE TAKES INCOMING DATA BUFFERS FROM THE QNA AND CHECK FOR
                                ;       ERRORS.  THIS PROCESS CONTINUES FOR ALL COMPLETED BUFFERS
                                ;
                                ; CALLING SEQUENCE
                                ;
                                ;       CALL    RECEVE
                                ;       P$POP   P1
                                ;
                                ; INPUT
                                ;
                                ;       NONE
                                ;
                                ; IMPLICIT INPUT
                                ;
                                ;       NONE
                                ;
                                ; OUTPUT
                                ;
                                ;       NUMBER OF PACKETS PROCESSED BY THIS ROUTINE
                                ;
                                ; IMPLICIT OUTPUT
                                ;
                                ;       NONE
                                ;
                                ; COMPLETION CODES
                                ;
                                ;       NONE
                                ;
                                ; SIDE EFFECTS
                                ;
                                ;       NONE
                                ;
                                ; REGISTERS USED
                                ;
                                ;       NONE
                                ;
                                ; DEBUG
                                ;
                                ;       NONE
                                ;-
073740                          RECEVE:
073740  005000                          CLR     R0                      ; SET STATUS TO SUCCESS
073742  005002                          CLR     R2                      ; CLEAR PACKETS HANDLED COUNTER
073744                          10$:
073744  005737  050256                  TST     NIRCNT                  ; SEE IF ANY PACKETS TO RECEIVE
073750  001002                          BNE     20$                     ; IF YES, CONTINUE
073752  000137  074626                  JMP     180$                    ; ELSE, EXIT
073756                          20$:
073756  013704  024222                  MOV     RRGCUR,R4               ; MOVE CURRENT RCV RING POINTER TO R4
073762  016401  000010                  MOV     STAT1(R4),R1            ; MOVE STATUS OF PACKET TO R1
073766  001002                          BNE     30$                     ; BRANCH IF WE OWN IT
073770  000137  074606                  JMP     160$                    ; STILL OWNED BY QNA, ERROR
```

```
4090 073774                                30$:
4091 073774  032764  020000  000010              BIT     #MRSETP,STAT1(R4)       ; WAS A SETUP PACKET RECEIVED?
4092 074002  001402                              BEQ     32$                     ; BRANCH IF NOT
4093 074004  000137  074560                      JMP     150$                    ; ELSE, LEAVE
4094 074010                                32$:
4095 074010                                      CALL    BMPCNT  R4,#BMPRCV      ; UPDATE COUNTERS FOR RECEIVE
     074010  012725  000000                                                              MOV     #BMPRCV,(R5)+
     074014  010425                                                                      MOV     R4,(R5)+
     074016  004437  070656                                                              JSR     R4,PREG14
     074022  003736                                                                      .WORD   BMPCNT-ANCHOR
4096 074024  016403  000004                      MOV     LOADD(R4),R3            ; MOVE BUFFER ADDRESS INTO R3
4097 074030  005737  002200                      TST     BLDFLG                  ; ARE WE IN BUILD MODE?
4098 074034  001035                              BNE     321$                    ; BRANCH IF WE ARE
4099 074036  023713  004010                      CMP     PHYADR,(R3)             ; CHECK TO SEE IF THE DESTINATION ADDRESS IS OUR
4100 074042  001010                              BNE     350$                    ; BRANCH IF NOT
4101 074044  023763  004012  000002              CMP     PHYADR+2,2(R3)          ; CHECK NEXT WORD OF ADDRESS
4102 074052  001004                              BNE     350$                    ; BRANCH IF NOT OURS
4103 074054  023763  004014  000004              CMP     PHYADR+4,4(R3)          ; CHECK FINAL ADDRESS WORD
4104 074062  001411                              BEQ     380$                    ; BRANCH IF EVERYTHING IS OKAY
4105
4106 074064                                350$:                                 ;; N.M.
4107 074064  032701  004000                      BIT     #MRUNT,R1               ; RUNT PACKET
4108 074070  001037                              BNE     322$                    ; IF YES, THROW OUT THE PACKET.
4109 074072                                      ERRHRD  61,EMSG61               ; REPORT UNKNOWN RECEIVER ERROR
     074072  104456                                                                      TRAP    C$ERHRD
     074074  000075                                                                      .WORD   61
     074076  066412                                                                      .WORD   EMSG61
     074100  000000                                                                      .WORD   0
4110 074102  000137  074544                      JMP     130$                    ; CONTINUE
4111
4112 074106                                380$:                                 ;; N.M.
4113 074106  032701  004000                      BIT     #MRUNT,R1               ; RUNT PACKET
4114 074112  001451                              BEQ     38$                     ; BRANCH IF NOT
4115 074114                                      ERRHRD  54,EMSG54               ; REPORT THE ERROR
     074114  104456                                                                      TRAP    C$ERHRD
     074116  000066                                                                      .WORD   54
     074120  065773                                                                      .WORD   EMSG54
     074122  000000                                                                      .WORD   0
4116 074124  000137  074544                      JMP     130$                    ; CONTINUE
4117
4118 074130                                321$:
4119 074130  023713  004032                      CMP     MCSTAD,(R3)             ; SEE IF CONSOLE ID MULITCAST IS DESTINATION
4120 074134  001011                              BNE     35$                     ; IF NOT, REPORT ERROR
4121 074136  023763  004034  000002              CMP     MCSTAD+2,2(R3)          ; SEE IF CONSOLE ID MULITCAST IS DESTINATION
4122 074144  001005                              BNE     35$                     ; IF NOT, REPORT ERROR
4123 074146  023763  004036  000004              CMP     MCSTAD+4,4(R3)          ; SEE IF CONSOLE ID MULITCAST IS DESTINATION
4124 074154  001001                              BNE     35$                     ; IF NOT, REPORT ERROR
4125 074156  000427                              BR      38$                     ; ITS A OKAY, MOVE ON
4126 074160                                35$:
4127 074160                                      ERRHRD  68,EMSG68               ; ELSE, REPORT THE ERROR
     074160  104456                                                                      TRAP    C$ERHRD
     074162  000104                                                                      .WORD   68
     074164  067121                                                                      .WORD   EMSG68
     074166  000000                                                                      .WORD   0
4128
4129 074170                                322$:
4130 074170  010403                              MOV     R4,R3                   ; GET DESCRIPTOR ADDRESS
```

```
4131 074172  023703  024232                    CMP     RRGLST,R3               ; IS THIS THE LAST DESCRIPTOR?
4132 074176  001003                            BNE     36$                     ; BRANCH IF NOT
4133 074200  013703  024216                    MOV     RRGSRT,R3               ; OTHERWISE, GET ADDRESS OF FIRST DESCRIPTOR
4134 074204  000402                            BR      37$
4135 074206                              36$:
4136 074206  062703  000014                    ADD     #14,R3                  ; POINT TO NEXT DESCRIPTOR
4137 074212                              37$:
4138 074212  005763  000010                    TST     STAT1(R3)               ; HAS THE NEXT DESCRIPTOR BEEN USED?
4139 074216  001406                            BEQ     375$                    ; BRANCH IF NOT
4140 074220  022737  000001  050256            CMP     #1,NIRCNT               ; IF IT IS USED, DO WE RECORD ONLY THE BOGUS RECEIVE
?
4141 074226  001002                            BNE     375$                    ; BRANCH IF THE GOOD AND THE BAD HAS BEEN RECORDED
4142 074230  005237  050256                    INC     NIRCNT                  ; ELSE, BUMP COUNT SO WE DON T MISS GOOD PACKET
4143 074234                              375$:
4144 074234  000433                            BR      50$                     ; THROW THE PACKET OUT
4145 074236                              38$:
4146 074236  026337  000014  050306            CMP     PROTOT(R3),PROTOO       ; SEE IF IT IS A LOOP SERVER TYPE CODE
4147 074244  001023                            BNE     40$                     ; IF NOT, CONT.
4148 074246  062703  000016                    ADD     @FASKIP,R3              ; POINT TO SKIP COUNT
4149 074252  062303                            ADD     (R3)+,R3                ; ADD SKIP COUNT TO DATA POINTER
4150 074254  022713  000002                    CMP     @FORWRD,(R3)            ; NEED TO FORWARD MESSAGE?
4151 074260  001004                            BNE     45$                     ; IF NOT, MOVE ON
4152 074262  005737  003202                    TST     RSPFLG                  ; ELSE, ARE WE ACTING AS A RESPONDER?
4153 074266  001416                            BEQ     50$                     ; IF NOT, JUST THROW IT OUT
4154 074270  000444                            BR      60$                     ; ELSE, GIVE IT TO THE CALLING ROUTINE
4155 074272                              45$:
4156 074272  022713  000001                    CMP     @REPLY,(R3)             ; ELSE, IS THIS A REPLY MESSAGE
4157 074276  001441                            BEQ     60$                     ; BRANCH IF YES
4158 074300                                    ERRHRD  52,EMSG52               ; ELSE REPORT ERROR
     074300  104456                                                                    TRAP    C$ERHRD
     074302  000064                                                                    .WORD   52
     074304  065633                                                                    .WORD   EMSG52
     074306  000000                                                                    .WORD   0
4159 074310  005200                            INC     R0                      ; PUT FAILURE STATUS INTO R0
4160 074312  000545                            BR      180$                    ; LEAVE
4161 074314                              40$:
4162 074314  026337  000014  050310            CMP     PROTOT(R3),PROTO2       ; ELSE CHECK FOR CONSOLE TYPE CODE
4163 074322  001427                            BEQ     60$                     ; IF OK, CONT.
4164 074324                              50$:
4165 074324  012764  000001  000002            MOV     #1,DESC(R4)             ; MARK IT AS A THROW AWAY
4166 074332                                    CALL    GETRNX  #0,@RRGCUR      ; AND THROW IT OUT
     074332  012725  024222                                                             MOV     @RRGCUR,(R5)+
     074336  012725  000000                                                             MOV     #0,(R5)+
     074342  004437  070656                                                             JSR     R4,PREG14
     074346  006450                                                                    .WORD   GETRNX-ANCHOR
4167 074350  020437  024226                    CMP     R4,RRGNXT               ; IS THE CURRENT POINTER SAME AS NEXT?
4168 074354  001007                            BNE     55$                     ; BRANCH IF NOT
4169 074356                                    CALL    GETRNX  #1,@RRGNXT      ; ELSE, UPDATE NEXT POINTER AS WELL
     074356  012725  024226                                                             MOV     @RRGNXT,(R5)+
     074362  012725  000001                                                             MOV     #1,(R5)+
     074366  004437  070656                                                             JSR     R4,PREG14
     074372  006450                                                                    .WORD   GETRNX ANCHOR
4170 074374                              55$:
4171 074374  005337  050256                    DEC     NIRCNT                  ; DECREMENT RECEIVE COUNTER
4172 074400  000510                            BR      170$                    ; EXIT
4173 074402                              60$:
4174 074402  032701  040000                    BIT     @MRERRS,R1              ; SEE IF ANY ERRORS
4175 074406  001462                            BEQ     140$                    ; BRANCH IF NOT
```

```
                                                BIT     @MRSETP!MDISC!MFRAM!MCRC!MOVF!MRUNT,R1 ; WAS THE PACKET TOO LARGE
                                                BNE     80$                     ; BRANCH IF NOT
                                                ERRHRD  53,EMSG53               ; REPORT THE ERROR
                                                                                        TRAP    C$ERHRD
                                                                                        .WORD   53
                                                                                        .WORD   EMSG53
                                                                                        .WORD   0
                                                BR      130$                    ; CONTINUE
                                        80$:
                                                BIT     @MDISC,R1               ; ARE THERE ANY OTHER VALID ERRORS?
                                                BEQ     140$                    ; BRANCH IF NOT
                                                BIT     @MOVF,R1                ; OVERFLOW?
                                                BEQ     90$                     ; BRANCH IF NOT
                                                ERRHRD  55,EMSG55               ; REPORT OVERFLOW ERROR
                                                                                        TRAP    C$ERHRD
                                                                                        .WORD   55
                                                                                        .WORD   EMSG55
                                                                                        .WORD   0
                                                BR      130$                    ; BRANCH
                                        90$:
                                                BIT     @MCRC,R1                ; CRC ERROR?
                                                BEQ     110$                    ; BRANCH IF NOT
                                                ERRHRD  56,EMSG56               ; ELSE REPORT IT
                                                                                        TRAP    C$ERHRD
                                                                                        .WORD   56
                                                                                        .WORD   EMSG56
                                                                                        .WORD   0
                                        100$:
                                                BIT     @MFRAM,R1               ; WAS FRAMING ERROR ALSO SET?
                                                BEQ     110$                    ; BRANCH IF NOT
                                                ERRHRD  57,EMSG57               ; REPORT THE ERROR
                                                                                        TRAP    C$ERHRD
                                                                                        .WORD   57
                                                                                        .WORD   EMSG57
                                                                                        .WORD   0
                                                BR      130$                    ; CONTINUE
                                        110$:
                                                BIT     @MSHORT,R1              ; PACKET LESS THAN 10 BYTES?
                                                BEQ     120$                    ; BRANCH IF NOT
                                                ERRHRD  60,EMSG60               ; REPORT IT
                                                                                        TRAP    C$ERHRD
                                                                                        .WORD   60
                                                                                        .WORD   EMSG60
                                                                                        .WORD   0
                                                BR      130$
                                        120$:
                                                ERRHRD  61,EMSG61               ; REPORT UNKNOWN RECEIVER ERROR
                                                                                        TRAP    C$ERHRD
                                                                                        .WORD   61
                                                                                        .WORD   EMSG61
                                                                                        .WORD   0
                                        130$:
                                                INC     RCVERR                  ; ELSE, INCREMENT RECEIVE ERROR COUNTER
                                                INC     R0                      ; SET STATUS TO FAILURE
                                                BR      50$                     ; UPDATE POINTERS AND RETURN
                                        140$:
```

```
4209 074554  005237  050300                  INC     RCVBUF                  ; INCREMENT GOOD BUFFERS RECEIVED COUNTER
4210 074560                          150$:
4211 074560  005202                  INC     R2                      ; KEEP COUNT OF HOW MANY BUFFERS RECEIVED
4212 074562  005337  050256          DEC     NIRCNT                  ; KEEP BOOKKEEPING IN ORDER
4213 074566                          CALL    GETRNX  #0,#RRGCUR      ; UPDATE "RECEIVE RING CURRENT" POINTER
     074566  012725  024222                                                  MOV     #RRGCUR,(R5)+
     074572  012725  000000                                                  MOV     #0,(R5)+
     074576  004437  070656                                                  JSR     R4,PREG14
     074602  006450                                                          .WORD   GETRNX-ANCHOR
4214 074604  000406                  BR      170$
4215 074606                          160$:
4216 074606  012700  000002          MOV     #2,R0                   ; SET FAILURE STATUS IN R0
4217 074612                          ERRDF   11,EMSG11,ERR1          ; RECEIVE RING BOOKKEEPING ERROR
     074612  104455                                                          TRAP    C$ERDF
     074614  000013                                                          .WORD   11
     074616  063735                                                          .WORD   EMSG11
     074620  070306                                                          .WORD   ERR1
4218 074622                          170$:
4219 074622  005064  000002          CLR     DESC(R4)                ; INVALIDATE CURRENT DESCRIPTOR
4220 074626                          180$:
4221 074626                          RETURN  R2                      ; RETURN WITH NUMBER OF ENTRYS HANDLED
     074626  010225                                                          MOV     R2,(R5)+
     074630  000207                                                          RTS     PC
4222
```

```
                                .SBTTL  BMPCNT   UPDATE STATISTICS
                        ;**
                        ; FUNCTIONAL DESCRIPTION:
                        ;
                        ;       THIS ROUTINE IS USED TO UPDATE THE STATISTICS COUNTERS FROM THE
                        ;       INFO IN STATUS WORDS 1 AND 2 IN BUFFER DESCRIP.OR ENTRIES
                        ;
                        ; CALLING SEQUENCE
                        ;
                        ;       CALL    BMPCNT  XFER TYPE,DESCRIPTOR ADDRESS
                        ;
                        ; INPUT
                        ;
                        ;       DESCRIPTOR ADDRESS   ADDRESS OF THE RING DESCRIPTOR ENTRY
                        ;       XFER TYPE - 0 = RING DESCRIPTOR FROM RCV LIST
                        ;                 - 1 = RING DESCRIPTOR FROM XMIT LIST
                        ;
                        ; IMPLICIT INPUT
                        ;
                        ;       NONE
                        ;
                        ; OUTPUT
                        ;
                        ;       COUNTER ARE UPDATED
                        ;
                        ; IMPLICIT OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; COMPLETION CODES
                        ;
                        ;       NONE
                        ;
                        ; SIDE EFFECTS
                        ;
                        ;       NONE
                        ;
                        ; REGISTERS USED
                        ;
                        ;       R1 - HOLDS ADDRESS OF COUNTER TABLE
                        ;       R2 - HOLDS ADDRESS OF RING DESCRIPTOR
                        ;       R3 - SCRATCH PAD
                        ;       R4 - SCRATCH PAD
                        ;
                        ; DEBUG
                        ;
                        ;       NONE
                        ;-
074632                  BMPCNT:
074632  012701  050144          MOV     #CNTRS,R1               ; GET ADDRESS OF COUNTER TABLE
074636                          P$POP   R2                      ; GET ADDRESS OF DESCRIPTOR
074636  014502                                                          MOV     -(R5),R2
074640                          P$POP   R3                      ; XMIT OR RECEIVE?
074640  014503                                                          MOV     -(R5),R3
074642  001504                  BEQ     BMPREC                  ; BRANCH IF RECEIVE PACKET
074644  016204  000010          MOV     STAT1(R2),R4            ; GET STATUS WORD
074650  032704  000400          BIT     #MFAIL,R4               ; DID WE GET HEARTBEAT FAILURE?
```

```
4279 074654  001402                        BEQ     3$                      ; BRANCH IF NOT
4280 074656  005261  000074                INC     C.COLL(R1)              ; ELSE, INCREMENT NUMBER OF COLLISION TEST FAILURES
4281                                ;
4282                                ; SEE IF XMISSION WAS ABORTED; IF IT WAS, SET REASON MASK.  NOTE THAT 16
4283                                ; RETRIES WILL HAVE BEEN ATTEMPTED FOR CARRIER LOSS AND NO CARRIER
4284                                ;
4285 074662                        3$:
4286 074662  032704  016000                BIT     @#ABORT!#LOSS!#NOCAR,R4 ; ABORT ?
4287 074666  001453                        BEQ     50$                     ; BRANCH IF NOT
4288 074670  005261  000072                INC     C.XABT(R1)              ; BUMP ABORT COUNT
4289 074674  032704  004000                BIT     @#ABORT,R4              ; IS ABORT SET?
4290 074700  001410                        BEQ     10$                     ; BRANCH IF NOT
4291 074702                                PUSH    R4                      ; SAVE R4
     074702  010446                                                                MOV     R4, (SP)
4292 074704  042704  177417                BIC     @177417,R4             ; CLEAR EVERYTHING BUT RETRY COUNT
4293 074710  001003                        BNE     5$                      ; BRANCH IF THE COUNT IS NOT ZERO
4294 074712  052761  000001  000070        BIS     @C.RTRY,C.XABB(R1)      ; ELSE, SET RETRY FAILURE BIT IN BIT MASK
4295 074720                        5$:
4296 074720                                POP     R4                      ; RESTORE CONTENTS OF R4
     074720  012604                                                                MOV     (SP)+,R4
4297 074722                        10$:
4298 074722  032704  010000                BIT     @#LOSS,R4               ; WAS LOST CARRIER REASON FOR ABORT?
4299 074726  001406                        BEQ     20$                     ; BRANCH IF NOT
4300 074730  032704  002000                BIT     @#NOCAR,R4              ; TEST FOR NO CARRIER SENSED
4301 074734  001003                        BNE     20$                     ; BRANCH IF ZERO
4302 074736  052761  000002  000070        BIS     @C. CAR,C.XABB(R1)      ; SET CARRIER LOST BIT
4303 074744                        20$:
4304 074744  032704  010000                BIT     @#LOSS,R4               ; WAS LOST CARRIER REASON FOR ABORT?
4305 074750  001006                        BNE     30$                     ; BRANCH IF NOT
4306 074752  032704  002000                BIT     @#NOCAR,R4              ; TEST FOR NO CARRIER SENSED
4307 074756  001403                        BEQ     30$                     ; BRANCH IF ZERO
4308 074760  032761  000004  000070        BIT     @C.SHRT,C.XABB(R1)      ; YES, IT'S SET, SET SHORTED BIT
4309 074766                        30$:
4310 074766  032761  000001  000070        BIT     @C.RTRY,C.XABB(R1)      ; RETRY ABORT SET?
4311 074774  001407                        BEQ     40$                     ; BRANCH IF NOT
4312 074776  032761  000004  000070        BIT     @C.SHRT,C.XABB(R1)      ; ELSE, IS THE SHORT BIT SET?
4313 075004  001003                        BNE     40$                     ; BRANCH IF NOT
4314 075006  052761  000010  000070        BIS     @C.OPEN,C.XABB(R1)      ; ELSE, RETRY ABORT AND NOT A SHORT, SET OPEN BIT
4315 075014                        40$:
4316 075014  000560                        BR      BMPEXI                  ; NO XMISSION OCCURRED, IGNORE BYTCNT
4317 075016                        50$:
4318 075016  042704  177417                BIC     @177417,R4              ; CLEAR ALL BUT RETRY COUNT
4319 075022  001460                        BEQ     BMPMUL                  ; COUNTER IS ZERO, NO RETRIES OCCURRED
4320 075024  162704  000020                SUB     @20,R4                  ; DECREMENT COUNT
4321 075030  001455                        BEQ     BMPMUL                  ; BRANCH IF ONLY ONE RETRY
4322 075032  162704  000020                SUB     @20,R4                  ; ELSE, DECREMENT COUNT AGAIN
4323 075036  001003                        BNE     60$                     ; BRANCH IF MORE THAN TWO
4324 075040  005261  000050                INC     C.PXM2(R1)              ; ELSE, BUMP 2 RETRIES COUNT
4325 075044  000447                        BR      BMPMUL                  ; FINISH COUNTERS
4326 075046                        60$:
4327 075046  005261  000044                INC     C.PXM3(R1)              ; INCREMENT 3 OR MORE RETRIES COUNTER
4328 075052  000444                        BR      BMPMUL                  ; FINISH COUNTERS
4329 075054                        BMPREC:
4330 075054  016204  000010                MOV     STAT1(R2),R4            ; GET STATUS WORD 1
4331 075060  032704  040000                BIT     @#MRERRS,R4             ; TEST ERROR BIT
4332 075064  001001                        BNE     5$                      ; BRANCH IF ERROR BIT IS SET
4333 075066  000436                        BR      BMPMUL
```

```
4334 075070
4335 075070  032704  010000            5$:     BIT     #MDISC,R4               ; IS RUNT OR DISCARD SET?
4336 075074  001424                            BEQ     40$                     ; BRANCH IF NOT
4337 075076                            10$:
4338 075076  005261  000016                    INC     C.RERR(R1)              ; INCREMENT RECEIVE ERROR COUNT
4339 075102  032704  000001                    BIT     #MOVF,R4                ; OVERFLOW?
4340 075106  001402                            BEQ     20$                     ; BRANCH IF NOT
4341 075110  005261  000030                    INC     C.RLIN(R1)              ; SET INTERNAL BUFFER ERROR
4342 075114                            20$:
4343 075114  032704  000002                    BIT     #MCRC,R4                ; CRC ERROR SET?
4344 075120  001411                            BEQ     35$                     ; BRANCH IF NOT
4345 075122  052761  000001  000014            BIS     #C.CRC,C.RERB(R1)       ; SET CRC ERROR BIT
4346 075130                            30$:
4347 075130  032704  000004                    BIT     #MFRAM,R4               ; FRAMING ERROR?
4348 075134  001403                            BEQ     35$                     ; BRANCH IF NOT
4349 075136  052761  000002  000014            BIS     #C.FRAM,C.RERB(R1)      ; ELSE, SET FRAMING ERROR BIT
4350 075144                            35$:
4351 075144  000504                            BR      BMPEXI                  ; EXIT
4352 075146                            40$:
4353 075146  032704  004000                    BIT     #MRUNT,R4               ; RUNT PACKET?
4354 075152  001404                            BEQ     BMPMUL                  ; BRANCH IF NOT
4355 075154  052761  000010  000014            BIS     #C.RUNT,C.RERB(R1)      ; SET RUNT PACKET BIT
4356 075162  000475                            BR      BMPEXI                  ; EXIT
4357 075164                            BMPMUL:
4358 075164                                    P$PUSH  LOADD(R2)               ; SAVE ADDRESS OF DATA BUFFER
                                                                                        MOV     LOADD(R2),(R5)+
     075164  016225  000004
4359 075170  005703                            TST     R3                      ; XMIT OR RECEIVE?
4360 075172  001405                            BEQ     5$                      ; BRANCH IF RECEIVE
4361 075174  016204  000006                    MOV     WRDCNT(R2),R4           ; GET XMITTED WORD COUNT
4362 075200  005404                            NEG     R4                      ; COMPLEMENT IT
4363 075202  006304                            ASL     R4                      ; TURN INTO BYTE COUNT
4364 075204  000413                            BR      15$                     ; CONTINUE
4365 075206                            5$:
4366 075206  042704  174377                    BIC     #174377,R4              ; EXTRACT RCV BYTE LENGTH 10:8
4367 075212                                    P$PUSH  R2                      ; SAVE THE PACKET ADDRESS
                                                                                        MOV     R2,(R5)+
     075212  010225
4368 075214  016202  000012                    MOV     STAT2(R2),R2            ; GET RCV BYTE LENGTH 7:0
4369 075220  042702  177400                    BIC     #^C377,R2               ; CLEAR COPY OF THE COUNT IN HIGH BYTE
4370 075224  050204                            BIS     R2,R4                   ; MERGE THE TWO BYTE LENGTHS
4371 075226  062704  000074                    ADD     #60.,R4                 ; MAKE UP FOR BYTES LOST DURING ADDRESS CHECK
4372 075232                                    P$POP   R2                      ; RESTORE THE PACKET ADDRESS
                                                                                        MOV     -(R5),R2
     075232  014502
4373 075234                            15$:
4374 075234  162704  000016                    SUB     #16,R4                  ; SUBTRACT PROTOCOL BYTES, COUNT ONLY DATA BYTES
4375 075240  005703                            TST     R3                      ; SEE IF XMIT OR RECEIVE
4376 075242  001430                            BEQ     50$                     ; BRANCH IF RECEIVE
4377 075244  005261  000034                    INC     C.PXMT(R1)              ; INCREMENT NUMBER OF PACKETS XMITTED
4378 075250  032762  000100  000002            BIT     #MHIBYT,DESC(R2)        ; DID DATA START ON BYTE BOUNDARY?
4379 075256  001401                            BEQ     10$                     ; BRANCH IF NOT
4380 075260  005304                            DEC     R4                      ; DECREMENT COUNT BY ONE
4381 075262                            10$:
4382 075262  032762  000200  000002            BIT     #MLOBYT,DESC(R2)        ; DID DATA START ON BYTE BOUNDARY?
4383 075270  001401                            BEQ     20$                     ; BRANCH IF NOT
4384 075272  005304                            DEC     R4                      ; DECREMENT COUNT BY ONE
4385 075274                            20$:
4386 075274                                    P$POP   R2                      ; GET DATA BUFFER ADDRESS
                                                                                        MOV     (R5)+,R2
     075274  014502
```

BMPCNT UPDATE STATISTICS

```
4387 075276  032762  000001  000000          BIT     #BIT00,DESTIN(R2)       ; MULTICAST ADDRESS?
4388 075304  001404                          BEQ     40$                     ; NO IF ZERO, MOVE ON
4389 075306  005261  000040                  INC     C.MXMT(R1)              ; BUMP MULTICAST XMIT COUNT
4390 075312  060461  000064                  ADD     R4,C.XMDB(R1)           ; ADD BYTE COUNT TO MULTICAST BYTE CNT
4391 075316                          40$:
4392 075316  060461  000060                  ADD     R4,C.XDAT(R1)           ; ADD BYTE COUNT TO TOTAL DATA
4393 075322  000415                          BR      BMPEXI                  ; EXIT
4394 075324                          50$:
4395 075324                          P$POP   R2                              ; GET DATA BUFFER ADDRESS
     075324  014502                                                                      MOV     -(R5),R2
4396 075326  005261  000004                  INC     C.PREC(R1)              ; BUMP PACKETS RECEIVED
4397 075332  032762  000001  000000          BIT     #BIT00,DESTIN(R2)       ; MULTICAST ADDRESS?
4398 075340  001404                          BEQ     60$                     ; BRANCH NOT SET
4399 075342  005261  000010                  INC     C.MREC(R1)              ; BUMP MULTICAST RECEIVE PACKET COUNT
4400 075346  060461  000024                  ADD     R4,C.RMDB(R1)           ; BUMP MULTICAST RECEIVE BYTE COUNT
4401 075352                          60$:
4402 075352  060461  000020                  ADD     R4,C.RDAT(R1)           ; BUMP RECEIVED DATA COUNT
4403 075356                          BMPEXI:
4404 075356                          RETURN                                  ; ADIOS, AVOIR, AUWEIDERSEN
     075356  000207                                                                     RTS     PC
```

```
                                        .SBTTL  EDPACK   ETHERNET DATA PACKING ROUTINE

                                ;++
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ;  .
                                ;       EDPACK                                          ETHERNET DATA PACK
                                ;
                                ;       THIS ROUTINE WILL CONVERT A STRING OF HEX CHARACTERS INTO A RIGHT
                                ;       JUSTIFIED BINARY  STREAM  (WITH LEADING ZEROS).  COMPATIBLE  WITH
                                ;       ETHERNET CONVENTIONS.   THE SOURCE STRING MUST BE FORMATTED USING
                                ;       EITHER  A  WORD  BY  WORD  HEX  DESCRIPTION OR A BYTE BY BYTE HEX
                                ;       DESCRIPTION.  THE  RETURNED  STRING  WILL  BE  BYTE  ORIENTED  AS
                                ;       REQUIRED BY THE ETHERNET:
                                ;       LO-BYTE-WORD0 HI-BYTE-WORD0 LO-BYTE-WORD1 HI-BYTE-WORD1, ETC.
                                ;
                                ;       INPUT ARGUMENTS         P1 - ADDRESS OF THE SOURCE  (HEX)  STRING
                                ;                                    TO BE CONVERTED TO A BINARY  STREAM.
                                ;                               P2   ADDRESS OF THE  DESIRED  DESTINATION
                                ;                                    BUFFER WHICH WILL ACCEPT BINARY DATA
                                ;                               P3   LENGTH (IN BYTES) OF THE DESTINATION
                                ;                                    BUFFER
                                ;       EXPLICIT OUTPUTS        P4 - ZERO IF SUCCESSFUL, -1 IF BUFFER TOO
                                ;                                    LONG OR ODD NUMBER OF HEX CHARACTERS
                                ;       IMPLIED OUTPUTS         THE  BUFFER  AT  P2  WILL CONTAIN A RIGHT
                                ;                               JUSTIFIED  BINARY STREAM W/ LEADING ZEROS
                                ;                               AND  CORRESPONDING  TO  HEX STRING AT R5.
                                ;       SUBORDINATE PROCEDURES  HXFORM. (STRIP NONHEX CHARACTERS)
                                ;                               HEXBIN. (HEX TO BINARY CONVERSION)
                                ;       CALLING PROCEDURE       CALL EDPACK P1,P2,P3    ;INPUT P1-P3 PARAMETERS
                                ;                               P$POP   P4              ;OUTPUT    P4 PARAMETER
                                ;
                                ;
                                ;-
075360                          LOCDST: .BLKB   74.             ;MAX NUMBER OF CHARACTERS THAT MAY BE ENTERRED
075472  000000                  SOURCE: .WORD                   ;SOURCE ADDRESS
075474                          EDPACK: P$POP   SOURCE,R4,R3            ;R4-DESTINATION, R3-NUMBER OF CHARS REQD
075474  014537  075472                                                      MOV     (R5),SOURCE
075500  014504                                                              MOV     -(R5),R4
075502  014503                                                              MOV     -(R5),R3
                                                                ;SOURCE-SRC ADDRESS,  ORIENT-WORD/BYTE?
075504  005002                          CLR     R2              ;ASSUME NO ERRORS, VALUE RETURNED
075506  006303                          ASL     R3              ;NUMBER OF CHARACTERS REQUIRED W/ "0'S
075510                                  CALL    HXFORM SOURCE,@LOCDST,R3
075510  010325                                                              MOV     R3,(R5)+
075512  012725  075360                                                      MOV     @LOCDST,(R5)+
075516  013725  075472                                                      MOV     SOURCE,(R5)+
075522  004437  070656                                                      JSR     R4,PREG14
075526  004672                                                              .WORD   HXFORM ANCHOR
075530                                  P$POP   R1,R2           ;R1=ADDRESS OF LAST CHAR
075530  014501                                                              MOV     -(R5),R1
075532  014502                                                              MOV     -(R5),R2
                                                                ;R2=SUCCESS/FAIL CODE (0/ 1)
075534  005702                          TST     R2              ;R1 WILL POINT TO RIGHTMOST CHARACTER
075536  001010                          BNE     10$             ;RIGHT JUSTIFY BUFFER
```

```
 4453                                                          ;CONVERT HEX AT LOCDST TO BINARY
 4454 075540  006203                    ASR     R3             ;R3 BYTES IN OUTPUT BIT STREAM
 4455 075542                            CALL    HEXBIN @LOCDST,R4,R3
      075542  010325                                                   MOV     R3,(R5)+
      075544  010425                                                   MOV     R4,(R5)+
      075546  012725  075360                                           MOV     @LOCDST,(R5)+
      075552  004437  070656                                           JSR     R4,PREG14
      075556  005030                                                   .WORD   HEXBIN ANCHOR
 4456
 4457 075560                    10$:    RETURN R2              ;RETURN WITH SUCCESS/FAILURE INDICATION
      075560  010225                                                   MOV     R2,(R5)+
      075562  000207                                                   RTS     PC
 4458
```

```
                                .SBTTL  HXFORM - HEX FORMAT ROUTINE
                        ;**
                        ; FUNCTIONAL DESCRIPTION:
                        ;
                        ;       HXFORM                                  HEX FORMAT ROUTINE
                        ;
                        ;       THIS ROUTINE WILL ACCEPT A HEX STRING, AND STRIP OUT  THE  NON-HEX
                        ;       CHARACTERS (WITHOUT FLAGGING AN ERROR FOR THE NON-HEX CHARACTERS).
                        ;       A NULL WILL BE LEFT AS THE END OF STRING DELIMITTER.
                        ;
                        ;       INPUT ARGUMENTS     P1  ADDRESS OF THE SOURCE  ASCII  STRING  (NULL
                        ;                               DELIMITTER AT END OF STRING)
                        ;                           P2  ADDRESS OF THE DESTINATION ASCII HEX STRING
                        ;                               (STRIPPED OF NON-HEX AND RIGHT  JUSTIFIED)
                        ;                           P3- THE NUMBER OF HEX  CHARACTERS  DESIRED  @R4
                        ;       EXPLICIT OUPUTS     P4- WILL  CONTAIN  THE  ADDRESS  OF  THE  LAST
                        ;                               CHARACTER IN THE OUTPUT BUFFER.
                        ;                           P5- THE SUCCESS/FAILURE (0/-1) INDICATOR
                        ;
                        ;       IMPLICIT OUTPUTS        THE BUFFER AT R4  WILL  CONTAIN  FORMATTED
                        ;                               HEX (ASCII) STRING.
                        ;       CALLING PROCEDURE       CALL HXFORM P1,P2,P3
                        ;                               P$POP P4,P5
                        ;
075564  000000          HXN:    .WORD
075566                  HXFORM: P$POP   R3,R2,HXN       ;ADDRESS OF SOURCE STRING
075566  014503                                                  MOV     -(R5),R3
075570  014502                                                  MOV     (R5),R2
075572  014537  075564                                         MOV     (R5),HXN
                                                        ;ADDRESS OF DESTINATION STRING
                                                        ;NUMBER OF HEX CHARACTERS DESIRED
                        ;
075576  010204                  MOV     R2,R4           ;DESTINATION ADDRESS, R2: DESTINATION POINTER
075600  063704  075564          ADD     HXN,R4          ;POINT TO END OF OUTPUT BUFFER (DESTINATION)
                                                        ;DO WHILE NO NULL FOUND IN SOURCE STRING
075604  112301          10$:    MOVB    (R3)+,R1        ;GET BYTE IN QUESTION (& POINT TO NEXT BYTE)
075606  142701  000200  20$:    BICB    #200,R1         ;ENSURE HI BIT IS LO (SEVEN BIT ASCII)
                                                        ;CHECK FOR VALID HEX CHARACTER
075612  001414                  BEQ     30$             ;THE NULL IS A VALID CHARACTER (BUT THE LAST)
075614  120127  000060          CMPB    R1,#60          ;IF GREATER THAN "0" THEN RANGE OK
075620  100413                  BMI     40$             ;40$-OUT OF RANGE, BYTE VALUE TOO SMALL
075622  120127  000072          CMPB    R1,#72          ;IF BYTE IS LESS THAN 72 AND >=60 THEN RANGE OK
075626  100406                  BMI     30$             ;RANGE OK IF >=60 AND <72 ELSE, CONTINUE CHECK
075630  120127  000101          CMPB    R1,#101         ;BYTE MUST BE >=101 TO CONTINUE CHECK
075634  100405                  BMI     40$             ;IF BYTE >71 AND <101 THEN BYTE OUT OF RANGE.
075636  122701  000106          CMPB    #106,R1         ;BYTE MUST BE <=106 TO BE OK, ELSE: NOT OK.
075642  100402                  BMI     40$             ;BYTE NOT OK, DON'T PLACE THIS BYTE IN OUTPUT.
                                                        ;PLACE THE HEX BYTE IN THE OUTPUT BUFFER.
075644  110122          30$:    MOVB    R1,(R2)+        ;BYTE IN RANGE, CONFIRMED, BYTE OK, POINT TO
075646  001405                  BEQ     60$             ;NEXT BYTE DEST ADDRESS. IF NULL, THEN EXIT.
                                                        ;(NO ERRORS)
                                                        ;IF NOT NULL, THEN CHECK FOR BUFFER OVERFLOW.
                                                        ;R4 POINTS TO LAST CHARACTER POSITION (OUTPUT)
075650  020402          40$:    CMP     R4,R2           ;R2 - PRESENT WRITE ADDRESS
075652  100354                  BPL     10$             ;(SHOULD BE POSITIVE RESULT OR 0) (MORE TO DO)
075654  012704  177777  50$:    MOV     #-1,R4          ;SET ERROR CONDITION (EXIT WITH ERROR)
075660  000404                  BR      70$             ;ERROR DETECTED EXIT PATH -> (TOO MANY CHARS)
```

```
4514
4515                                                    ;SUCCESSFUL EXIT PATH
4516 075662  005302                60$:  DEC     R2     ;POINT TO THE LAST ACTUAL CHARACTER AT DEST BFR
4517 075664  020402                      CMP     R4,R2  ;CHECK FOR MINIMUN OF 12 CHARACTERS.
4518 075666  001372                      BNE     50$    ;BRANCH IF LESS THAN 12, ERROR.
4519 075670  005004                      CLR     R4     ;INDICATE SUCESS
4520 075672                        70$:  RETURN  R2,R4  ;ADDRESS OF LAST CHARACTER (R2) IS P4
     075672  010425                                            MOV     R4,(R5)+
     075674  010225                                            MOV     R2,(R5)+
     075676  000207                                            RTS     PC
4521                                                    ;ERROR INDICATOR (R4) IS P5
```

```
                                        .SBTTL          HEXBIN   HEX TO BINARY CONVERSION
                                ;
                                ;       HEXBIN                          HEX TO BINARY CONVERSION PROCEDURE
                                ;
                                ;       THIS PROCEDURE WILL CONVERT A STRING OF HEX (ASCII) CHARACTERS
                                ;       DIRECTLY TO A BINARY STREAM.  THE DESTINATION BINARY STREAM WILL
                                ;       REQUIRE ONLY HALF AS MANY BYTES AS THE HEX STRING BECAUSE TWO
                                ;       HEX CHARACTERS ARE REQUIRED TO REPRESENT A SINGLE BINARY BYTE.
                                ;
                                ;       INPUTS          P1   SOURCE STRING ADDRESS (DELIMITTED BY A NULL)
                                ;                       P2   DESTINATION ADDRESS FOR THE BINARY DATA.
                                ;                       P3   THE NUMBER OF BINARY BYTES REQUIRED (HALF THE
                                ;                            NUMBER OF CHARACTERS AT P1.
                                ;       OUTPUTS                 NO EXPLICIT OUTPUTS
                                ;       IMPLIED OUTPUTS         THE BUFFER AT P2 WILL CONTAIN THE BINARY
                                ;                               STREAM, CONVERTED DIRECTLY FROM THE BUFFER
                                ;                               AT P1.
                                ;       SUBORDINATE PROCEDURES  NONE
                                ;       CALLING PROCEDURE       CALL    HEXBIN P1,P2,P3
                                ;--
                                HN:     .WORD
075700  000000                  CMPSTR: .ASCIZ  /0123456789ABCDEF/
075702  060  061  062
075705  063  064  065
075710  066  067  070
075713  071  101  102
075716  103  104  105
075721  106  000
                                        .EVEN
075724                          HEXBIN: P$POP   R1,R2,HN        ;R1=SOURCE STRING ADDRESS
075724  014501                                                          MOV     (R5),R1
075726  014502                                                          MOV     (R5),R2
075730  014537  075700                                                  MOV     -(R5),HN
                                                                ;R2=DESTINATION STRING ADDRESS
                                                                ;HN=NUMBER OF BYTES REQUIRED
075734  060237  075700                  ADD     R2,HN           ;HN NOW POINTS TO THE LAST_BYTE POSITION+1
075740  012704  075702          10$:    MOV     @CMPSTR,R4      ;POINTER IN THE COMPARE STRING
075744  121124                  20$:    CMPB    (R1),(R4)+      ;COMPARE CURRENT CHAR WITH A CHAR IN CMPSTR
075746  001376                          BNE     20$             ;REPEAT UNTIL CHARACTER FOUND IN LIST
075750  005201                          INC     R1              ;POINT TO THE NEXT ASCII BYTE
075752  162704  075703                  SUB     @CMPSTR+1,R4    ;R4 NOW CONTAINS THE ACTUAL BINARY VALUE FOR
                                                                ;THE NIBBLE DESCRIBED BY THE CURRENT BYTE.
                                                                ;NOTE: NIBBLE IS THE HI PORTION OF THE BYTE
075756  006304                          ASL     R4              ;MOVE NIBBLE TO THE HI END OF THE BYTE
075760  006304                          ASL     R4
075762  006304                          ASL     R4
075764  006304                          ASL     R4
075766  010403                          MOV     R4,R3           ;SAVE THE HI NIBBLE
075770  012704  075702                  MOV     @CMPSTR,R4      ;POINTER INTO COMPARE STRING
075774  121124                  30$:    CMPB    (R1),(R4)+      ;COMPARE CURRENT CHAR WITH A CHAR IN CMPSTR
075776  001376                          BNE     30$             ;REPEAT UNTIL MATCH FOUND IN CMPSTR LIST
076000  005201                          INC     R1              ;POINT TO THE NEXT ASCII BYTE
076002  162704  075703                  SUB     @CMPSTR+1,R4    ;R4 NOW CONTAINS THE ACTUAL BINARY VALUE FOR
                                                                ;THE NIBBLE DESCRIBED BY THE CURRENT BYTE.
                                                                ;NOTE: NIBBLE IS THE HI PORTION OF THE BYTE
```

HEXBIN HEX TO BINARY CONVERSION

```
 4569 076006  050403                    BIS     R4,R3           ;NOW THE TWO CHARACTERS HAVE MADE A SINGLE BYTE
 4570                                                            ;NOW PLACE THE COMPLETE BYTE IN THE DESTINATION
 4571 076010  110322                    MOVB    R3,(R2)+        ;AND POINT TO THE NEXT DESTINATION BYTE
 4572 076012  020237  075700            CMP     R2,HN           ;IF THE DESTINATION POINTER (R2) REACHES THE
 4573 076016  100750                    BMI     10$             ;LAST CHARACTER POSITION+1 (HN) THEN DONE.
 4574 076020                            RETURN                  ;RETURN TO CALLER
      076020  000207                                                    RTS     PC
```

```
                                        .SBTTL          BINHEX - BINARY TO HEX CONVERSION PROCEDURE
                                ;+---
                                ;       BINHEX                                  BINARY TO HEX CONVERSION PROCEDURE
                                ;
                                ;       THIS PROCEDURE WILL CONVERT A BINARY DATA STREAM INTO A HEX STRING.
                                ;
                                ;       INPUTS  -  P1- BINARY DATA BUFFER ADDRESS
                                ;                  P2- NUMBER OF BYTES IN THE BUFFER
                                ;                  P3- ADDRESS OF OUTPUT BUFFER FOR HEX STRING.
                                ;                       HEX CHARACTER PAIRS SEPERATED BY "-"'S
                                ;                      (NOTE: THIS BUFFER MUST BE AT LEAST 3*P2 BYTES LONG)
                                ;       OUTPUTS -  NONE
                                ;       IMPLICIT OUTPUTS        THE BUFFER AT P3 WILL CONTAIN THE HEX STRING
                                ;                               FOLLOWED BY A NULL CHARACTER.
                                ;       SUBORDINATE ROUTINES    NONE
                                ;       CALLING PROCEDURE       CALL BINHEX P1,P2,P3
                                ;---
076022  060     061     062     HEXC:   .ASCII  /0123456789ABCDEF/
076025  063     064     065
076030  066     067     070
076033  071     101     102
076036  103     104     105
076041  106
076042  000000                  LST:    .WORD
076044                          BINHEX: P$POP   R1,LST,R2       ;R1 HAS THE INPUT BUFFER ADDRESS
076044  014501                                                          MOV     -(R5),R1
076046  014537  076042                                                  MOV     -(R5),LST
076052  014502                                                          MOV     -(R5),R2
                                                                ;LST: HAS THE NUMBER OF BYTES IN INPUT BUFFER
                                                                ;R2 HAS THE OUTPUT BUFFER ADDRESS
076054  060137  076042                  ADD     R1,LST          ;LST IS NOW ADDRESS OF LAST SOURCE BYTE + 1
076060  112103                  10$:    MOVB    (R1)+,R3        ;GET THE CURRENT BYTE AND POINT TO NEXT BYTE
076062  110304                          MOVB    R3,R4           ;SEPARATE NIBBLES AND GET CHARACTERS SEPARATELY
076064  042703  177760                  BIC     #177760,R3      ;ONLY RIGHT BINARY NIBBLE REMAINS IN R3
076070  006204                          ASR     R4              ;SHIFT OVER FOR LEFT BINARY NIBBLE IN R4
076072  006204                          ASR     R4
076074  006204                          ASR     R4
076076  006204                          ASR     R4
076100  042704  177760                  BIC     #177760,R4      ;ONLY LEFT  BINARY NIBBLE REMAINS IN R4
                                                                ;R4 IS THE MOST  SIGNIFICANT NIBBLE (FIRST)
                                                                ;R3 IS THE LEAST SIGNIFICANT NIBBLE (SECOND)
076104  116422  076022                  MOVB    HEXC(R4),(R2)+  ;PUT THE ASCII BYTE INTO THE BUFFER HI POSITION
076110  116322  076022                  MOVB    HEXC(R3),(R2)+  ;PUT THE ASCII BYTE INTO THE BUFFER LO POSITION
076114  112722  000055                  MOVB    #'-,(R2)+       ;PUT - BETWEEN HEX PAIRS
076120  020137  076042                  CMP     R1,LST          ;RESULT IS NEGATIVE UNTIL R1=LST
076124  103755                          BLO     10$             ;UNTIL R1=LST. (TRANSFER ALL SOURCE BYTES)
076126  105042                          CLRB    (R2)            ;TERMINATE OUTPUT BUFFER WITH A NULL
076130                          RETURN
076130  000207                                                          RTS     PC
```

```
BLDLD   BUILD LOOP DIRECT DATA BUFFERS FOR TRANSMIT.

                        .SBTTL          BLDLD   BUILD LOOP DIRECT DATA BUFFERS FOR TRANSMIT.
                ;+
                ; FUNCTIONAL DESCRIPTION:
                ;               THIS SUBROUTINE BUILDS LOOP DIRECT PACKETS FOR TRANSMISSION
                ;               FROM THE QNA. SOURCE ADDRESS, DESTINATION ADDRESS,
                ;               PROT. TYPE, AND LOOP DIRECT HEADER INFO ARE ADDED
                ;               TO THE MESSAGE BUFFER.  THE MESSAGE BUFFER IS BUILT
                ;               BY A CALL TO BLDBUF.
                ;
                ; INPUTS -      P1 - THE ADDRESS OF THE DESTINATION ADDRESS (FROM NODE TABLE)
                ;               IMPLICIT - P$SIZE CONTAINS THE SIZE OF THE MESSAGE BUFFER DATA
                ;                          XRGNXT POINTS TO THE NEXT AVAILABLE RING ENTRY
                ;                          PHYADR HOLDS THE CURRENT LOCAL QNA PHYSICAL ADDRESS
                ;
                ; OUTPUTS -     BUILT MESSAGE PACKET.
                ;
                ; CALLING PROCEDURE - CALL BLDLD P1
                ;
                ; SIDE EFFECTS - THE MESSAGE PACKET IS BUILD AND CONTAINED IN THE
                ;                 BUFFER POINTED TO BY XRGNXT WHEN THE ROUTINE WAS ENTERED.
                ;                XRGNXT IS UPDATED TO POINT TO THE NEXT RING ENTRY
                ;
                ; REGISTER USAGE - R1 HOLDS ADDRESS OF DESTINATION ADDRESS
                ;                  R2 IS A POINTER FOR THE LOOP DIRECT HEADER INFO
                ;                  R3 HOLDS THE PACKET LENGTH
                ;                  R4 HOLDS ADDRESS OF NEXT RING ENTRY DATA BUFFER
                ;
                ;-
                BLDLD::
076132                  P$POP   R1                      ; PUT ADDRESS OF DEST. ADDRESS IN R1
076132  014501                                                  MOV     -(R5),R1
076134  013704  024224          MOV     XRGNXT,R4       ; MOVE NEXT PACKET ADDRESS TO R4
076140  016404  000004          MOV     LOADD(R4),R4    ; POINT R4 TO DATA BLOCK
076144  013703  002374          MOV     P$SIZE,R3       ; PUT MESSAGE SIZE INTO R3
076150  062703  000040          ADD     #40,R3          ; ADD HEADER AND LOOP DIRECT INFO TO LENGTH
076154  010337  050324          MOV     R3,XFER         ; PUT BYTES TRANSFERED INTO WORD
076160  162737  000016  050324  SUB     #16,XFER        ;  AND CORRECT FOR HEADER
076166  022703  001100          CMP     #XPKLEN,R3      ; SEE IF LONGER THAN ONE PACKET
076172  002474                  BLT     10$             ;  IF YES, ERROR
076174  010337  050332          MOV     R3,BUFLEN       ; PUT PACKET LENGTH IN BUFLEN
076200  012164  000000          MOV     (R1)+,DESTIN(R4)        ; MOVE FIRST TWO BYTES OF ADDRESS
076204  012164  000002          MOV     (R1)+,DESTIN+2(R4)      ; MOVE BYTES THREE AND FOUR
076210  011164  000004          MOV     (R1),DESTIN+4(R4)       ; MOVE BYTES FIVE AND SIX
076214  005064  000006          CLR     SOURCC(R4)              ; LEAVE BLANK SPACE FOR SOURCE ADDRESS
076220  005064  000010          CLR     SOURCC+2(R4)            ;   SIX BYTES WORTH
076224  005064  000012          CLR     SOURCC+4(R4)
076230  013764  050306  000014  MOV     PROTOO,PROTOT(R4)       ; MOVE PROTOCALL TYPE INTO HEADER
076236  012702  050346          MOV     #LOPDIR,R2              ; MOVE LOOPDIRECT FORMAT HEADER LOC. TO R2
076242  012264  000016          MOV     (R2)+,LDSKIP(R4)        ;  SKIP COUNT
076246  011264  000020          MOV     (R2),LDFCT1(R4)         ;  FUNCTION CODE (FORWARD)
076252  013764  004010  000022  MOV     PHYADR,LDADR1(R4)       ;  LOCAL NODE ADDRESS
076260  013764  004012  000024  MOV     PHYADR+2,LDADR1+2(R4)   ;   SIX BYTES
076266  013764  004014  000026  MOV     PHYADR+4,LDADR1+4(R4)   ;
076274  016264  000010  000030  MOV     10(R2),LDFCT2(R4)       ;  FUNCTION CODE (REPLY)
076302  013764  004010  000032  MOV     PHYADR,LDADR2(R4)       ;  LOCAL NODE ADDRESS
076310  013764  004012  000034  MOV     PHYADR+2,LDADR2+2(R4)   ;   SIX BYTES
076316  013764  004014  000036  MOV     PHYADR+4,LDADR2+4(R4)   ;
```

```
4673 076324  062704  000040                    ADD     #LDADR2+6,R4            ; POINT R4 TO FIRST DATA BYTE
4674 076330  010437  050334                    MOV     R4,CMPBUF               ; STORE BUFFER LOCATION FOR DATA COMPARE
4675 076334                                    CALL    BLDBUF R4               ; BUILD DATA BUFFER
     076334  010425                                                                    MOV     R4,(R5)+
     076336  004437  070656                                                            JSR     R4,PREG14
     076342  006756                                                                    .WORD   BLDBUF-ANCHOR
4676 076344                                    CALL    GETXNX  #0,#XRGNXT      ; UPDATE POINTER TO NEXT RING ENTRY
     076344  012725  024224                                                            MOV     #XRGNXT,(R5)+
     076350  012725  000000                                                            MOV     #0,(R5)+
     076354  004437  070656                                                            JSR     R4,PREG14
     076360  006534                                                                    .WORD   GETXNX-ANCHOR
4677 076362  000404                            BR      20$                     ; EXIT
4678 076364                            10$:    ERRDF   14,EMSG14,ERR1          ; MESSAGE SIZE TOO BIG
     076364  104455                                                                    TRAP    C$ERDF
     076366  000016                                                                    .WORD   14
     076370  063774                                                                    .WORD   EMSG14
     076372  070306                                                                    .WORD   ERR1
4679 076374                            20$:    RETURN
     076374  000207                                                                    RTS     PC
```

```
                                        .SBTTL          BLDFAS - BUILD PACKET FOR FULL ASSIST TRANSMISSION
                                ;+
                                ; FUNCTIONAL DESCRIPTION:
                                ;               THIS SUBROUTINE BUILDS FULL ASSIST PACKETS FOR TRANSMISSION
                                ;               FROM THE QNA.  SOURCE ADDRESS, DESTINATION ADDRESS, PROT. TYPE
                                ;               AND FULL ASSIST HEADER INFO ARE ADDED TO THE MESSAGE BUFFER.
                                ;               THE MESSAGE BUFFER IS BUILT BY A CALL TO BLDBUF.
                                ;
                                ; INPUTS -      P1 - THE ADDRESS OF THE DESTINATION ADDRESS (FROM NODE TABLE)
                                ;               IMPLICIT - P$SIZE CONTAINS THE SIZE OF THE MESSAGE BUFFER DATA
                                ;                          XRGNXT POINTS TO THE NEXT AVAILABLE RING ENTRY
                                ;                          PHYADR HOLDS THE CURRENT LOCAL NODE ADDRESS
                                ;
                                ; OUTPUTS       THE BUILT BUFFER
                                ;
                                ; CALLING PROCEDURE - CALL BLDFAS P1
                                ;
                                ; SIDE EFFECTS   XRGNXT SI UPDATED TO POINT TO THE NEXT RING ENTRY
                                ;
                                ; REGISTER USAGE - R1 HOLDS ADDRESS OF TARGET NODE ADDRESS
                                ;                 R2 HOLDS ADDRESS OF ASSIST NODE ADDRESS
                                ;                 R3 HOLDS THE PACKET LENGTH
                                ;                 R4 HOLDS ADDRESS OF NEXT RING ENTRY DATA BUFFER
                                ;
                                ;-
 4706 076376                    BLDFAS::
 4707 076376                            P$POP   R1,R2                   ; PUT ADDRESS OF TARGET ADDRESS INTO R1
      076376  014501                                                    MOV     -(R5),R1
      076400  014502                                                    MOV     -(R5),R2
 4708                                                                   ;  AND ADDRESS OF ASSIST ADDRESS INTO R2
 4709 076402  013704  024224                    MOV     XRGNXT,R4       ; MOVE NEXT PACKET ADDRESS TO R4
 4710 076406  016404  000004                    MOV     LOADD(R4),R4    ; POINT R4 TO DATA BLOCK
 4711 076412  013703  002374                    MOV     P$SIZE,R3       ; PUT MESSAGE SIZE INTO R3
 4712 076416  062703  000060                    ADD     #60,R3          ; ADD HEADER INFO TO LENGTH
 4713 076422  010337  050324                    MOV     R3,XFER         ; PUT 'BYTES TRANSFERED' INTO WORD
 4714 076426  162737  000016  050324            SUB     #16,XFER        ;  AND CORRECT FOR HEADER
 4715 076434  022703  001100                    CMP     #XPKLEN,R3      ; SEE IF LONGER THAN ONE PACKET
 4716 076440  002524                            BLT     10$             ;  IF YES, ERROR
 4717 076442  010337  050332                    MOV     R3,BUFLEN       ; PUT PACKET LENGTH IN BUFLEN
 4718 076446  011264  000000                    MOV     (R2),DESTIN(R4) ; MOVE FIRST TWO BYTES OF ADDRESS
 4719 076452  016264  000002  000002            MOV     2(R2),DESTIN+2(R4) ; MOVE BYTES THREE AND FOUR
 4720 076460  016264  000004  000004            MOV     4(R2),DESTIN+4(R4) ; MOVE BYTES FIVE AND SIX
 4721 076466  005064  000006                    CLR     SOURCC(R4)      ; LEAVE BLANK SPACE FOR SOURCE ADDRESS
 4722 076472  005064  000010                    CLR     SOURCC+2(R4)    ;   SIX BYTES WORTH
 4723 076476  005064  000012                    CLR     SOURCC+4(R4)
 4724 076502  013764  050306  000014            MOV     PROTOO,PROTOT(R4) ; MOVE PROTOCALL TYPE INTO HEADER
 4725 076510  012764  000C00  000016            MOV     #0,FASKIP(R4)   ;  SKIP COUNT
 4726 076516  012764  000002  000020            MOV     #2,FAFCT1(R4)   ;  FUNCTION CODE (FORWARD)
 4727 076524  011164  000022                    MOV     (R1),FAADR1(R4) ;  TARGET NODE ADDRESS
 4728 076530  016164  000002  000024            MOV     2(R1),FAADR1+2(R4) ;   SIX BYTES
 4729 076536  016164  000004  000026            MOV     4(R1),FAADR1+4(R4) ;
 4730 076544  012764  000002  000030            MOV     #2,FAFCT2(R4)   ;  FUNCTION CODE (FORWARD)
 4731 076552  011264  000032                    MOV     (R2),FAADR2(R4) ;  ASSIST NODE ADDRESS
 4732 076556  016264  000002  000034            MOV     2(R2),FAADR2+2(R4) ;   SIX BYTES
 4733 076564  016264  000004  000036            MOV     4(R2),FAADR2+4(R4) ;
 4734 076572  012764  000002  000040            MOV     #2,FAFCT3(R4)   ;  FUNCTION CODE (FORWARD)
 4735 076600  013764  004010  000042            MOV     PHYADR,FAADR3(R4) ;  LOCAL NODE ADDRESS
```

BLDFAS BUILD PACKET FOR FULL ASSIST TRANSMISSION

```
 4736 076606  013764  004012  000044          MOV     PHYADR+2,FAADR3+2(R4)   ;   SIX BYTES
 4737 076614  013764  004014  000046          MOV     PHYADR+4,FAADR3+4(R4)   ;
 4738 076622  012764  000001  000050          MOV     #1,FAFCT4(R4)           ;  FUNCTION CODE (REPLY)
 4739 076630  013764  004010  000052          MOV     PHYADR,FAADR4(R4)       ;  LOCAL NODE ADDRESS
 4740 076636  013764  004012  000054          MOV     PHYADR+2,FAADR4+2(R4)   ;   SIX BYTES
 4741 076644  013764  004014  000056          MOV     PHYADR+4,FAADR4+4(R4)   ;
 4742 076652  062704  000060                  ADD     #FAADR4+6,R4            ; POINT R4 TO FIRST DATA BYTE
 4743 076656  010437  050334                  MOV     R4,CMPBUF               ; STORE BUFFER LOCATION FOR DATA COMPARE
 4744 076662                                  CALL    BLDBUF R4               ; BUILD DATA BUFFER
      076662  010425                                                                      MOV     R4,(R5)+
      076664  004437  070656                                                              JSR     R4,PREG14
      076670  006756                                                                      .WORD   BLDBUF ANCHOR
 4745 076672                                  CALL    GETXNX  #0,#XRGNXT      ; UPDATE POINTER TO NEXT RING ENTRY
      076672  012725  024224                                                              MOV     #XRGNXT,(R5)+
      076676  012725  000000                                                              MOV     #0,(R5)+
      076702  004437  070656                                                              JSR     R4,PREG14
      076706  006534                                                                      .WORD   GETXNX ANCHOR
 4746 076710  000404                          BR      20$                     ; EXIT
 4747 076712                          10$:    ERRDF   29,EMSG14,ERR1          ; MESSAGE SIZE TOO BIG
      076712  104455                                                                      TRAP    C$ERDF
      076714  000035                                                                      .WORD   29
      076716  063774                                                                      .WORD   EMSG14
      076720  070306                                                                      .WORD   ERR1
 4748 076722                          20$:    RETURN
      076722  000207                                                              RTS     PC
```

```
                                        .SBTTL          BLDAST - BUILD TRANSMIT AND RECEIVE ASSIST PACKETS
                                ;*
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ;       THIS ROUTINE BUILDS RECEIVE ASSIST PACKETS
                                ;
                                BLDAST::
076724                                  P$POP   R1,R2                   ; PUT DESTINATION ADDRESS INTO R1
076724  014501                                                                  MOV     -(R5),R1
076726  014502                                                                  MOV     -(R5),R2
                                                                        ;  ASSIST ADDRESS INTO R2
076730  013704  024224                  MOV     XRGNXT,R4               ; MOVE NEXT PACKET ADDRESS TO R4
076734  016404  000004                  MOV     LOADD(R4),R4            ; POINT R4 TO DATA BLOCK
076740  013703  002374                  MOV     P$SIZE,R3               ; PUT MESSAGE SIZE INTO R3
076744  062703  000050                  ADD     #50,R3                  ; ADD HEADER INFO INTO LENGTH
076750  010337  050324                  MOV     R3,XFER                 ; PUT 'BYTES TRANSFERED  INTO WORD
076754  162737  000016  050324          SUB     #16,XFER                ;  AND CORRECT FOR HEADER
076762  022703  001100                  CMP     #XPKLEN,R3              ; SEE IF LONGER THAN ONE PACKET
076766  002511                          BLT     10$                     ;  IF YES, ERROR
076770  010337  050332                  MOV     R3,BUFLEN               ; PUT PACKET LENGTH INTO BUFLEN
076774  011164  000000                  MOV     (R1),DESTIN(R4)         ; MOVE DESTINATION ADDRESS INTO HEADER
077000  016164  000002  000002          MOV     2(R1),DESTIN+2(R4)      ;  SIX BYTES WORTH
077006  016164  000004  000004          MOV     4(R1),DESTIN+4(R4)
077014  005064  000006                  CLR     SOURCC(R4)              ; LEAVE BLANK SPACE FOR SOURCE ADDRESS
077020  005064  000010                  CLR     SOURCC+2(R4)            ;  SIX BYTES WORTH
077024  005064  000012                  CLR     SOURCC+4(R4)
077030  013764  050306  000014          MOV     PROTOO,PROTOT(R4)       ; MOVE PROTOCALL TYPE INTO HEADER
077036  012764  000000  000016          MOV     #0,FASKIP(R4)           ; SKIP COUNT
077044  012764  000002  000020          MOV     #2,FAFCT1(R4)           ; FUNCTION CODE (FORWARD)
077052  011264  000022                  MOV     (R2),FAADR1(R4)         ;  TARGET NODE ADDRESS
077056  016264  000002  000024          MOV     2(R2),FAADR1+2(R4)      ;   SIX BYTES
077064  016264  000004  000026          MOV     4(R2),FAADR1+4(R4)
077072  012764  000002  000030          MOV     #2,FAFCT2(R4)           ; FUNCTION CODE (FORWARD)
077100  013764  004010  000032          MOV     PHYADR,FAADR2(R4)       ;  LOCAL NODE ADDRESS
077106  013764  004012  000034          MOV     PHYADR+2,FAADR2+2(R4)   ;   SIX BYTES WORTH
077114  013764  004014  000036          MOV     PHYADR+4,FAADR2+4(R4)
077122  012764  000001  000040          MOV     #1,FAFCT3(R4)           ; FUNCTION CODE (REPLY)
077130  013764  004010  000042          MOV     PHYADR,FAADR3(R4)       ; LOCAL NODE ADDRESS
077136  013764  004012  000044          MOV     PHYADR+2,FAADR3+2(R4)
077144  013764  004014  000046          MOV     PHYADR+4,FAADR3+4(R4)
077152  062704  000050                  ADD     #FAADR3+6,R4            ; POINT R4 TO FIRST DATA BYTE
077156  010437  050334                  MOV     R4,CMPBUF               ; STORE BUFFER LOCATION FOR DATA COMPARE
077162                                  CALL    BLDBUF  R4              ; BUILD DATA BUFFER
077162  010425                                                                  MOV     R4,(R5)+
077164  004437  070656                                                          JSR     R4,PREG14
077170  006756                                                                  .WORD   BLDBUF ANCHOR
```

```
BLDAST  BUILD TRANSMIT AND RECEIVE ASSIST PACKETS

 4791 077172                                    CALL    GETXNX  #0,#XRGNXT     ; UPDATE RING POINTER
      077172  012725  024224                                                    MOV     #XRGNXT,(R5)+
      077176  012725  000000                                                    MOV     #0,(R5)+
      077202  004437  070656                                                    JSR     R4,PREG14
      077206  006534                                                           .WORD   GETXNX-ANCHOR
 4792 077210  000404                            BR      20$
 4793 077212                            10$:    ERRDF   36,EMSG14,ERR1          ; MESSAGE SIZE TOO BIG ERROR
      077212  104455                                                           TRAP    C$ERDF
      077214  000044                                                           .WORD   36
      077216  063774                                                           .WORD   EMSG14
      077220  070306                                                           .WORD   ERR1
 4794 077222                            20$:    RETURN
      077222  000207                                                          RTS     PC
```

```
                                        .SBTTL          BLDREQ - BUILD REQUEST ID PACKETS FOR TRANSMIT
                                ;-+
                                ; FUNCTIONAL DESCRIPTION:
                                ;               THIS SUBROUTINE BUILDS REQUEST ID PACKETS FOR TRANSMISSION
                                ;               FROM THE QNA.  SOURCE ADDRESS, DESTINATION ADDRESS,
                                ;               PROTOCALL TYPE, SEQUENCE NUMBER AND REQUEST ID
                                ;               HEADER INFOR ARE BUILT IN THIS SUBROUTINE.
                                ;
                                ; INPUTS        IMPLICIT - THE DESTINATION ADDRESS IS CONTAINED IN ADRBUF.
                                ;
                                ; OUTPUTS -     BUILT MESSAGE PACKET
                                ;
                                ; CALLING PROCEDURE - CALL BLDREQ
                                ;
                                ; SIDE EFFECTS - THE MESSAGE PACKET IS BUILT AND CONTAINED IN THE
                                ;               BUFFER POINTED TO BY XRGNXT WHEN THE ROUTINE WAS
                                ;               ENTERED.  XRGNXT IS UPDATED TO POINT TO THE NEXT ENTRY.
                                ;
                                ; REGISTER USAGE - R1 HOLDS ADDRESS OF DESTINATION ADDRESS.
                                ;                 R2 IS A POINTER FOR REQUEST ID HEADER INFO.
                                ;                 R4 HOLDS ADDRESS OF NEXT RING ENTRY DATA BUFFER.
                                ;
                                ;-+
                                BLDREQ::
077224  013704  024224                  MOV     XRGNXT,R4               ; MOVE NEXT PACKET ADDRESS TO R4
077230  016404  000004                  MOV     LOADD(R4),R4            ; POINT R4 TO DATA BLOCK
077234  012737  000100  050332          MOV     #100,BUFLEN             ; MOVE BUFFER SIZE TO BUFLEN
077242  012701  002316                  MOV     #ADRBUF,R1              ; MOVE ADDRESS OF DEST. ADR. TO R1
077246  012164  000000                  MOV     (R1)+,DESTIN(R4)        ; MOVE FIRST TWO BYTE OF DEST. ADR.
077252  012164  000002                  MOV     (R1)+,DESTIN+2(R4)      ;  AND BYTES THREE AND FOUR
077256  011164  000004                  MOV     (R1),DESTIN+4(R4)       ;  AND LAST TWO BYTES
077262  005064  000006                  CLR     SOURCC(R4)              ; LEAVE BLANK SPACE FOR SOURCE ADDR.
077266  005064  000010                  CLR     SOURCC+2(R4)            ;  SIX BYTES WORTH
077272  005064  000012                  CLR     SOURCC+4(R4)
077276  013764  050310  000014          MOV     PROTO2,PROTOT(R4)       ; MOVE PROTOCALL TYPE INTO HEADER
077304  012702  050340                  MOV     #REQID,R2               ; MOVE REQUEST ID HEADER LOC. TO R2
077310  012264  000016                  MOV     (R2)+,HEADER(R4)        ;  BYTE COUNT
077314  012264  000020                  MOV     (R2)+,HEADER+2(R4)      ;  FUNCTION CODE (REQUEST ID)
077320  011264  000022                  MOV     (R2),HEADER+4(R4)       ;  RECEIPT NO.
077324                                  CALL    GETXNX #0,@XRGNXT       ; UPDATE POINTER TO NEXT RING ENTRY
077324  012725  024224                                                          MOV     @XRGNXT,(R5)+
077330  012725  000000                                                          MOV     #0,(R5)+
077334  004437  070656                                                          JSR     R4,PREG14
077340  006534                                                                  .WORD   GETXNX ANCHOR
077342                                  RETURN
077342  000207                                                                  RTS     PC
```

```
                                .SBTTL          GET?NX - GET NEXT TRANSMIT OR RECEIVE RING ENTRY
                        ;
                        ; FUNCTIONAL DESCRIPTION
                        ;               THIS SUBROUTINE GETS THE NEXT TRANSMIT OR RECEIVE RING
                        ;               ENTRY.  IT IS ENTERED AT SEPERATE POINTS DEPENDING ON
                        ;               WHICH RING IS BEING USED.
                        ;
                        ; INPUTS        P1   FLAG INDICATING IF THE PREVIOUS DESCRIPTOR NEEDS REVALIDATING
                        ;               P2   THE ADDRESS OF THE RING POINTER TO BE UPDATED.
                        ;
                        ; OUTPUTS -     THE RING POINTER IS UPDATED TO POINT TO THE NEXT AVAILABLE
                        ;                 ENTRY.
                        ;
                        ; CALLING PROCEDURE - CALL  GETXNX  @P1,@P2     ; FOR TRANSMIT UPDATES
                        ;                     CALL  GETRNX  @P1,@P2     ; FOR RECEIVE UPDATES
                        ;
                        ; SIDE EFFECTS - NONE
                        ;
                        ; REGISTER USAGE - R1 POINTS TO THE FIRST ENTRY IN THE RING
                        ;                  R2 POINTS TO THE LAST ENTRY IN THE RING
                        ;                  R3 IS THE ADDRESS OF THE RING POINTER TO BE UPDATED
                        ;
                        ;---
077344                  GETRNX::
077344  013701  024216          MOV     RRGSRT,R1               ; MOVE FIRST RING ENTRY TO R1
077350  013702  024232          MOV     RRGLST,R2               ; MOVE LAST RING ENTRY TO R2
077354                          P$POP   R3                      ; SHOULD WE VALIDATE PREVIOUS DESCRIPTOR
077354  014503                                                          MOV     -(R5),R3
077356  001431                  BEQ     GETCOM                  ; BRANCH IF NOT
077360                  RNXPRV::
077360  013703  024234          MOV     RRGPRV,R3               ; ELSE, GET PREVIOUS DESCRIPTOR
077364  005063  000000          CLR     FLAG(R3)                ; REINIT THE FLAG
077370  005063  000010          CLR     STAT1(R3)               ; REINIT THE STATUS
077374  005063  000012          CLR     STAT2(R3)               ; REINIT STATUS 2
077400  012763  100000  000002  MOV     @MVALID,DESC(R3)        ; VALIDATE IT
077406  020203                  CMP     R2,R3                   ; IS THIS THE LAST DATA DESCRIPTOR?
077410  001404                  BEQ     10$                     ; BRANCH IF YES
077412  062737  000014  024234  ADD     @12.,RRGPRV             ; ELSE UPDATE PREVIOUS POINTER
077420  000410                  BR      GETCOM                  ; AND DO COMMON STUFF
077422                  10$:
077422  010137  024234          MOV     R1,RRGPRV               ; LAST ENTRY, POINT TO START OF LIST
077426  000405                  BR      GETCOM                  ; GO TO COMMON MODE
077430                  GETXNX::
077430                          P$POP   R3                      ; THROW AWAY FIRST PARAMETER
077430  014503                                                          MOV     (R5),R3
077432  013701  024214          MOV     XRGSRT,R1               ; MOVE FIRST RING ENTRY TO R1
077436  013702  024230          MOV     XRGLST,R2               ; MOVE LAST RING ENTRY TO R2
077442                  GETCOM:
077442                          P$POP   R3                      ; GET ADDRESS OF RING POINTER IN R3
077442  014503                                                          MOV     -(R5),R3
077444  021302                  CMP     (R3),R2                 ; SEE IF POINTER POINTS TO LAST RING
077446  001437                  BEQ     10$                     ;  IF YES, BRANCH
077450  062713  000014          ADD     @12.,(R3)               ;  ELSE, ADD ENTRY LENGTH TO POINTER
077454  020327  024226          CMP     R3,@RRGNXT              ; IS THIS THE NEXT RECEIVE POINTER?
077460  001073                  BNE     20$                     ; BRANCH IF NOT
077462  012704  024226          MOV     @RRGNXT,R4              ; ELSE GET NEXT RING DESCRIPTOR
077466  062704  000014          ADD     @14,R4                  ; POINT TO NEXT DESCRIPTOR
```

```
4892 077472  005764  000002                    TST     DESC(R4)          ; IS IT A THROW AWAY?
4893 077476  100407                            BMI     5$                ; BRANCH IF STILL VALID
4894 077500  001406                            BEQ     5$                ; BRANCH IF IT IS NOT A THROWAWAY
4895 077502  062737  000014  024226            ADD     #14,RRGNXT        ; ELSE THROW IT AWAY
4896 077510                                    P$PUSH  @RRGNXT           ; PUT THE ADDRESS BACK ONTO THE PARAMETER STACK
     077510  012725  024226                                                       MOV     @RRGNXT,(R5)+
4897 077514  000721                            BR      RNXPRV            ; AND UPDATE RECEVE RING POINTERS AGAIN
4898 077516                            5$:
4899 077516  023737  024222  024226            CMP     RRGCUR,RRGNXT     ; DID NEXT POINTER PASS BY CURRENT POINTER?
4900 077524  002051                            BGE     20$               ; IF NO, THEN EXIT
4901 077526  005737  050336                    TST     LAPCNT            ; ELSE, IS CURRENT POINTER A LAP(S) AHEAD?
4902 077532  001046                            BNE     20$               ; IF YES, THEN EXIT
4903 077534                                    ERRHRD  63,EMSG63         ; ELSE, REPORT ERROR
     077534  104456                                                               TRAP    C$ERHRD
     077536  000077                                                               .WORD   63
     077540  066526                                                               .WORD   EMSG63
     077542  000000                                                               .WORD   0
4904 077544  000441                            BR      20$               ; EXIT
4905 077546                            10$:
4906 077546  022713  024226                    CMP     @RRGNXT,(R3)      ; IS THIS THE "NEXT RECEIVE DESCR" POINTER?
4907 077552  001007                            BNE     12$               ; BRANCH IF NOT
4908 077554  005337  050336                    DEC     LAPCNT            ; ELSE, DECREMENT THE LAPCOUNT
4909 077560  100004                            BPL     12$               ; BRANCH IF "CURRENT RCV DESCR" POINTER NOT PASSED
4910 077562                                    ERRHRD  63,EMSG63         ; PRINT "NEXT POINTER PASSED BY CURRENT PONTER"
     077562  104456                                                               TRAP    C$ERHRD
     077564  000077                                                               .WORD   63
     077566  066526                                                               .WORD   EMSG63
     077570  000000                                                               .WORD   0
4911 077572                            12$:
4912 077572  022713  024222                    CMP     @RRGCUR,(R3)      ; IS THIS THE "CURRENT RCV DESC" POINTER?
4913 077576  001002                            BNE     18$               ; BRANCH IF NOT
4914 077600  005237  050336                    INC     LAPCNT            ; ELSE, INCREMENT THE LAP COUNTER
4915 077604                            18$:
4916 077604  010113                            MOV     R1,(R3)           ; POINT POINTER TO FIRST ENTRY IN RING
4917 077606  022703  024226                    CMP     @RRGNXT,R3        ; IS THIS THE NEXT RECEIVE POINTER?
4918 077612  001016                            BNE     20$               ; BRANCH IF NOT
4919 077614  012704  024226                    MOV     @RRGNXT,R4        ; GET NEXT RING DESCRIPTOR
4920 077620  062704  000014                    ADD     #14,R4            ; POINT TO NEXT DESCRIPTOR
4921 077624  005764  000002                    TST     DESC(R4)          ; IS IT A THROW AWAY?
4922 077630  100407                            BMI     20$               ; BRANCH IF STILL VALID
4923 077632  001406                            BEQ     20$               ; BRANCH IF IT IS NOT A THROWAWAY
4924 077634  062737  000014  024226            ADD     #14,RRGNXT        ; ELSE THROW IT AWAY
4925 077642                                    P$PUSH  @RRGNXT           ; PUT ADDRESS BACK ONTO PARAMETER STACK
     077642  012725  024226                                                       MOV     @RRGNXT,(R5)+
4926 077646  000644                            BR      RNXPRV            ; GO UPDATE POINTERS AGAIN
4927 077650                            20$:
4928 077650                                    RETURN
     077650  000207                                                               RTS     PC
```

```
                                        .SBTTL          BLDBUF   BUILD MESSAGE BUFFERS
                                ;+
                                ; FUNCTIONAL DESCRIPTION
                                ;               THIS SUBROUTINE CREATES A MESSAGE BUFFER TO BE USED
                                ;               FOR TRANSMISSION.
                                ;
                                ; INPUTS        P1   BUFFER LOCATION TO PUT BUILT MESSAGE
                                ;               IMPLICIT   P$SIZE CONTAINS THE SIZE THE BUFFER IS TO BE
                                ;                          P$TYPE CONTAINS THE MESSAGE TYPE
                                ; OUTPUTS -     IMPLICIT - BUFFER STARTING AT LOCATION P1 CONTAINS A
                                ;                          MESSAGE P$SIZE BYTES LONG USING THE MESSAGE
                                ;                          TYPE SPECIFIED BY P$TYPE.
                                ;
                                ; CALLING PROCEEDURE   CALL BLDBUF P1
                                ;
                                ; SIDE EFFECTS   NONE
                                ;
                                ; REGISTER USAGE - R1 POINTS TO THE NEXT BYTE OF STORED MESSAGE TO BE BUILT
                                ;                      IN THE MESSAGE BUFFER.
                                ;                 R2 = (MESSAGE TYPE X 2), USED AS OFFSET FOR POINTERS
                                ;                 R3 POINTS TO THE NEXT BYTE OF THE BUFFER UNDER CONSTRUCTION
                                ;                 R4 POINTS TO THE LAST BYTE OF THE BUFFER UNDER CONSTRUCTION
                                ;
                                ;-
                                BLDBUF::
 077652                                 P$POP   R3                      ; PUT BUFFER ADDRESS INTO R3
 077652  014503                                                                 MOV     -(R5),R3
 077654  013702  002372                 MOV     P$TYPE,R2               ; PUT MESSAGE TYPE INTO R2
 077660  006302                         ASL     R2                      ; MULTIPLY BY 2
 077662  013704  002374                 MOV     P$SIZE,R4               ; PUT SIZE INTO R4
 077666  060304                         ADD     R3,R4                   ; MAKE R4 = LAST BYTE OF BUFFER
 077670  016201  003360                 MOV     MSGAD(R2),R1            ; POINT R1 TO FIRST BYTE OF STORED MESSAGE
 077674  005037  050304                 CLR     COUNT                   ; CLEAR BYTE COUNTER
 077700  005237  050304         10$:    INC     COUNT                   ; COUNT NO. OF BYTES COPIED
 077704  112123                         MOVB    (R1)+,(R3)+             ; PUT BYTE IN BUFFER
 077706  026237  003342  050304         CMP     MSGCNT(R2),COUNT        ; ARE WE AT END OF STORED MESSAGE
 077714  001004                         BNE     20$                     ;  IF NO, CHECK IF DONE
 077716  016201  003360                 MOV     MSGAD(R2),R1            ;  ELSE, POINT R1 TO BEGINING
 077722  005037  050304                 CLR     COUNT                   ;  AND CLEAR COUNTER
 077726  020304                 20$:    CMP     R3,R4                   ; IS BUFFER FILLED?
 077730  001363                         BNE     10$                     ;  IF NO, LOOP
 077732                                 RETURN                          ;  ELSE, RETURN
 077732  000207                                                                 RTS     PC
```

```
                        .SBTTL          DATCMP - COMPARE DATA BUFFERS
                ;+
                ; FUNCTIONAL DESCRIPTION
                ;               THIS SUBROUTINE COMPARES TWO DATA BUFFERS BYTE BY BYTE.
                ;               IF COMPARISON ERRORS OCCURED, LOCATION, EXPECTED DATA
                ;               AND RECEIVED DATA ARE PRINTED OUT FOR THE FIRST FIVE
                ;               ERRORS.  THE TOTAL NUMBER OF ERRORS IS ALSO PRINTED.
                ;
                ; INPUTS        P1 - THE SIZE (IN BYTES) OF THE BUFFER TO BE COMPARED.
                ;               P2 - THE ADDRESS OF BUFFER TO COMPARE OTHER BUFFER AGAINST.
                ;               P3   THE ADDRESS OF THE SECOND BUFFER.
                ;
                ; OUTPUTS       P4   THE NUMBER OF COMPARISON ERRORS.
                ;
                ; CALLING PROCEDURE - CALL DATCMP P1,P2,P3
                ;                     P$POP P4
                ;
                ; SIDE EFFECTS   NONE.
                ;
                ; REGISTER USAGE - R1 CONTAINS THE COMPARE SIZE
                ;                  R2 CONTAINS THE ADDRESS OF THE BYTE BEING COMPARED IN BUFFER 1
                ;                  R3 CONTAINS THE ADDRESS OF THE BYTE BEING COMPARED IN BUFFER 2
                ;                  R4 CONTAINS THE BYTE OFFSET (BYTES FROM BEGINING OF BUFFER
                ;
                ;-
                DATCMP::
 077734                 P$POP   R1,R2,R3                ; PUT COMPARE SIZE IN R1
 077734  014501                                                 MOV     -(R5),R1
 077736  014502                                                 MOV      (R5),R2
 077740  014503                                                 MOV      (R5),R3
                                                        ;  BUFFER 1 ADDRESS IN R2 AND
                                                        ;  BUFFER 2 ADDRESS IN R3
 077742  005004                 CLR     R4              ; INITIALIZE BYTE OFFSET
 077744  005037  050314         CLR     TEMP            ;  AND ERROR COUNTER
 077750  005204         10$:    INC     R4              ; INCREMENT OFFSET COUNTER
 077752  121213                 CMPB    (R2),(R3)       ; COMPARE BUFFERS
 077754  001421                 BEQ     20$             ;  IF SAME, BRANCH
 077756  005237  050314         INC     TEMP            ; INCREMENT ERROR COUNTER
 077762  022737  000005  050314 CMP     #5,TEMP         ; IF MORE THAN 5 ERRORS,
 077770  002413                 BLT     20$             ; DON'T PRINT MESSAGE
 077772                         PRINTX  #CMPER1,R4,(R2),(R3)    ; PRINT ERROR MESSAGE
 077772  011346                                                 MOV     (R3),-(SP)
 077774  011246                                                 MOV     (R2),-(SP)
 077776  010446                                                 MOV     R4,-(SP)
 100000  012746  067602                                         MOV     #CMPER1,-(SP)
 100004  012746  000004                                         MOV     #4,-(SP)
 100010  010600                                                 MOV     SP,R0
 100012  104415                                                 TRAP    C$PNTX
 100014  062706  000012                                         ADD     #12,SP
 100020  005202         20$:    INC     R2              ; INCREMENT BUFFER 1 POINTER
 100022  005203                 INC     R3              ; INCREMENT BUFFER 2 POINTER
 100024  005301                 DEC     R1              ; DECREMENT COMPARE SIZE
 100026  001350                 BNE     10$             ;  IF NOT FINISHED, GO BACK FOR MORE
 100030  022737  000000  050314 CMP     #0,TEMP         ; WERE THERE ANY ERRORS?
 100036  001412                 BEQ     30$             ;  IF NO, EXIT
 100040                         PRINTX  #CMPER2,TEMP
 100040  013746  050314                                         MOV     TEMP,-(SP)
```

```
        100044  012746  067677                                          MOV     #CMPER2, (SP)
        100050  012746  000002                                          MOV     #2, (SP)
        100054  010600                                                  MOV     SP,R0
        100056  104415                                                  TRAP    C$PNTX
        100060  062706  000006                                          ADD     #6,SP
   5017 100064                          30$:    RETURN  TEMP          ; RETURN WITH ERROR COUNT ON STACK
        100064  013725  050314                                          MOV     TEMP,(R5)+
        100070  000207                                                  RTS     PC
```

```
                                        .SBTTL          WRITES - WRITE DATA ONTO SUMMARY TABLE
                                ;-+
                                ; FUNCTIONAL DESCRIPTION:
                                ;               THIS SUBROUTINE UPDATES THE SUMMARY TABLE DATA FOR
                                ;               THE NODES SPECIFIED IN THE CALL STATEMENT.  EITHER ONE
                                ;               OR TWO NODES CAN UPDATED PER CALL.  AFTER THE CALL,
                                ;               THE SUMMARY DATA COUNTERS ARE CLEARED.  THE SUMMARY TABLE
                                ;               IS CHECKED FOR A MATCHING NODE ADDRESS AND UPDATES THE
                                ;               DATE FOR THAT NODE, OR ADDS THE NODE TO THE TABLE IF IT
                                ;               DOESN'T EXIT.  AN ERROR IS REPORTED IF THE END OF THE TABLE
                                ;               IS REACHED.
                                ;
                                ; INPUTS        P1   THE NUMBER OF NODES TO UPDATE (1 OR 2).
                                ;               P2 - THE ADDRESS OF THE FIRST NODE ADDRESS.
                                ;               P3   THE ADDRESS OF THE SECOND NODE ADDRESS IF P1 = 2 OR
                                ;                    BLANK IF P1 = 1.
                                ;
                                ; OUTPUTS -     THE SUMMARY TABLE IS UPDATED.
                                ;
                                ; CALLING PROCEDURE - CALL WRITES P1,P2(,P3)
                                ;
                                ; SIDE EFFECTS - THE SUMMARY COUNTERS ARE CLEARED.
                                ;
                                ; REGISTER USAGE - R1 POINTS TO THE CURRENT LOCATION IN THE SUMMARY TABLE.
                                ;                 R2 POINTS TO THE NODE TO BE UPDATED'S ADDRESS.
                                ;                 R3 IS SCRATCH
                                ;                 R4 HOLDS THE SECOND NODE TO BE UPDATED ADDRESS.
                                ;
                                ;-+
100072                          WRITES: P$POP   TEMP                    ; SEE HOW MANY NODES TO WRITE
100072  014537  050314                                                          MOV     -(R5),TEMP
100076  023727  050314  000001          CMP     TEMP,#1                 ;  IF ONLY ONE, GET ADDRESS
100104  001002                         BNE     10$
100106                                  P$POP   R2
100106  014502                                                                  MOV     (R5),R2
100110  000402                         BR 20$
100112                          10$:    P$POP   R2,R4                   ;  IF TWO, GET BOTH ADDRESSES
100112  014502                                                                  MOV     -(R5),R2
100114  014504                                                                  MOV     -(R5),R4
100116  012701  002714          20$:    MOV     #STATBL,R1              ; MOVE STATISTICAL TABLE ADDRESS INTO R1
100122  005711                  30$:    TST     (R1)                    ; SEE IF SLOT IS EMPTY
100124  001415                         BEQ     40$                     ;  IF YES, BR
100126  021127  177777                 CMP     (R1),#-1                ; SEE IF TABLE FULL
100132  001454                         BEQ     80$                     ;  IF YES, ERROR
100134                                  CALL    CMPADR  R1,R2           ; LOOK FOR MATCHING ADDRESS
100134  010225                                                                  MOV     R2,(R5)+
100136  010125                                                                  MOV     R1,(R5)+
100140  004437  070656                                                          JSR     R4,PREG14
100144  026004                                                                  .WORD   CMPADR-ANCHOR
100146                                  P$POP   R3
100146  014503                                                                  MOV     -(R5),R3
100150  001412                         BEQ     50$                     ; IF YES, BR
100152  062701  000026                 ADD     #26,R1                  ; ELSE, POINT R1 TO NEXT ENTRY
100156  000761                         BR      30$                     ;  AND C[illegible] AGAIN
100160  011211                  40$:    MOV     (R2),(R1)               ; ADD N[illegible]DRESS TO TABLE
100162  016261  000002  000002         MOV     2(R2),2(R1)             ;  SIX B[illegible]S WORTH
100170  016261  000004  000004         MOV     4(R2),4(R1)             ;
```

WRITES - WRITE DATA ONTO SUMMARY TABLE

```
5067 100176  062701  000006          50$:    ADD     #6,R1               ; POINT R1 TO DATA
5068 100202  063721  050032                  ADD     S.NREC,(R1)+        ;  UPDATE SUMMARY DATA, RECEIVES NOT COMPLETE
5069 100206  063721  050030                  ADD     S.REC,(R1)+         ;   RECEIVES COMPLETE
5070 100212  063721  050034                  ADD     S.LEN,(R1)+         ;   LENGTH ERRORS
5071 100216  063721  050036                  ADD     S.COMP,(R1)+        ;   COMPARE ERRORS
5072 100222  063721  050040                  ADD     S.BYTE,(R1)+        ;   BYTES COMPARED
5073 100226  103001                          BCC     60$                 ;    IF OVERFLOW, INCREMENT NEXT WORD
5074 100230  005511                          ADC     (R1)
5075 100232  062701  000002          60$:    ADD     #2,R1               ;    POINT R1 TO NEXT DATA
5076 100236  063721  050042                  ADD     S.XFER,(R1)+        ;   BYTES TRANSFERED
5077 100242  103001                          BCC     70$                 ;    IF OVERFLOW, INCREMENT NEXT WORD
5078 100244  005511                          ADC     (R1)
5079 100246  062701  000002          70$:    ADD     #2,R1               ;    POINT R1 TO NEXT DATA
5080 100252  005337  050314                  DEC     TEMP                ; DECR NO OF NODES COUNTER
5081 100256  001414                          BEQ     90$                 ; IF NO MORE, EXIT
5082 100260  010402                          MOV     R4,R2               ; POINT R2 TO NEXT NODE
5083 100262  000715                          BR      20$                 ;  AND UPDATE SUMMARY DATA
5084 100264                          80$:    PRINTF  #TABFUL,#SUMM       ; PRINT TABLE FULL MESSAGE
     100264  012746  053422                                                      MOV     #SUMM, (SP)
     100270  012746  053274                                                      MOV     #TABFUL, (SP)
     100274  012746  000002                                                      MOV     #2, (SP)
     100300  010600                                                              MOV     SP,R0
     100302  104417                                                              TRAP    C$PNTF
     100304  062706  000006                                                      ADD     #6,SP
5085 100310  005037  050032          90$:    CLR     S.NREC              ; CLEAR SUMMARY DATA COUNTERS
5086 100314  005037  050030                  CLR     S.REC
5087 100320  005037  050034                  CLR     S.LEN
5088 100324  005037  050036                  CLR     S.COMP
5089 100330  005037  050040                  CLR     S.BYTE
5090 100334  005037  050042                  CLR     S.XFER
5091 100340                                  RETURN
     100340  000207                                                      RTS     PC
```

```
                                .SBTTL          BINDEC  CONVERT A 32 BIT BINARY NUMBER TO DECIMAL
                                ;--+
                                ; FUNCTIONAL DESCRIPTION:
                                ;               THIS SUBROUTINE CONVERTS A 32 BIT BINARY NUMBER TO
                                ;               A DECIMAL NUMBER REPRESENTED AS AN ASCIZ STRING.
                                ;
                                ; INPUTS -      P1 - THE ADDRESS OF THE FIRST WORD OF BINARY DATA
                                ;                    BITS 0-15.  THE SECOND WORD, BITS 16-31, IS
                                ;                    EXPECTED TO IMMEDIATELY FOLLOW THE FIRST WORD.
                                ;
                                ; OUTPUTS -     THE ASCII STRING WILL BE LOCATED STARTING AT DECSTR
                                ;
                                ; SIDE EFFECTS - NONE
                                ;
                                ; REGISTER USAGE - R1 POINTS TO BITS 0-15 OF BINARY DATA
                                ;                  R2 POINTS TO BITS 16-31 OF BINARY DATA
                                ;                  R3 POINTS TO THE OUTPUT STRING
                                ;                  R4 POINTS TO THE POWERS OF 10 TABLE
                                ;
                                ;--+
100342                          BINDEC::
100342                                  P$POP   R1                      ; PUT ADDRESS OF BINARY WORD INTO R1
100342  014501                                                                  MOV     -(R5),R1
100344  010546                          MOV     R5,-(SP)
100346  010102                          MOV     R1,R2                   ; PUT ADDRESS OF SECOND WORD INTO R2
100350  062702  000002                  ADD     #2,R2
100354  012703  100526                  MOV     #DECSTR,R3              ; PUT ADDRESS OF OUPUT STRING INTO R3
100360  012704  100456                  MOV     #TENPWR,R4              ; ADDRESS OF TEN POWER TABLE
100364  012705  100460                  MOV     #TENPWR+2,R5
100370  012737  000012  100444          MOV     #10.,40$
100376  005037  100542          10$:    CLR     PART                    ; CLEAR PARTIAL COUNTER
100402  161411                  20$:    SUB     (R4),(R1)               ; SUBTRACT 10 POWER
100404  005612                          SBC     (R2)
100406  161512                          SUB     (R5),(R2)
100410  002403                          BLT     30$                     ; BRANCH IF 10 POWER TOO LARGE
100412  005237  100542                  INC     PART                    ; ELSE ADD 1 TO PARTIAL
100416  000771                          BR      20$                     ; LOOP
100420  062411                  30$:    ADD     (R4)+,(R1)              ; RESTORE BINARY WORDS
100422  005512                          ADC     (R2)                    ; AND POINT R4 TO NEXT TABLE ENTRIES
100424  062412                          ADD     (R4)+,(R2)
100426  022525                          CMP     (R5)+,(R5)+
100430  052737  000060  100542          BIS     #'0,PART                ; CHANGE PARTIAL TO ASCII
100436  113723  100542                  MOVB    PART,(R3)+              ; AND PUT INTO OUTPUT STRING
100442  005327                          DEC     (PC)+                   ; HAVE WE DONE ALL 10 DIGITS
100444  000000                  40$:    .WORD   0
100446  001353                          BNE     10$                     ; IF NO, BRANCH
100450  105023                          CLRB    (R3)+                   ; IF YES, TERMINATE WITH ZERO
100452  012605                          MOV     (SP)+,R5
100454                                  RETURN
100454  000207                                                                  RTS     PC
100456                          TENPWR:
100456  145000                          145000                          ; 1.0 E09
100460  035632                          35632
100462  160400                          160400                          ; 1.0 E08
100464  002765                          2765
100466  113200                          113200                          ; 1.0 E07
100470  000230                          230
```

```
 5148 100472  041100                          041100                          ; 1.0 E06
 5149 100474  000017                          17
 5150 100476  103240                          103240                          ; 1.0 E05
 5151 100500  000001                          1
 5152 100502  023420                          23420                           ; 1.0 E04
 5153 100504  000000                          0
 5154 100506  001750                          1750                            ; 1.0 E03
 5155 100510  000000                          0
 5156 100512  000144                          144                             ; 1.0 E02
 5157 100514  000000                          0
 5158 100516  000012                          12                              ; 1.0 E01
 5159 100520  000000                          0
 5160 100522  000001                          1                               ; 1.0 E00
 5161 100524  000000                          0
 5162
 5163 100526                        DECSTR::.BLKB   12.                       ; 12 BYTES FOR ASCIZ OUTPUT STRING
 5164 100542  000000                PART::  .WORD   0                         ; PARTIAL COUNTER
```

COMMAND LINE TRAVERSE ROUTINES

```
                                        .SBTTL          COMMAND LINE TRAVERSE ROUTINES

                                ;**
                                ;       P$TRV SUBROUTINE
                                ;
                                ;PARSE THE COMMAND LINE SUBROUTINE
                                ;TAKE ACTIONS (VIA ACTION TREE) AS PARSING LINE
                                ;PARSING DIRECTIONS FROM "CLI PARSING NODES'
                                ;   REGS USED:
                                ;
                                ;       R1,R5=SCRATCH                                   P$NUM=NUMERIC CODE FROM DATA
                                ;       R2=ACTION CODE PARAMETER FROM TREE
                                ;       R3=PARSE TREE POINTER
                                ;       R4=INPUT STRING POINTER
                                ; CALLING SEQUENCE:
                                ;       JSR     PC,P$TRV
                                ; -

100544                          P$TRV:
100544  013704  003204                  MOV     P$BUFA,R4
100550  013703  003206                  MOV     P$TREE,R3
100554  121327  000003          P$TR5:  CMPB    (R3),#3                 ;SEE IF ONE OF FIRST THREE SPECIAL CODES
100560  003405                          BLE     10$                     ;IF YES, DON'T CHECK INPUT STRING
100562  105714                          TSTB    (R4)                    ;SEE IF ANY CHARS LEFT IN INPUT STRING
100564  001441                          BEQ     P$EXIT                  ;BR IF NO
100566  121327  000013                  CMPB    (R3),#11.               ;SEE IF SPECIAL CLI CHAR CODE OR ASCII
100572  003023                          BGT     30$                     ;BR IF REGULAR ASCII CHAR.
100574  111301                  10$:    MOVB    (R3),R1                 ;GET SPECIAL CHAR CODE INTO R5
100576  006301                          ASL     R1
100600  016101  100614                  MOV     20$(R1),R1              ;BUILD TRAVERSE ROUTINE ADDRESS
100604  062701  100614                  ADD     #20$,R1
100610  004711                          JSR     PC,(R1)                 ;JSR TO SPECIAL CLI TRAVERSE ROUTINE
100612  000760                          BR      P$TR5                   ;GO SEE IF MORE OF STRING LEFT

                                20$:    .WORD   TRVERR-20$              ;TRAVERSE TABLE FOR "CLI FUNCTIONS'
                                        .WORD   TRVEXI-20$              ;1
                                        .WORD   TRVBR-20$               ;2
                                        .WORD   TRVPIF-20$              ;3
                                        .WORD   TRVSPA-20$              ;4
                                        .WORD   TRVNUM-20$              ;5
                                        .WORD   TRVALP-20$              ;6
                                        .WORD   TRVALN-20$              ;7
                                        .WORD   TRVOCT-20$              ;8
                                        .WORD   TRVDEC-20$              ;9
                                        .WORD   TRVSTR-20$              ;10

                                ;NOT A SPECIAL CODE

                                30$:    CMPB    (R3),(R4)               ;SEE IF FIRST CHAR OF STRING IS A MATCH
                                        BEQ     40$                     ;BR IF A MATCH
100646  004737  100712                  JSR     PC,TRVBRC               ;IF NOT A MATCH, GO TAKE MISS BRANCH
100652  000740                          BR      P$TR5                   ; THEN GO BACK PT'G TO MISS NODE
100654  005204                  40$:    INC     R4                      ;IF A MATCH, INCR. CHAR POINTER
100656  004737  100672                  JSR     PC,TRVACT               ; GO DO ACTION DEFINED BY
100662  062703  000004                  ADD     #4,R3                   ; ACTION CODE IN CLI NODE, THEN
                                                                        ; ADJUST PTR TO NEXT CLI NODE
```

```
5223 100666  000732                    BR      P$TR5
5224
5225 100670  000207            P$EXIT: RTS     PC                      ;RETURN FROM PARSER
5226
5227                                    ;  -  -  - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
5228
5229                                    ;GOTO USER ACTION ROUTINE
5230 100672  116302  000001    TRVACT: MOVB    1(R3),R2                ;GET ACTION CODE FROM CLI NODE
5231 100676  042702  177400            BIC     #177400,R2              ;CLEAR ANY SIGN EXTENSION
5232 100702  013701  003210            MOV     P$ACT,R1                ;GET ADDRESS OF CLI ACTION ROUTINE
5233 100706  004711                    JSR     PC,(R1)                 ;GO DO ACTION DEFINED BY CODE
5234 100710  000207                    RTS     PC                      ;RETURN TO CALLING CODE
5235
5236                                    ;TAKE BRANCH IN TREE
5237 100712  016301  000002    TRVBRC: MOV     2(R3),R1                ;GET BRANCH DISPLACEMENT FROM TREE
5238 100716  060103                    ADD     R1,R3                   ; AND POINT R3 TO THE "MISS" NODE
5239 100720  000207                    RTS     PC                      ; RETURN TO P$TRV
5240
5241                                    ;NO BRANCH TAKEN
5242 100722  062703  000004    TRVNOB: ADD     #4,R3                   ;THINGS OK, UPDATE R3 TO POINT TO NEXT
5243 100726  000207                    RTS     PC                      ; NODE AND RETURN TO P$TRV
5244
5245                                    ;------- --------- ----- -- - -  -- ---------- -- - -
5246                                    ;ERROR HANDLING
5247 100730  004737  100672    TRVERR: JSR     PC,TRVACT               ;TAKE ERROR ACTION
5248 100734  112737  177777  003221            MOVB    #-1,P$GDBD      ;SET ERROR RETURN FLAG
5249 100742  005726                    TST     (SP)+                   ;GET RID OF "JSR PUSH TO TRVERR'
5250 100744  000137  100670            JMP     P$EXIT                  ;RETURN DIRECT TO EXIT OF P$TRV ROUTINE
5251
5252                                    ;EXIT ACTION CODE
5253 100750  004737  100672    TRVEXI: JSR     PC,TRVACT               ;TAKE EXIT ACTION
5254 100754  105037  003221            CLRB    P$GDBD                  ;SET GOOD/BAD FLAG TO "SUCCESS (0)'
5255 100760  005726                    TST     (SP)+                   ;GET RID OF "JSR PUSH TO TRVEXI
5256 100762  000137  100670            JMP     P$EXIT                  ;RETURN DIRECT TO EXIT OF P$TRV ROUTINE
5257
5258                                    ;BRANCH ACTION CODE
5259 100766  004737  100672    TRVBR:  JSR     PC,TRVACT               ;GO TAKE BRANCH ACTION
5260 100772  000137  100712            JMP     TRVBRC
5261
5262                                    ;BRANCH-IF ACTION CODE
5263 100776  004737  100672    TRVBIF: JSR     PC,TRVACT
5264 101002  105737  003221            TSTB    P$GDBD                  ;SEE IF P$GDBD SET OR CLEARED BY ACTION
5265 101006  001402                    BEQ     10$                     ;IF CLEAR FALL THRU TO NEXT NODE
5266 101010  000137  100712            JMP     TRVBRC                  ;ELSE TAKE THE "MISS' BRANCH
5267 101014  000137  100722    10$:    JMP     TRVNOB                  ;JUST UPDATE TO NEXT NODE IF THINGS OK
5268
5269                                    ;SPACE ACTION CODE
5270 101020  005001            TRVSPA: CLR     R1                      ;CLEAR "SPACE OR TAB FOUND" FLAG
5271 101022  121427  000011    10$:    CMPB    (R4),#11                ;SEE IF CHAR. IN CMD LINE= TAB
5272 101026  001003                    BNE     20$                     ;BR IF NO, NOT A TAB
5273 101030  005204                    INC     R4                      ;INC INPUT STRING POINTER
5274 101032  005201                    INC     R1                      ;INDICATE A TAB FOUND
5275 101034  000772                    BR      10$                     ;GO CHECK NEXT CHAR
5276
5277 101036  121427  000040    20$:    CMPB    (R4),#40                ;SEE IF CHAR. IN CMD LINE= SPACE
5278 101042  001003                    BNE     30$                     ;BR IF NO, NON SPACE OR NON TAB CHAR.
5279 101044  005204                    INC     R4                      ;INC INPUT STRING POINTER
```

```
5280 101046  005201                        INC     R1                      ;INDICATE A SPACE FOUND
5281 101050  000764                        BR      10$                     ;GO CHECK NEXT CHAR
5282 101052  005701                30$:    TST     R1                      ;SEE IF ANY SPACES OR TABS FOUND
5283 101054  001404                        BEQ     40$                     ;BR IF NO, TAKE NO ACTION
5284 101056  004737  100672                JSR     PC,TRVACT               ;GO TAKE ACTION IF ANY FOUND
5285 101062  000137  100722                JMP     TRVNOB                  ;JUST GO UPDATE R3 TO NEXT NODE IF OK
5286 101066  000137  100712          40$:  JMP     TRVBRC                  ;TAKE BRANCH (MISS) IF NONE FOUND
5287
5288
5289 101072  012737  000012  003216  TRVDEC: MOV   #10.,P$RADX             ;USE DECIMAL AS RADIX AND ASSUME .
5290 101100  000137  101112          JMP     TRVNMA
5291 101104                        TRVOCT: ;(SAME AS TRVNUM SINCE DEFAULT RADIX IS OCTAL)
5292 101104  012737  000010  003216  TRVNUM: MOV   #8.,P$RADX              ;USE OCTAL AS RADIX AND ASSUME .
5293 101112                        TRVNMA: PUSH    R5
     101112  010546                                                        MOV     R5,-(SP)
5294 101114  005001                        CLR     R1                      ;CLEAR DIGIT COUNTER
5295 101116  121427  000053                CMPB    (R4),#'.                ;SEE IF THERE'S A . SIGN THERE
5296 101122  001001                        BNE     10$                     ; BR IF NO
5297 101124  000406                        BR      20$                     ; ELSE P$RADX ALREADY SAYS ., JUST BR
5298 101126  121427  000055          10$:  CMPB    (R4),#'-                ;SEE IF THERE'S A - SIGN THERE
5299 101132  001004                        BNE     30$                     ; BR IF NO
5300 101134  112737  177777  003217        MOVB    #-1,P$RADX+1            ;SET "MINUS FLAG" (HI BYTE OF P$RADX)
5301 101142  005204                20$:    INC     R4                      ;BUMP R4 TO POINT TO FIRST CHAR
5302
5303 101144  121427  000060          30$:  CMPB    (R4),#60                ;SEE IF CHAR. LESS THAN A "0"
5304 101150  002434                        BLT     60$                     ;BR IF YES (NOT NUMERIC)
5305 101152  121427  000067                CMPB    (R4),#67                ;SEE IF CHAR. GREATER THAN A  "7"
5306 101156  003426                        BLE     50$                     ; BR IF YES
5307 101160  123727  003216  000012        CMPB    P$RADX,#10.             ;SEE IF IN DECIMAL MODE
5308 101166  001417                        BEQ     40$                     ; BR IF YES (CAN USE HIGHER LIMIT)
5309 101170  121427  000071                CMPB    (R4),#71                ;SEE IF DIGIT WAS A 8 OR 9
5310 101174  003022                        BGT     60$                     ;BR IF NON-NUMERIC
5311 101176                                PRINTF  #CLIBRX                 ;ELSE WAS A 8 OR 9 WHEN IN OCTAL RADIX
     101176  012746  052222                                                        MOV     #CLIBRX,-(SP)
     101202  012746  000001                                                        MOV     #1,-(SP)
     101206  010600                                                                MOV     SP,R0
     101210  104417                                                                TRAP    C$PNTF
     101212  062706  000004                                                        ADD     #4,SP
5312 101216  112737  177777  003221        MOVB    #-1,P$GDBD              ;SET ERROR RETURN FLAG
5313 101224  000475                        BR      110$                    ; PRINT ERROR AND TAKE MISS
5314
5315 101226  121427  000071          40$:  CMPB    (R4),#71                ;SEE IF CHAR. GREATER THAN A  9
5316 101232  003003                        BGT     60$                     ;BR IF YES (NOT NUMERIC)
5317 101234  005204                50$:    INC     R4                      ;UPDATE CMD LINE PTR TO NEXT CHAR.
5318 101236  005201                        INC     R1                      ;INDICATE A NUMERIC FOUND
5319 101240  000741                        BR      30$                     ;GO LOOK AT NEXT CHAR.
5320
5321 101242  005701                60$:    TST     R1                      ;SEE IF FOUND ANY NUMERICS
5322 101244  001465                        BEQ     110$                    ;BR IF NO, TAKE "MISS" BRANCH
5323 101246  010405                        MOV     R4,R5                   ;GET POINTER TO START OF NUMERIC STRING
5324 101250  160105                        SUB     R1,R5
5325 101252  005037  003214                CLR     P$NUM                   ;CLEAR LOC. WHERE VALUE WILL BE STORED
5326 101256  112502                70$:    MOVB    (R5)+,R2                ;GET ASCII CHAR AND CONVERT IT TO A #
5327 101260  162702  000060                SUB     #60,R2
5328 101264  006337  003214                ASL     P$NUM                   ;SHIFT CURRENT VALUE TO MAKE ROOM
5329 101270  103440                        BCS     100$                    ;ERROR IF NUMBER TOO BIG
5330 101272  013737  003214  003212        MOV     P$NUM,P$CNT             ;SAVE FOR LATER IN CASE DECIMAL RADIX
```

```
5331 101300 006337 003214          ASL     P$NUM
5332 101304 103432                 BCS     100$                ;ERROR IF NUMBER TOO BIG
5333 101306 006337 003214          ASL     P$NUM
5334 101312 103427                 BCS     100$                ;ERROR IF NUMBER TOO BIG
5335 101314 123727 003216 000012   CMPB    P$RADX,#10.         ;SEE IF DECIMAL RADIX
5336 101322 001004                 BNE     80$                 ;BR IF NOT EQUAL
5337 101324 063737 003212 003214   ADD     P$CNT,P$NUM
5338 101332 103417                 BCS     100$                ;ERROR IF NUMBER TOO BIG
5339 101334 060237 003214          80$:    ADD     R2,P$NUM
5340 101340 103414                 BCS     100$                ;ERROR IF NUMBER TOO BIG
5341 101342 005301                 DEC     R1
5342 101344 001344                 BNE     70$
5343 101346 105737 003217          TSTB    P$RADX+1            ;SEE IF NUM WAS PRECEDED BY A   SIGN
5344 101352 001402                 BEQ     90$                 ; BR IF NO
5345 101354 005437 003214          NEG     P$NUM               ; ELSE NEGATE THE NUMBER BEFORE LEAVING
5346 101360                        90$:    POP     R5          ;RESTORE R5
     101360 012605                                                      MOV     (SP)+,R5
5347 101362 004737 100672          JSR     PC,TRVACT           ;SINCE NUMERIC FOUND, GO TAKE ACTION
5348 101366 000137 100722          JMP     TRVNOB              ;GO POINT R3 TO NEXT NODE
5349
5350 101372                        100$:   PRINTF  #CLINBG     ;PRINT NUMBER TOO BIG ERROR
     101372 012746 052175                                               MOV     #CLINBG, (SP)
     101376 012746 000001                                               MOV     #1, (SP)
     101402 010600                                                      MOV     SP,R0
     101404 104417                                                      TRAP    C$PNTF
     101406 062706 000004                                               ADD     #4,SP
5351 101412 112737 177777 003221   MOVB    #-1,P$GDBD          ;SET ERROR RETURN FLAG
5352 101420                        110$:   POP     R5          ;RESTORE R5
     101420 012605                                                      MOV     (SP)+,R5
5353 101422 000137 100712          JMP     TRVBRC              ;TAKE "MISS" BRANCH
5354
5355
5356 101426 005001                 TRVALP: CLR     R1          ;CLEAR ALPHA FOUND FLAG
5357 101430 121427 000101          10$:    CMPB    (R4),#101   ;SEE IF CHAR. LESS THAN A "A"
5358 101434 002406                 BLT     20$                 ;BR IF YES (NOT ALPHA)
5359 101436 121427 000132          CMPB    (R4),#132           ;SEE IF CHAR. GREATER THAN A "Z"
5360 101442 003003                 BGT     20$                 ;BR IF YES (NOT ALPHA)
5361 101444 005204                 INC     R4                  ;UPDATE CMD LINE PTR TO NEXT CHAR
5362 101446 005201                 INC     R1                  ;INDICATE AN ALPHA WAS FOUND
5363 101450 000767                 BR      10$                 ;GO LOOK AT NEXT CHAR.
5364 101452 005701                 20$:    TST     R1          ;SEE IF ANY ALPHA'S WERE FOUND
5365 101454 001404                 BEQ     30$                 ;BR IF NO
5366 101456 004737 100672          JSR     PC,TRVACT           ;IF ANY FOUND TAKE ACTION
5367 101462 000137 100722          JMP     TRVNOB              ;THEN UPDATE R3 TO NEXT NODE  NO BRANCH
5368 101466 000137 100712          30$:    JMP     TRVBRC      ;NONE FOUND, TAKE MISS BRANCH
5369
5370 101472 005001                 TRVALN: CLR     R1          ;CLEAR ALPHANUM FOUND FLAG
5371 101474 121427 000060          10$:    CMPB    (R4),#60    ;SEE IF CHAR. LESS THAN A "0"
5372 101500 002417                 BLT     30$                 ;BR IF YES (NOT NUMERIC OR ALPHA)
5373 101502 121427 000072          CMPB    (R4),#72            ;SEE IF CHAR. GREATER THAN A "9"
5374 101506 003003                 BGT     20$                 ;BR IF YES (NOT NUMERIC)
5375 101510 005204                 INC     R4                  ;UPDATE CMD LINE PTR TO NEXT CHAR.
5376 101512 005201                 INC     R1                  ;INDICATE A NUMERIC FOUND
5377 101514 000767                 BR      10$                 ;GO LOOK AT NEXT CHAR.
5378 101516 121427 000101          20$:    CMPB    (R4),#101   ;SEE IF CHAR. LESS THAN A "A"
5379 101522 002406                 BLT     30$                 ;BR IF YES (NOT ALPHA)
5380 101524 121427 000132          CMPB    (R4),#132           ;SEE IF CHAR. GREATER THAN A "Z"
```

```
5381 101530 003003                      BGT     30$             ;BR IF YES (NOT ALPHA)
5382 101532 005204                      INC     R4              ;UPDATE CMD LINE PTR TO NEXT CHAR
5383 101534 005201                      INC     R1              ;INDICATE AN ALPHA FOUND
5384 101536 000756                      BR      10$             ;GO LOOK AT NEXT CHAR.
5385 101540 005701                30$:  TST     R1              ;SEE IF ANY ALPHANUM'S WERE FOUND
5386 101542 001404                      BEQ     40$             ;BR IF NO
5387 101544 004737 100672               JSR     PC,TRVACT       ;IF ANY FOUND TAKE ACTION
5388 101550 000137 100722               JMP     TRVNOB          ;THEN UPDATE R3 TO NEXT NODE -NO BRANCH
5389 101554 000137 100712         40$:  JMP     TRVBRC          ;NONE FOUND, TAKE MISS BRANCH
5390 101560                       TRVSTR: PUSH    R5              ;SAVE R5
     101560 010546                                                       MOV     R5,-(SP)
5391 101562 010401                      MOV     R4,R1           ;POINT R1 TO CMD STRING
5392 101564 010305                      MOV     R3,R5
5393 101566 062705 000006               ADD     #6,R5           ;POUNT R5 TO MATCH STRING FROM CLI NODE
5394 101572 005037 003212               CLR     P$CNT           ;CLEAR CHAR MATCH COUNT
5395 101576 105715                10$:  TSTB    (R5)            ;SEE IF END OF MATCH STRING YET
5396 101600 001411                      BEQ     20$             ;BR IF YES
5397 101602 105711                      TSTB    (R1)            ;SEE IF END OF CMD LINE YET
5398 101604 001407                      BEQ     20$             ;BR IF YES
5399 101606 121115                      CMPB    (R1),(R5)       ;SEE IF CHARACTERS MATCH
5400 101610 001005                      BNE     20$             ;BR IF NO
5401 101612 005237 003212               INC     P$CNT           ;MATCH -INCREMENT MATCH COUNT
5402 101616 005201                      INC     R1              ;UPDATE STRING POINTERS
5403 101620 005205                      INC     R5
5404 101622 000765                      BR      10$             ;BR TO CONTINUE CHECKING CHARS.
5405
5406 101624 005737 003212         20$:  TST     P$CNT           ;WHEN DONE SEE IF ANY MATCHES FOUND
5407 101630 001407                      BEQ     30$             ;BR IF NO, GO TAKE THE MISS BRANCH
5408 101632 010104                      MOV     R1,R4           ;POINT CMD POINTER TO END OF STRING &
5409 101634                             POP     R5              ;RESTORE R5
     101634 012605                                                      MOV     (SP)+,R5
5410 101636 004737 100672               JSR     PC,TRVACT       ;IF A MATCH FOUND, GO DO MATCH ACTION
5411 101642 066303 000004               ADD     4(R3),R3        ;UPDATE R3 TO NEXT NODE (NO BRANCH)
5412 101646 000207                      RTS     PC              ; (NO RETURN THRU TRVNOB SINCE DIFFERENT
5413                                                            ;  DISPLACEMENT DUE TO MATCH STRING)
5414 101650                       30$:  POP     R5              ;RESTORE R5
     101650 012605                                                      MOV     (SP)+,R5
5415 101652 000137 100712               JMP     TRVBRC          ; GO TAKE BRANCH
5416                                                            ; (PARSED OK),  1 IF ILL CMD.....
```

```
5418                                    .SBTTL  TRVADR   TRAVERSE COMMAND LINE INPUT ADDRESS
5419                            ;
5420                            ;       TRVADR                          TRAVERSE COMMAND LINE INPUT ADDRESS
5421                            ;
5422                            ;       THIS ROUTINE IS CALLED BY TWO DIFFERENT ACTION ROUTINES.  THE
5423                            ;       NODE ACTION ROUTINE CALLS IT TO PARSE THROUGH THE NODE
5424                            ;       ADDRESS INPUT BY THE OPERATOR.  THE OPRSEL ACTION ROUTINE
5425                            ;       CALLS TRVADR TO PARSE THROUGH THE "OPERATOR SELECTED" MESSAGE
5426                            ;       WHICH HAS BEEN INPUT IN THE COMMAND LINE.  FOR A NODE ADDRESS,
5427                            ;       THE ROUTINE LOOKS FOR A '/' AS A DELIMETER FOR THE ADDRESS,
5428                            ;       AND REPLACES THE / WITH A NULL BYTE FOR USE BY THE ADDRESS
5429                            ;       PACKING ROUTINE.  WHEN CALLED BY THE OPRSEL ROUTINE, A '"'
5430                            ;       IS EXPECTED AS THE DELIMETER FOR THE OPERATOR SELECTED MESSAGE.
5431                            ;       IF A NULL STRING IS ENTERED, AN ERROR MESSAGE IS PRINTED.
5432                            ;
5433                            ;       INPUTS -                R4 - POINTS TO THE BEGINING OF THE ADDRESS
5434                            ;                                    OR MESSAGE IN THE COMMAND LINE
5435                            ;       OUTPUTS -               SUMMARIZED IN TABLE BELOW
5436                            ;
5437                            ;       COMMAND LINE                            OUTPUTS
5438                            ;       INPUT CONDITION ! P$GDBD  ! R4 POINTS TO  !  CFLAG CONTAINS  ! P$MERR
5439                            ;       ----------------!---- -- --!-----------  !---  ---- -----  !  ----
5440                            ;       ILLEGAL CHAR.   !   -1     ! ILL. CHAR.  !                 ! N/A
5441                            ;       ADR./ASSIST     !   0      ! END OF LINE !     CASIST      ! N/A
5442                            ;       ADR./TARGET     !          !             !                 !
5443                            ;       ADR./           !   0      ! END OF LINE !     CTARGT      ! N/A
5444                            ;       ADR.            !          !             !                 !
5445                            ;       ADR./CHAR. OR   !          !             !                 !
5446                            ;       "OPR SEL/CHAR.  !          !             !                 !
5447                            ;         OTHER THAN "A"!   -1     ! /           !     CTARGT      ! N/A
5448                            ;         "T" OR BLANK  !          !             !                 !
5449                            ;       ""              !   0      ! CHAR. AFTER " !                 !  1
5450                            ;       "OPR SEL"       !   0      ! CHAR. AFTER " !     OPRSEL      !  0
5451                            ;
5452                            ;               CALLING PROCEDURE       JSR  PC,TRVADR
5453                            ;               REGISTER USAGE          R1 IS USED AS A COUNTER TO REPORT ERROR MESSAGES
5454                            ;                                          IF NULL STRINGS ARE ENTERED.
5455                            ;                                       R4 POINTS TO THE NEXT CHAR. IN THE COMMAND LINE
5456                            ;
5457                            ;---
5458 101656  005001            TRVADR: CLR     R1                      ;CLEAR HEX DIGIT FOUND FLAG
5459 101660  121427  000000     10$:    CMPB    (R4),#0                 ;SEE IF NUL CHAR.
5460 101664  001423                    BEQ     30$                     ; IF YES, RETURN
5461 101666  121427  000040            CMPB    (R4),#40                ;SEE IF ILLEGAL CHARACTER
5462 101672  002414                    BLT     20$                     ;IF YES; BRANCH TO ERROR ROUTINE
5463 101674  121427  000042            CMPB    (R4),#42                ;SEE IF CHAR. IS A ''
5464 101700  001454                    BEQ     70$                     ;IF YES ,BRANCH TO 70$
5465 101702  121427  000057            CMPB    (R4),#57                ;SEE IF CHAR. IS A "/"
5466 101706  001420                    BEQ     50$                     ;BRANCH IF YES
5467 101710  121427  000132            CMPB    (R4),#132               ;SEE IF CHAR. GREATER THAN 'F
5468 101714  003003                    BGT     20$                     ; IF YES, ILLEGAL CHAR.
5469 101716  005204                    INC     R4                      ;UPDATE CMD LINE POINTER TO NEXT CHAR.
5470 101720  005201                    INC     R1                      ;INCIDATE A VALID CHAR. FOUND
5471 101722  000756                    BR      10$                     ;LOOK AT NEXT CHAR.
5472 101724  112737  177777  003221    20$:    MOVB    #-1,P$GDBD              ;SET ERROR FLAG
5473 101732  000464                    BR      90$                     ;RETURN
5474 101734  005701                    30$:    TST     R1                      ;SEE IF VALID CHARACTERS FOUND
```

```
5475 101736 001772                          BEQ     20$                 ; IF NO, ILLEGAL CHAR.
5476 101740 012737 000000 003734 40$:       MOV     #CTARGT,CFLAG       ;SET TARGET FLAG
5477 101746 000456                          BR      90$                 ;RETURN
5478 101750 005701                  50$:    TST     R1                  ;SEE IF VALID CHARACTERS FOUND
5479 101752 001764                          BEQ     20$                 ; IF NO, ILLEGAL CHAR.
5480 101754 112714 000000                   MOVB    #0,(R4)             ; IF YES, REPLACE "/" WITH NULL CHAR.
5481 101760 005204                          INC     R4                  ;UPDATE CMD. LINE POINTER TO NEXT CHAR.
5482 101762 121427 000000                   CMPB    (R4),#0             ;IS NEXT CHAR. NULL
5483 101766 001764                          BEQ     40$                 ;  IF YES, TAKE DEFAULT OF TARGET
5484 101770 121427 000101                   CMPB    (R4),#'A            ;IS NEXT CHAR. "A"
5485 101774 001412                          BEQ     60$                 ;  IF YES, BR 60$
5486 101776 121427 000124                   CMPB    (R4),#'T            ;IS NEXT CHAR. 'T'
5487 102002 001756                          BEQ     40$                 ;  IF YES, SET TARGET FLAG
5488 102004 112737 177777 003221            MOVB    #-1,P$GDBD          ;  ELSE, SET ERROR FLAG,
5489 102012 005304                          DEC     R4                  ;  READJUST COMMAND LINE POINTER
5490 102014 112714 000057                   MOVB    #'/,(R4)            ;  AND REPLACE / IN CMD LINE TO FIX ERROR
5491 102020 000747                          BR      40$                 ;  SET TARGET FLAG AND RETURN
5492 102022 012737 000001 003734 60$:       MOV     #CASIST,CFLAG       ;SET ASSIST FLAG
5493 102030 000425                          BR      90$
5494 102032 005701                  70$:    TST     R1                  ;SEE IF ANY CHARACTERS TYPED
5495 102034 001407                          BEQ     80$                 ;IF NO, BRANCH TO 80$
5496 102036 112714 000000                   MOVB    #0,(R4)             ;ELSE, REPLACE '"' WITH NULL
5497 102042 012737 000006 003734            MOV     #OPRSEL,CFLAG       ;SET OPERATOR SELECTED FLAG
5498 102050 005204                          INC     R4
5499 102052 000414                          BR      90$                 ;RETURN
5500 102054                         80$:    PRINTF  #NULSTR             ;PRINT NULL STRING ERROR MESSAGE
     102054 012746 053076                                                        MOV     #NULSTR,-(SP)
     102060 012746 000001                                                       MOV     #1, (SP)
     102064 010600                                                              MOV     SP,R0
     102066 104417                                                              TRAP    C$PNTF
     102070 062706 000004                                                       ADD     #4,SP
5501 102074 112737 177777 003223            MOVB    #-1,P$MERR          ;SET OPER. SELECTED MSG. ERROR FLAG
5502 102102 005204                          INC     R4                  ;MOVE CMD. LINE POINTER TO NEXT CHAR.
5503 102104 000207                  90$:    RTS     PC                  ;RETURN
```

```
                                        .SBTTL          REPORT CODING SECTION


                                ;++
                                ; THE REPORT CODING SECTION CONTAINS THE
                                ; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
                                ;

                                        BGNRPT
                                                                                L$RPT::
                                ;*******************************************************************************
                                ;       THIS SECTION, WHICH IS OPTIONAL, CONTAINS THE CODE FOR PRINTING
                                ;       STATISTICAL INFORMATION GATHERED BY THE DIAGNOSTIC.  IT IS
                                ;       EXECUTED BY THE OPERATOR COMMAND "PRINT" OR BY THE MACRO CALL
                                ;       "DORPT".  USE THE PRINTS MACRO TO PRINT THE INFORMATION.
                                ;       USE FORMAT STATEMENTS AS IN THE PRINTB/PRINTX MACROS.  IT IS
                                ;       THE PROGRAMMER'S RESPONSIBILTY TO DEVISE AND IMPLEMENT THE
                                ;       FORM AND CONTENT OF THE STATISTICS.
                                ;*******************************************************************************

102106  004737  105170                  JSR     PC,ACTSUM
102112                                  EXIT    RPT
102112  000167                                                          .WORD   J$JMP
102114  000000                                                          .WORD   L10006-.-2
                                ;*******************************************************************************
                                ;     INSERT LOCAL STORAGE THAT IS USED ONLY
                                ;     DURING THE REPORT SECTION.
                                ;*******************************************************************************

                                ;*******************************************************************************
                                ;     INSERT MESSAGES THAT ARE USED ONLY
                                ;     DURING THE REPORT SECTION.
                                ;*******************************************************************************


                                        .EVEN

102116                                  ENDRPT
102116                                                          L10006:
102116  104425                                                          TRAP    C$RPT
```

```
                                        .SBTTL          PROTECTION TABLE
                                ;++
                                ; THIS TABLE IS USED BY THE RUNTIME SERVICES
                                ; TO PROTECT THE LOAD MEDIA.
                                ;
                                        BGNPROT
                                                                        L$PROT::
 5551 102120  177777                     1              ;OFFSET INTO P-TABLE FOR CSR ADDRESS
 5552 102122  177777                    -1              ;OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
 5553 102124  177777                    -1              ;OFFSET INTO P-TABLE FOR DRIVE NUMBER
 5554 102126                            ENDPROT
                                ;*******************************************************************************
                                ;       INSERT BYTE OFFSET FOR DATA NOTED IN COMMENTS ABOVE.  (OFFSET
                                ;       REFERS TO THE NUMBER OF BYTES FROM THE BEGINNING OF A PTABLE
                                ;       ENTRY TO THE ITEM IN QUESTION.)  IF THE PARTICULAR
                                ;       ITEM DOES NOT APPLY, LEAVE ENTRY AS -1.  WHEN THE RUNTIME
                                ;       SERVICES EXECUTES A GPHARD, IT USES THESE OFFSETS (IF NOT
                                ;       SET TO -1) TO GET THE ITEMS AND COMPARE WITH THOSE SAVED
                                ;       IN THE XXDP+ MONITOR.  IF THE UNIT BEING REQUESTED MATCHES THE
                                ;       LOAD DEVICE, THE RUNTIME SERVICES RETURN AN INCOMPLETE FLAG ON
                                ;       THE GPHARD.
                                ;*******************************************************************************
```

```
                                        .SBTTL          INITIALIZE SECTION
                                ;++
                                ; THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
                                ; AT THE BEGINNING OF EACH PASS.
                                ;
                                        BGNINIT
102126                                                                  L$INIT::
102126
                                ;
                                ;*****************************************************************************
                                ;       THE INITIALIZE CODE IS EXECUTED UNDER FIVE CONDITIONS.  THERE
                                ;       ARE SUPERVISOR EVENT FLAGS THAT ARE USED TO LET THE
                                ;       DIAGNOSTIC KNOW UNDER WHICH CONDITION THE EXECUTION IS TAKING
                                ;       PLACE.  THE EVENT FLAGS ARE READ USING THE "READEF" MACRO.
                                ;       THE CONDITIONS UNDER WHICH THE INIT CODE IS EXECUTED AND THE
                                ;       CORRRESPONDING EVENT FLAGS ARE:
                                ;               START COMMAND           EF.START
                                ;               RESTART COMMAND         EF.RESTART
                                ;               CONTINUE COMMAND        EF.CONTINUE
                                ;               POWERDOWN/POWERUP       EF.PWR
                                ;               NEW PASS                EF.NEW
                                ;       EXAMPLE OF EVENT FLAG USE:
                                ;               READEF  @EF.START
                                ;               BCOMPLETE       STARTCODE
                                ;       DURING THE INIT CODE, USE THE "GPHARD" MACRO TO OBTAIN P TABLE
                                ;       INFORMATION FOR DEVICE TESTING.  GET ONE UNIT'S INFORMATION IF
                                ;       THIS IS A SEQUENTIAL DIAGNOSTIC.  GET INFORMATION ON ALL
                                ;       UNITS AVAILABLE FOR TESTING IF THIS IS AN EXERCISER.  THE NUMBER
                                ;       OF UNITS AVAILABLE IS IN A HEADER LOCATION: "L$UNIT".
                                ;
                                ;*****************************************************************************
                                ;
                                ; ROUTINE STEPS
                                ;
                                ;       CHECK ENTRY FLAGS AND TAKE CORRECT ACTION
                                ;       INIT THE STACKS
                                ;       LOCATE FREE MEMORY
                                ;       VERIFY THAT A CLOCK OF SOME SORT EXISTS
                                ;       GET THE QNA ADDRESS FROM THE PTABLE
                                ;       SET UP THE QNA INTERRUPT VECTOR
                                ;       CLEAR NODE TABLE
                                ;       CLEAR SUMMARY TABLE
                                ;       CLEAR COUNTER DATA BASE
                                ;       SET UP FOR CLOCK INTERRUPTS
                                ;       CALL THE QNA INIT ROUTINE
                                ;       GET THE DEFAULT PHYSICAL ADDRESS AND PRINT IT
                                ;       RESET THE CLOCK
                                ;       EXIT
                                ;
102126                          INIT:
102126                                  PUSH    R2,R3,R4                ; SAVE THE REGISTERS
102126  010446                                                                  MOV     R4, (SP)
102130  010346                                                                  MOV     R3, (SP)
102132  010246                                                                  MOV     R2, (SP)
102134  022737  000020  003734          CMP     #CEXIT,CFLAG            ;SEE IF EXIT COMMAND TYPED
102142  001004                          BNE     INIT1                   ; IF NO, DO INIT CODE
102144  005037  003734                  CLR     CFLAG                   ; ELSE, CLEAR EXIT FLAG
102150  000137  103344                  JMP     INICLN                  ; AND DO CLEANUP
```

INITIALIZE SECTION

```
5624 102154                                INIT1:
5625 102154                                        READEF  @EF.START               ;IF HERE BECAUSE OF "START', DO INIT
     102154  012700  000040                                                                MOV     @EF.START,R0
     102160  104447                                                                       TRAP    C$REFG
5626 102162                                        BCOMPLETE       START
     102162  103424                                                                       BCS     START
5627 102164                                        READEF  @EF.RESTART             ;IF HERE BECAUSE OF "RESTART', DO SOME INIT
     102164  012700  000037                                                                MOV     @EF.RESTART,R0
     102170  104447                                                                       TRAP    C$REFG
5628 102172                                        BNCOMPLETE      10$
     102172  103002                                                                       BCC     10$
5629 102174  000137  103234                       JMP     RESTRT
5630 102200                                10$:
5631 102200                                        READEF  @EF.CONTINUE            ;IF HERE BECAUSE OF "CONTINUE", EXIT
     102200  012700  000036                                                                MOV     @EF.CONTINUE,R0
     102204  104447                                                                       TRAP    C$REFG
5632 102206                                        BNCOMPLETE      20$
     102206  103002                                                                       BCC     20$
5633 102210  000137  103234                       JMP     RESTRT
5634 102214                                20$:
5635 102214                                        READEF  @EF.NEW                 ;IF HERE ON NEW PASS, SKIP SOME INIT
     102214  012700  000035                                                                MOV     @EF.NEW,R0
     102220  104447                                                                       TRAP    C$REFG
5636 102222                                        BNCOMPLETE      30$
     102222  103002                                                                       BCC     30$
5637 102224  000137  103334                       JMP     NEW
5638 102230                                30$:
5639 102230  000137  103354                       JMP     INIEXI                  ;IF DON'T KNOW WHY WE'RE HERE, EXIT
5640 102234                                START:
5641 102234                                        I$STACK @1000,SP                ;SET PARAMETER STACK POINTER
     102234  012705  001000                        MOV     @1000,R5        ;INITIALIZE THE PARAMETER STACK POINTER.
     102240  010606                                MOV     SP,SP   ;INITIALIZE THE HARDWARE STACK POINTER.
5642 102242                                        MEMORY  FRESIZ                  ;GET FREE MEMORY INFO
     102242  104431                                                                       TRAP    C$MEM
     102244  010037  050022                                                                MOV     R0,FRESIZ
5643 102250  013737  050022  050024               MOV     FRESIZ,FREMEM           ;SIZE OF FREE MEMORY IN FRESIZ
5644 102256  062737  000002  050024               ADD     @2,FREMEM               ;START OF FREE MEMORY IN FREMEM
5645 102264  012702  003736                       MOV     @CLKCSR,R2              ;SETUP R2 AS A PRT. TO CLOCK INFO. BLOCK
5646 102270                                        CLOCK   L,R1                    ;GET LINE CLOCK INFO
     102270  012700  000114                                                                MOV     @'L,R0
     102274  104462                                                                       TRAP    C$CLCK
     102276  010001                                                                       MOV     R0,R1
5647 102300                                        BNCOMPLETE      10$             ;IF NONE, SEE IF P CLOCK PRESENT
     102300  103006                                                                       BCC     10$
5648 102302  004737  070504                       JSR     PC,CLKSET               ;SET UP CLOCK INFO TABLE AND VECTOR
5649 102306  012737  000100  003746               MOV     @LCLKEN,CLKEN           ;SET UP THE ENABLE LINE CLOCK DATA
5650 102314  000514                               BR      40$
5651 102316                                10$:
5652 102316                                        CLOCK   P,R1                    ;GET P CLOCK INFO
     102316  012700  000120                                                                MOV     @'P,R0
     102322  104462                                                                       TRAP    C$CLCK
     102324  010001                                                                       MOV     R0,R1
5653 102326                                        BNCOMPLETE      20$             ;IF NO CLOCK, ERROR
     102326  103017                                                                       BCC     20$
5654 102330  004737  070504                       JSR     PC,CLKSET               ; ELSE SET UP CLOCK INFO AND VECTOR
5655 102334  062737  000002  003736               ADD     @2,CLKCSR               ; POINT CLKCSR TO P CLK COUNT SET REG.
5656 102342  012777  001600  101366               MOV     @PCLKCT,@CLKCSR         ;LOAD CLK SET REG. WITH COUNT VALUE
```

```
5657 102350  162737  000002  003736          SUB     #2,CLKCSR               ;POINT CLKCSR BACK TO P-CLK CSR
5658 102356  012737  000111  003746          MOV     #PCLKEN,CLKEN           ;SETUP TO ENABLE P-CLK DATA
5659 102364  000470                          BR      40$
5660 102366                          20$:
5661 102366  012737  000100  003742          MOV     #100,CLKVEC             ; SET UP INTERRUPT VECTOR
5662 102374  012737  003746  003736          MOV     #CLKEN,CLKCSR           ; FAKE A CSR LOCATION
5663 102402  012737  000001  003756          MOV     #1,TIMER1               ; SET TIMER 1 VALUE
5664 102410                                  SETVEC  CLKVEC,#CLKINT,#0       ; ENABLE CLOCK INTERRUPTS
     102410  012746  000000                                                          MOV     #0,-(SP)
     102414  012746  070530                                                          MOV     #CLKINT,-(SP)
     102420  013746  003742                                                          MOV     CLKVEC,-(SP)
     102424  012746  000003                                                          MOV     #3,-(SP)
     102430  104437                                                                  TRAP    C$SVEC
     102432  062706  000010                                                          ADD     #10,SP
5665 102436                                  DELAY   50                      ; WAIT SOME TIME FOR CLOCK INTERRUPTS
     102436  012727  000050                                                          MOV     #50,(PC)+
     102442  000000                                                                  .WORD   0
     102444  013727  002116                                                          MOV     L$DLY,(PC)+
     102450  000000                                                                  .WORD   0
     102452  005367  177772                                                          DEC     6(PC)
     102456  001375                                                                  BNE     .-4
     102460  005367  177756                                                          DEC     -22(PC)
     102464  001367                                                                  BNE     .-20
5666 102466  005737  003756          TST     TIMER1                  ; DID TIMER1 DECREMENT TO ZERO?
5667 102472  001013                  BNE     30$                     ; NO, NO CLOCK EXISTS. REPORT ERROR
5668 102474                          GMANID  L5060,CLKHZ,D,377,50.,60.,YES ; ASK IF 50 OR 60 HZ
     102474  104443                                                          TRAP    C$GMAN
     102476  000406                                                          BR      10000$
     102500  003744                                                          .WORD   CLKHZ
     102502  000052                                                          .WORD   T$CODE
     102504  052242                                                          .WORD   L5060
     102506  000377                                                          .WORD   377
     102510  000062                                                          .WORD   T$LOLIM
     102512  000074                                                          .WORD   T$HILIM
     102514                                                          10000$:
5669 102514  004737  070504                  JSR     PC,CLKSET
5670 102520  000412                          BR      40$                     ; CONTINUE
5671 102522                          30$:
5672 102522                          PRINTF  #NOCLK                  ;ERROR MESSAGE
     102522  012746  061766                                                  MOV     #NOCLK,-(SP)
     102526  012746  000001                                                  MOV     #1,-(SP)
     102532  010600                                                          MOV     SP,R0
     102534  104417                                                          TRAP    C$PNTF
     102536  062706  000004                                                  ADD     #4,SP
5673 102542  000137  103344          JMP     INICLN                  ;CANNOT CONTINUE, DO CLEANUP
5674 102546                          40$:
5675 102546                          GPHARD  #0,R1                   ;GET P-TAB POINTER FOR THIS UNIT
     102546  012700  000000                                                  MOV     #0,R0
     102552  104442                                                          TRAP    C$GPHRD
     102554  010001                                                          MOV     R0,R1
5676 102556                          BCOMPLETE       50$             ;THIS ONE IS NOT AVAILABLE
     102556  103402                                                          BCS     50$
5677 102560  000137  103344          JMP     INICLN
5678 102564                          50$:
5679 102564  012137  047772          MOV     (R1)+,QNAADO            ;SAVE DEVICE ADDRESS
5680 102570  012137  047774          MOV     (R1)+,QNAVCO            ;SAVE VECTOR
5681 102574  012137  047776          MOV     (R1)+,QNAPRO            ;SAVE PRIORITY
```

INITIALIZE SECTION

```
5682 102600                                    SETVEC  QNAVCO,#QNAISR,QNAPRO   ;SETUP QNA INTERRUPT VECTOR
     102600  013746  047776                                                        MOV     QNAPRO,-(SP)
     102604  012746  073064                                                        MOV     #QNAISR,-(SP)
     102610  013746  047774                                                        MOV     QNAVCO,-(SP)
     102614  012746  000003                                                        MOV     #3,-(SP)
     102620  104437                                                               TRAP    C$SVEC
     102622  062706  000010                                                       ADD     #10,SP
5683 102626  012703  000050                    MOV     #TBLLEN,R3                ;CLEAR NODE TABLE
5684 102632  012702  002412                    MOV     #NODTBL,R2
5685 102636                            60$:
5686 102636  005022                          CLR     (R2)+
5687 102640  005303                          DEC     R3
5688 102642  001375                          BNE     60$
5689 102644  012703  000132                  MOV     #STBLEN,R3                ;CLEAR SUMMARY TABLE
5690 102650  012702  002714                  MOV     #STATBL,R2
5691 102654                            70$:
5692 102654  005022                          CLR     (R2)+
5693 102656  005303                          DEC     R3
5694 102660  001375                          BNE     70$
5695 102662  005037  050266                  CLR     BCOUNT                    ; CLEAR UNEXPLAINED ERROR COUNTER
5696 102666  005037  050270                  CLR     ERRFLG                    ; CLEAR ERROR FLAG
5697 102672  005037  050252                  CLR     FATFLG                    ; CLEAR FATAL FLAG ERROR
5698 102676  005037  050032                  CLR     S.NREC                    ; CLEAR SUMMARY DATA COUNTERS
5699 102702  005037  050030                  CLR     S.REC
5700 102706  005037  050034                  CLR     S.LEN
5701 102712  005037  050036                  CLR     S.COMP
5702 102716  005037  050040                  CLR     S.BYTE
5703 102722  005037  050042                  CLR     S.XFER
5704 102726  012702  050144                  MOV     #CNTRS,R2                 ; GET BASE ADDRESS OF COUNTER DATA BASE
5705 102732  012203                          MOV     (R2)+,R3                  ; GET SIZE OF COUNT DATA BASE
5706 102734  006203                          ASR     R3                        ; TURN BYTE COUNT INTO WORD COUNT
5707 102736                            80$:
5708 102736  005022                          CLR     (R2)+                     ; CLEAR THIS COUNT
5709 102740  005303                          DEC     R3                        ; KNOCK DOWN COUNT BY ONE
5710 102742  001375                          BNE     80$                       ; BRANCH IF MORE TO DO
5711 102744  005037  003750                  CLR     TIMMIN                    ;CLEAR TIME SINCE-START-LOCATIONS
5712 102750  005037  003752                  CLR     TIMSEC
5713 102754  013737  003744  003754          MOV     CLKHZ,TIMTCK              ; LOAD TICKS/SEC
5714 102762                                    SETVEC  CLKVEC,#CLKINT,CLKBR      ;SETUP CLOCK INTERRUPT VECTOR
     102762  013746  003740                                                        MOV     CLKBR,-(SP)
     102766  012746  070530                                                        MOV     #CLKINT,-(SP)
     102772  013746  003742                                                        MOV     CLKVEC,-(SP)
     102776  012746  000003                                                        MOV     #3,-(SP)
     103002  104437                                                               TRAP    C$SVEC
     103004  062706  000010                                                       ADD     #10,SP
5715 103010  013777  003746  100720          MOV     CLKEN,@CLKCSR             ;SET ENABLE BITS IN THE CLOCK TO START
5716 103016                                    SETPRI  #PRI00                    ;SET PRIORITY=0 SO CLOCK CAN INTERRUPT
     103016  012700  000000                                                        MOV     #PRI00,R0
     103022  104441                                                               TRAP    C$SPRI
5717 103024                                    CALL    QNAINI                    ;INITIALIZE THE QNA
     103024  004437  070656                                                        JSR     R4,PREG14
     103030  000220                                                                .WORD   QNAINI-ANCHOR
5718 103032                                    P$POP   R0                        ; GET RETURN STATUS
     103032  014500                                                                MOV     -(R5),R0
5719 103034  001401                          BEQ     90$                       ; BRANCH IF NO ERRORS
5720 103036  000542                          BR      INICLN
5721 103040                            90$:
```

INITIALIZE SECTION

```
5722 103040 012702 004010                    MOV     @PHYADR,R2              ; GET CSR ADDRESS
5723 103044 012703 003774                    MOV     @DEPADR,R3              ; GET DEFAULT PHYSICAL ADDRESS
5724 103050 012223                           MOV     (R2)+,(R3)+             ; GET FIRST WORD OF THE ADDRESS
5725 103052 012223                           MOV     (R2)+,(R3)+             ; GET SECOND WORD OF THE ADDRESS
5726 103054 011213                           MOV     (R2),(R3)               ; GET FINAL WORD OF THE ADDRESS
5727 103056 012703 047674                    MOV     #IDTDAT,R3              ; POINT TO SYSTEM ID MESSAGE
5728 103062 012702 004010                    MOV     @PHYADR,R2              ; POINT TO PHYSICAL ADDRESS
5729 103066 011263 000006                    MOV     (R2),SOURCC(R3)         ; OUR ADDRESS IS CONSOLE ID SOURCE
5730 103072 016263 000002 000010             MOV     2(R2),SOURCC+2(R3)      ; AGAIN
5731 103100 016263 000004 000012             MOV     4(R2),SOURCC+4(R3)      ; AND AGAIN
5732 103106 011263 000042                    MOV     (R2),SIADDR(R3)         ; OUR ADDRESS IS CONSOLE ID DEVICE ADDRESS
5733 103112 016263 000002 000044             MOV     2(R2),SIADDR+2(R3)      ; AGAIN
5734 103120 016263 000004 000046             MOV     4(R2),SIADDR+4(R3)      ; AND AGAIN
5735 103126                          100$:
5736 103126                                  CALL    BINHEX @DEPADR,#6,@STRBUF ; PUT ADDRESS INTO HEX FORMAT
     103126 012725 002324                                                        MOV     @STRBUF,(R5)+
     103132 012725 000006                                                        MOV     #6,(R5)+
     103136 012725 003774                                                        MOV     @DEPADR,(R5)+
     103142 004437 070656                                                        JSR     R4,PREG14
     103146 005150                                                               .WORD   BINHEX ANCHOR
5737 103150                                  PRINTS  #HDMSG1,@STRBUF         ; PRINT ADDRESS
     103150 012746 002324                                                        MOV     @STRBUF,-(SP)
     103154 012746 055046                                                        MOV     #HDMSG1,-(SP)
     103160 012746 000002                                                        MOV     #2,-(SP)
     103164 010600                                                               MOV     SP,R0
     103166 104416                                                               TRAP    C$PNTS
     103170 062706 000006                                                        ADD     #6,SP
5738 103174 012737 000000 002372             MOV     #ALPHA,P$TYPE           ;SET MESSAGE DEFAULT VALUES
5739 103202 012737 001000 002374             MOV     #512.,P$SIZE
5740 103210 012737 000001 002376             MOV     #1,P$CPYS
5741 103216 005037 003202                    CLR     RSPFLG                  ; CLEAR RESPONDER MODE FLAG
5742 103222 005037 003750                    CLR     TIMMIN                  ; CLEAR TIME SINCE START LOCATIONS
5743 103226 005037 050146                    CLR     CNTRS+C.SECS            ; CLEAR QNA TIME SINCE START
5744 103232 000450                           BR      INIEXI
5745 103234                          RESTRT:
5746 103234 005037 002200                    CLR     BLDFLG                  ; INDICATE THAT WE ARE NO LONGER BUILDING
5747 103240 005037 003202                    CLR     RSPFLG                  ; CLEAR RESPONDER MODE FLAG
5748 103244 013737 003744 003754             MOV     CLKHZ,TIMTCK            ;LOAD TICKS/SEC
5749 103252                                  SETVEC  CLKVEC,#CLKINT,CLKBR    ;SETUP CLOCK INTERRUPT VECTOR
     103252 013746 003740                                                        MOV     CLKBR,-(SP)
     103256 012746 070530                                                        MOV     #CLKINT,-(SP)
     103262 013746 003742                                                        MOV     CLKVEC,-(SP)
     103266 012746 000003                                                        MOV     #3,-(SP)
     103272 104437                                                               TRAP    C$SVEC
     103274 062706 000010                                                        ADD     #10,SP
5750 103300 013777 003746 100430             MOV     CLKEN,@CLKCSR           ;SET ENABLE BITS IN THE CLOCK TO START
5751 103306                                  SETPRI  #PRI00                  ;SET PRIORITY=0 SO CLOCK CAN INTERRUPT
     103306 012700 000000                                                        MOV     #PRI00,R0
     103312 104441                                                               TRAP    C$SPRI
5752 103314                                  CALL    QNAINI                  ;INITIALIZE THE QNA
     103314 004437 070656                                                        JSR     R4,PREG14
     103320 000220                                                               .WORD   QNAINI ANCHOR
5753 103322                                  P$POP   R0                      ; GET RETURN STATUS
     103322 014500                                                               MOV     (R5)+,R0
5754 103324 001401                           BEQ     10$                     ; BRANCH IF NO ERRORS
5755 103326 000406                           BR      INICLN
5756 103330                          10$:
```

```
                                        CLR     TIMSEC
                                NEW:
                                        MOV     CLKEN,@CLKCSR           ;SET ENABLE BITS IN THE CLOCK TO START
                                        BR      INIEXI                  ;EXIT
                                INICLN:
                                        POP     R2,R3,R4                ; RESTORE THE REGISTERS
                                                                                MOV     (SP)+,R2
                                                                                MOV     (SP)+,R3
                                                                                MOV     (SP)+,R4
                                        DOCLN                           ; ABORT PASS
                                                                                TRAP    C$DCLN
                                INIEXI:
                                        POP     R2,R3,R4                ; RESTORE THE REGISTERS
                                                                                MOV     (SP)+,R2
                                                                                MOV     (SP)+,R3
                                                                                MOV     (SP)+,R4
                                        EXIT    INIT                    ; EXIT INIT SECTION
                                                                                TRAP    C$EXIT
                                                                                .WORD   L10010-.

                                ;*******************************************************************************
                                ;       INSERT LOCAL STORAGE THAT IS USED ONLY
                                ;       DURING THE INITIALIZE SECTION.
                                ;*******************************************************************************

                                ;*******************************************************************************
                                ;       INSERT MESSAGES THAT ARE USED ONLY
                                ;       DURING THE INITIALIZE SECTION.
                                ;*******************************************************************************

                                        .EVEN

                                ENDINIT

                                L10010:
                                                TRAP    C$INIT
```

```
                                .SBTTL  AUTODROP SECTION

                        ;++
                        ; THIS CODE IS EXECUTED IMMEDIATELY AFTER THE INITIALIZE CODE IF
                        ; THE "ADR" FLAG WAS SET.  THE UNIT(S) UNDER TEST ARE CHECKED TO
                        ; SEE IF THEY WILL RESPOND.  THOSE THAT DON'T ARE IMMEDIATELY
                        ; DROPPED FROM TESTING.
                        ;

                                BGNAUTO
103370                                                          L$AUTO::
103370
                        ;*******************************************************************************
                        ;       INSERT CODE HERE TO CHECK DEVICE(S) TO SEE IF THEY RESPOND.
                        ;       ISSUE A "DODU" FOR THOSE THAT DON'T.
                        ;*******************************************************************************

                                ENDAUTO
103370                                                          L10011:
103370
103370  104461                                                          TRAP    C$AUTC
```

CLEANUP CODING SECTION

```
                                        .SBTTL  CLEANUP CODING SECTION

                                ;++
                                ; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFOR.D
                                ; AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.
                                ;
                                ; ROUTINE STEPS:
                                ;
                                ;       GET CSR ADDRESS
                                ;       CLEAN UP ANY RECEIVE BUFFERS HANGING AROUND
                                ;       CLEAR INTERRUPT AND RECEIVE ENABLE BITS
                                ;       DISABLE CLOCKS
                                ;       EXIT
                                ;
                                ;--

103372                          BGNCLN
103372                                                                  L$CLEAN::
                                ;*******************************************************************************
                                ;       INSERT YOUR CLEANUP CODING.  THIS CODING SHOULD
                                ;       RESTORE YOUR TEST-DEVICE TO A NEUTRAL STATE.
                                ;       THIS CODE WILL BE EXECUTED AFTER EACH PASS AND AFTER THE
                                ;       PROGRAM IS INTERRUPTED BY "†C".
                                ;*******************************************************************************
103372                          P$PUSH  R2                      ; PUT R2 ON THE STACK
103372  010225                                                          MOV     R2,(R5)+
                                10$:
103374                          SETPRI  PRI00                   ; MAKE SURE CLOCKS CAN INTERRUPT
103374  013700  000000                                                  MOV     PRI00,R0
103400  104441                                                          TRAP    C$SPRI
103402  005737  050256          TST     NIRCNT                  ; SEE IF RECEVE COUNT ZERO
103406  001417                  BEQ     20$                     ; IF YES, EXIT
103410                          CALL    RECEVE                  ; ELSE, CLEAR OUT UNWANTED ENTRIES
103410  004437  070656                                                  JSR     R4,PREG14
103414  003044                                                          .WORD   RECEVE ANCHOR
103416                          P$POP   R2
103416  014502                                                          MOV     -(R5) R2
103420  001765                  BEQ     10$                     ; IF ANY WERE FOUND, UPDATE NEXT POINTER
103422  013702  024226          MOV     RRGNXT,R2               ; GET THE DESCRIPTOR
103426                          CALL    GETRNX  #1,@RRGNXT      ; UPDATE NEXT AND PREVIOUS DESCRIPTOR POINTERS
103426  012725  024226                                                  MOV     @RRGNXT,(R5)+
103432  012725  000001                                                  MOV     #1,(R5)+
103436  004437  070656                                                  JSR     R4,PREG14
103442  006450                                                          .WORD   GETRNX ANCHOR
103444  000753                  BR      10$                     ;SEE IF ANY MORE ENTRIES
                                20$:
103446  013702  047772          MOV     QNAADO,R2               ; GET DEVICE ADDRESS
103452  012737  000001  003756  MOV     #1,TIMER1               ; TIMER FOR INITITIALIZE
103460  012762  000002  000016  MOV     #MRESET,CSR(R2)         ; RESET THE DEVICE
103466  012737  000001  003756  MOV     #1,TIMER1               ; TIMER FOR INITITIALIZE
                                30$:
103474  005737  003756          TST     TIMER1                  ; INIT DONE?
103500  001375                  BNE     30$                     ; BRANCH IF NOT
103502  005062  000016          CLR     CSR(R2)                 ; FINISH THE RESET
103506  005077  100224          CLR     @CLKCSR                 ; DISABLE CLOCK
103512                          SETPRI  #PRI07                  ; SET PROCESSOR PRIORITY BACK TO
```

```
CLEANUP CODING SECTION

        103512  012700  000340                                                  MOV     #PRIO7,R0
        103516  104441                                                          TRAP    C$SPRI
  5852  103520                          P$POP   R2                      ; REMOVE R2 FROM STACK
        103520  014502                                                          MOV     -(R5),R2
  5853  103522                          EXIT    CLN
        103522  104432                                                          TRAP    C$EXIT
        103524  000002                                                          .WORD   L10012-.
  5854
  5856                          ;*************************************************************************
  5857                          ;       INSERT LOCAL STORAGE THAT IS USED ONLY
  5858                          ;       DURING THE CLEANUP SECTION.
  5859                          ;*************************************************************************
  5860
  5861                          ;*************************************************************************
  5862                          ;       INSERT MESSAGES THAT ARE USED ONLY
  5863                          ;       DURING THE CLEANUP SECTION.
  5864                          ;*************************************************************************
  5866
  5867                                  .EVEN
  5868
  5869  103526                          ENDCLN
        103526                                                          L10012:
        103526  104412                                                          TRAP    C$CLEAN
```

```
                        .SBTTL  DROP UNIT SECTION

                        ;++
                        ; THE DROP UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
                        ; TO NO LONGER BE TESTED.
                        ;-

103530                          BGNDU
103530                                                          L$DU::

                        ;*****************************************************************************
                        ;       INSERT DROP CODE HERE.  THIS CODE WILL BE EXECUTED AFTER
                        ;       A "DROP' COMMAND OR A "DODU" MACRO EXECUTION.  THE PURPOSE
                        ;       OF THIS CODE IS TO DO ANY NECESSARY HOUSEKEEPING AFTER A
                        ;       UNIT HAS BEEN DROPPED.  THIS SECTION IS OPTIONAL.
                        ;*****************************************************************************

103530                          EXIT    DU
103530  000167                                                  .WORD   J$JMP
103532  000000                                                  .WORD   L10013-2 .

                        ;*****************************************************************************
                        ;     INSERT LOCAL STORAGE THAT IS USED ONLY
                        ;     DURING THE DROP-UNIT SECTION.
                        ;*****************************************************************************

                        ;*****************************************************************************
                        ;     INSERT MESSAGES THAT ARE USED ONLY
                        ;     DURING THE DROP-UNIT SECTION.
                        ;*****************************************************************************

                        .EVEN

103534                  ENDDU
103534                                                  L10013:
103534  104453                                                  TRAP    C$DU
```

```
                                .SBTTL  ADD UNIT SECTION

                                ;++
                                ; THE ADD-UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
                                ;  TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
                                ;  TO THE TEST CYCLE.
                                ;-

103536                                  BGNAU
103536                                                                  L$AU::

                                ;*****************************************************************************
                                ;       INSERT ADD CODE HERE.  THIS CODE WILL BE EXECUTED AFTER
                                ;       AN "ADD" COMMAND.  THE PURPOSE OF THIS CODE IS TO DO ANY
                                ;       HOUSEKEEPING THAT MAY BE NECESSARY AFTER A UNIT HAS BEEN ADDED.
                                ;       THIS SECTION IS OPTIONAL.
                                ;*****************************************************************************

103536                                  EXIT    AU
103536  000167                                                          .WORD   J$JMP
103540  000000                                                          .WORD   L10014-2 .

                                ;*****************************************************************************
                                ;     INSERT LOCAL STORAGE THAT IS USED ONLY
                                ;     DURING THE ADD-UNIT SECTION.
                                ;*****************************************************************************

                                ;*****************************************************************************
                                ;     INSERT MESSAGES THAT ARE USED ONLY
                                ;     DURING THE ADD-UNIT SECTION.
                                ;*****************************************************************************

                                .EVEN

103542                          ENDAU
103542                                                          L10014:
103542  104452                                                          TRAP    C$AU
```

```
                                .SBTTL          TEST 1:

                        ;*************************************************************************
                        ;       APPEND THE NAME OF THIS TEST TO THE .SBTTL, AS SHOWN IN THE
                        ;       FOLLOWING EXAMPLE.      .SBTTL  TEST 1:  NAME OF TEST
                        ;*************************************************************************


                        ;++
                        ; TEST TO ...
                        ;-

                        ;*************************************************************************
                        ;       CHANGE THE PHRASE "TEST TO ..." TO BE A FUNCTIONAL
                        ;       DESCRIPTION OF THE HARDWARE TEST WHICH FOLLOWS.
                        ;*************************************************************************

                        ;*************************************************************************
                        ;       INSERT PROGRAM EQUATES THAT ARE USED ONLY IN THIS TEST.
                        ;*************************************************************************

103544                          BGNTST
103544                                                          T1::

                        ;*************************************************************************
                        ;       INSERT THE CODING FOR THIS HARDWARE TEST.
                        ;*************************************************************************
```

```
                                        .SBTTL          GETCL   COMMAND LINE FETCH & INTERPRETATION SECTION
                                ;**
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ;       THIS PART OF TEST 1 IS USED TO PROMPT THE USER FOR A COMMAND
                                ;
                                ; CALLING SEQUENCE
                                ;
                                ;       ENTERED VIA DSRS TEST DISPATCH ROUTINE
                                ;
                                ; INPUT
                                ;
                                ;       NONE
                                ;
                                ; IMPLICIT INPUT
                                ;
                                ;       NONE
                                ;
                                ; OUTPUT
                                ;
                                ;       A COMMAND FROM THE USER
                                ;
                                ; IMPLICIT OUTPUT
                                ;
                                ;       NONE
                                ;
                                ; COMPLETION CODES
                                ;
                                ;       NONE
                                ;
                                ; SIDE EFFECTS
                                ;
                                ;       NONE
                                ;
                                ; REGISTERS USED
                                ;
                                ;       NONE
                                ;
                                ; DEBUG
                                ;
                                ;       NONE
                                ;-
103544                          GETCL:
103544  105037  003221                  CLRB    P$GDBD                  ;CLEAR CMD LINE PARSING ERROR FLAG
103550  105037  003220                  CLRB    P$NNUF
103554                          GMANID  CLI$PM,CMDBUF,A,0,1,72.,NO      ;GET CMD LINE FROM OPERATOR
103554  104443                                                                  TRAP    C$GMAN
103556  000406                                                                  BR      10000$
103560  002202                                                                  .WORD   CMDBUF
103562  000142                                                                  .WORD   T$CODE
103564  052104                                                                  .WORD   CLI$PM
103566  000000                                                                  .WORD   0
103570  000001                                                                  .WORD   T$LOLIM
103572  000110                                                                  .WORD   T$HILIM
103574                                                                  10000$:
103574  012737  002202  003204          MOV     #CMDBUF,P$BUFA
103602  012737  050516  003206          MOV     #CLITRE,P$TREE
```

```
 6027 103610  012737  103726  003210          MOV     #CLIACT,P$ACT
 6028 103616  005037  003734                  CLR     CFLAG           ;CLEAR QUALIFIER FLAG
 6029 103622  004737  100544                  JSR     PC,P$TRV        ;GO PARSE COMMAND TREE
 6030 103626  105737  003221                  TSTB    P$GDBD          ;SEE IF PARSED OK, OR AN ERROR
 6031 103632  001412                          BEQ     10$
 6032 103634                                  PRINTF  #CLIERM         ;IF NOT PRINT ERROR MESSAGE
      103634  012746  052113                                              MOV     #CLIERM, (SP)
      103640  012746  000001                                              MOV     #1, (SP)
      103644  010600                                                      MOV     SP,R0
      103646  104417                                                      TRAP    C$PNTF
      103650  062706  000004                                              ADD     #4,SP
 6033 103654  000137  103544                  JMP     GETCL
 6034 103660  105737  003220          10$:    TSTB    P$NNUF          ;SEE IF INCOMPLETE COMMAND TYPED
 6035 103664  001412                          BEQ 20$
 6036 103666                                  PRINTF  #CLINUF         ;IF NOT PRINT ERROR MESSAGE
      103666  012746  052144                                              MOV     #CLINUF, (SP)
      103672  012746  000001                                              MOV     #1, (SP)
      103676  010600                                                      MOV     SP,R0
      103700  104417                                                      TRAP    C$PNTF
      103702  062706  000004                                              ADD     #4,SP
 6037 103706  000137  103544                  JMP     GETCL
 6038 103712  022737  000020  003734  20$:    CMP     #CEXIT,CFLAG    ;WAS EXIT COMMAND TYPED?
 6039 103720  001311                          BNE     GETCL           ;  IF NOT GET NEW COMMAND LINE
 6040 103722                                  EXIT    TST             ;  ELSE EXIT
      103722  104432                                                      TRAP    C$EXIT
      103724  013164                                                      .WORD   L10015 .
```

```
                                        .SBTTL          CLI ACTION TABLE AND ROUTINES
                                ;**
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ;       THESE ARE THE ACTION TABLES TO DEFING WHAT TO DO WITH A COMMAND
                                ;       RECEIVED FROM A USER
                                ;       USER MUST CLEAR/SET P$GDBD IF USE "CLIBIF" IN CONNECTION WITH ACTION
                                ;
                                ;
                                ; CALLING SEQUENCE
                                ;
                                ;       JMP     CL
                                ;
                                ; INPUT
                                ;
                                ;       COMMAND FROM THE USER
                                ;
                                ; IMPLICIT INPUT
                                ;
                                ;       NONE
                                ;
                                ; OUTPUT
                                ;
                                ;       NONE
                                ;
                                ; IMPLICIT OUTPUT
                                ;
                                ;       NONE
                                ;
                                ; COMPLETION CODES
                                ;
                                ;       NONE
                                ;
                                ; SIDE EFFECTS
                                ;
                                ;       NONE
                                ;
                                ; REGISTERS USED
                                ;
                                ;       R2 - HOLDS ACTION CODE FROM THE PARSING (CLI) NODE
                                ;
                                ; DEBUG
                                ;
                                ;       NONE
                                ;--
103726                          CLIACT:
103726  006302                          ASL     R2              ;MULTIPLY ACTION CODE BY 2
103730  016202  103744                  MOV     10$(R2),R2      ;OFFSET VALUE
103734  062702  103744                  ADD     #10$,R2         ;ADD BASE VALUE
103740  004712                          JSR     PC,(R2)         ;GO DO ACTION
103742  000207                          RTS     PC              ;RETURN TO TRVACT
                                                                ;BRIEF DESCRIPTION OF ACTION TAKEN
103744  000124                  10$:    .WORD   ACTNUL-10$      ;0-NULL
103746  000126                          .WORD   ACTHLP-10$      ;1-HELP
103750  000164                          .WORD   ACTNOD-10$      ;2-NODE
103752  000474                          .WORD   ACTBLD 10$      ;3 BUILD
```

```
6099 103754  003520                .WORD   ACTRUN-10$      ;4-RUN SPECIFIED TEST
6100 103756  006366                .WORD   ACTPAT-10$      ;5-SET 'MESSAGE PATTERN' TEST FLAG
6101 103760  010172                .WORD   ACTSAV-10$      ;6-SAVE NODE TABLE
6102 103762  001224                .WORD   ACTSUM 10$      ;7-PRINT SUMMARY TABLE
6103 103764  001606                .WORD   ACTIDT-10$      ;10-REQUEST ID
6104 103766  002632                .WORD   ACTEXT-10$      ;11-EXIT
6105 103770  000116                .WORD   ACTNUF-10$      ;12-NOT ENOUGH INFO
6106 103772  002642                .WORD   ACTXAD-10$      ;13-EXTRACT NI NODE ADDRESS FROM INPUT LINE
6107 103774  002724                .WORD   ACTSR4-10$      ;14-SAVE POINTER TO BEGINING OF ADDRESS STRING
6108 103776  007732                .WORD   ACTSND-10$      ;15-SET 'NODE' FLAG FOR SHOW COMMAND
6109 104000  002732                .WORD   ACTALP-10$      ;16-SET 'ALPHA' FLAG
6110 104002  002742                .WORD   ACTONE-10$      ;17-SET 'ONES' FLAG
6111 104004  002752                .WORD   ACTZRO-10$      ;20-SET 'ZEROS' FLAG
6112 104006  002762                .WORD   ACT1AL-10$      ;21-SET '1ALT' FLAG
6113 104010  002772                .WORD   ACTOAL-10$      ;22-SET 'OALT' FLAG
6114 104012  003002                .WORD   ACTCTT-10$      ;23-SET 'CCITT' FLAG
6115 104014  003012                .WORD   ACTOPR-10$      ;24-SET 'OPER SEL' FLAG
6116 104016  003140                .WORD   ACTTYP-10$      ;25-DETERMINE MESSAGE TYPE
6117 104020  003146                .WORD   ACTSZE-10$      ;26-DETERMINE MESSAGE SIZE
6118 104022  003224                .WORD   ACTCPY-10$      ;27-DETERMINE MESSAGE COPIES
6119 104024  003302                .WORD   ACTNAD-10$      ;30-SET 'NODE/ADDRESS' FLAG
6120 104026  003440                .WORD   ACTNAL-10$      ;31-SET 'NODE/ALL' FLAG
6121 104030  003646                .WORD   ACTRNA-10$      ;32-SET 'ALL' FLAG FOR RUN COMMAND
6122 104032  004724                .WORD   ACTRNL-10$      ;33-SET 'LOOPPAIR' FLAG FOR RUN CMD
6123 104034  006450                .WORD   ACTSMS-10$      ;34-SHOW CURRENT MESSAGE PARAMETERS
6124 104036  006542                .WORD   ACTCMS-10$      ;35-RESET MESSAGE PARAMETERS TO DEFAULT
6125 104040  006646                .WORD   ACTCNT-10$      ;36-SET 'COUNTER' FLAG FOR SHOW COMMAND
6126 104042  010106                .WORD   ACTCNL-10$      ;37-CLEAR LOGICAL NODE NAMED FROM TABLE
6127 104044  000124                .WORD   ACTNUL-10$      ;40 - DO NOTHING
6128 104046  010236                .WORD   ACTUNS-10$      ;41-UNSAVE NODE TABLE
6129 104050  010340                .WORD   ACTCSU-10$      ;42-CLEAR SUMMARY TABLE
6130 104052  004272                .WORD   ACTDIR-10$      ;43-SET 'LOOP DIRECT' FLAG FOR RUN COMMAND
6131 104054  010414                .WORD   ACTDFT-10$      ;44-LOOK FOR PASS COUNT DEFAULT
6132 104056  010462                .WORD   ACTUSF-10$      ;45-UNSAVE NODE TABLE FROM A FILE
6133 104060  010744                .WORD   ACTRSP-10$      ;46   RESPONDER MODE
```

```
                                        ;
                                        ;ACTION ROUTINE TO INDICATE THAT NOT ENOUGH COMMAND
                                        ;INFORMATION HAS BEEN ENTERED
                                        ;
104062                                  ACTNUF:
104062  112737  177777  003220                MOVB    #1,P$NNUF               ;SET FLAG TO SAY NEED MORE OF COMMAND

                                        ;
                                        ;ACTION ROUTINE TO DO NOTHING
                                        ;

104070                                  ACTNUL:
104070  000207                                RTS     PC                      ;RETURN TO PARSER


                                        ;
                                        ;ACTION ROUTINE TO PRINT OUT HELP FILE
                                        ;

104072                                  ACTHLP:
104072                                        P$PUSH  R2                      ;SAVE R2
104072  010225                                                                        MOV     R2,(R5)+
104074  012702  003224                       MOV     #HLPTAB,R2              ;MOVE POINTER TO BEGINING OF HELP FILE
104100                                  10$:  PRINTF  (R2)+                   ;PRINT LINE AND INCREMENT POINTER
104100  012246                                                                          MOV     (R2)+,-(SP)
104102  012746  000001                                                                  MOV     #1,-(SP)
104106  010600                                                                        MOV     SP,R0
104110  104417                                                                        TRAP    C$PNTF
104112  062706  000004                                                                ADD     #4,SP
104116  020227  003322                       CMP     R2,#HLPEND              ;SEE IF ENTIRE FILE PRINTED
104122  001366                                BNE     10$                     ;IF NOT, PRINT MORE
104124                                        P$POP   R2                      ;ELSE, RESTORE R2 AND RETURN
104124  014502                                                                        MOV     -(R5),R2
104126  000207                                RTS     PC


                                        ;
                                        ;ACTION ROUTINE TO READ IN NODE PHY. ADDRESS, STORE IT IN ADRBUF
                                        ;AND ENTER IT INTO THE NODE TABLE
                                        ;

104130                                  ACTNOD:
104130  105037  003220                       CLRB    P$NNUF                  ;CLEAR NOTNUF FLAG
104134  004737  101656                       JSR     PC,TRVADR               ;TRAVERSE ADDRESS, CHECK IF TARGET OR ASSIST
104140  105737  003221                       TSTB    P$GDBD                  ;CHECK IF RESULTS OK
104144  001134                                BNE     60$                     ;IF NOT, RETURN WITH -1 IN P$GDBD
104146                                  10$:  CALL    EDPACK CBOADR,#ADRBUF,#6        ;GET ADDRESS INTO BUFFER
104146  012725  000006                                                                MOV     #6,(R5)+
104152  012725  002316                                                                MOV     #ADRBUF,(R5)+
104156  013725  002370                                                                MOV     CBOADR,(R5)+
104162  004437  070656                                                                JSR     R4,PREG14
104166  004600                                                                        .WORD   EDPACK-ANCHOR
104170                                        P$POP   R1                      ;CHECK RESULTS FOR NUMBER OF CHAR.S
104170  014501                                                                        MOV     -(R5),R1
104172  001411                                BEQ     20$                     ;IF OK, BRANCH TO 20$
104174                                        PRINTF  #CADRER                 ;ELSE PRINT ERROR MESSAGE
104174  012746  052746                                                                MOV     #CADRER, (SP)
```

```
                104200  012746  000001                                                  MOV     #1, (SP)
                104204  010600                                                          MOV     SP,R0
                104206  104417                                                          TRAP    C$PNTF
                104210  062706  000004                                                  ADD     #4,SP
    6178        104214  000510                  BR      60$                     ;AND RETURN
    6179        104216                  20$:    CALL    CMPADR #ADRBUF,#ILLADR          ;SEE IF ILLEGAL ADDRESS
                104216  012725  002656                                                  MOV     #ILLADR,(R5)+
                104222  012725  002316                                                  MOV     #ADRBUF,(R5)+
                104226  004437  070656                                                  JSR     R4,PREG14
                104232  026004                                                          .WORD   CMPADR ANCHOR
    6180        104234                  P$POP   R1
                104234  014501                                                          MOV     (R5),R1
    6181        104236  001021                  BNE     30$                     ;IF YES, PRINT ERROR MESSAGE
    6182        104240                  PRINTF  #ILADMS
                104240  012746  052565                                                  MOV     #ILADMS, (SP)
                104244  012746  000001                                                  MOV     #1, (SP)
                104250  010600                                                          MOV     SP,R0
                104252  104417                                                          TRAP    C$PNTF
                104254  062706  000004                                                  ADD     #4,SP
    6183        104260                  PRINTF  #ILADM1
                104260  012746  052651                                                  MOV     #ILADM1, (SP)
                104264  012746  000001                                                  MOV     #1, (SP)
                104270  010600                                                          MOV     SP,R0
                104272  104417                                                          TRAP    C$PNTF
                104274  062706  000004                                                  ADD     #4,SP
    6184        104300  000456                  BR      60$
    6185        104302                  30$:    CALL    BINHEX #ADRBUF,#6,#STRBUF       ;CONVERT BINARY ADDRESS
                104302  012725  002324                                                  MOV     #STRBUF,(R5)+
                104306  012725  000006                                                  MOV     #6,(R5)+
                104312  012725  002316                                                  MOV     #ADRBUF,(R5)+
                104316  004437  070656                                                  JSR     R4,PREG14
                104322  005150                                                          .WORD   BINHEX ANCHOR
    6186                                                                        ;INTO ASCII STRING
    6187        104324  022737  000001  003734          CMP     #CASIST,CFLAG           ;SEE IF TARGET OR ASSIST
    6188        104332  001407                  BEQ     40$
    6189        104334  012737  061757  002314          MOV     #ARGTY7,KEYWD2          ;MOVE 'TARGET' INTO KEYWD2
    6190        104342  012737  000000  002406          MOV     #CTARGT,NODTY           ;MOVE TARGET INTO NODE TYPE
    6191        104350  000406                  BR      50$
    6192        104352  012737  061747  002314  40$:    MOV     #ARGTY6,KEYWD2          ;MOVE 'ASSIST' INTO KEYWD2
    6193        104360  012737  000001  002406          MOV     #CASIST,NODTY
    6194        104366                  50$:    CALL    ENTRND                  ;CALL ROUTINE TO ENTER NODE ON TABLE
                104366  004437  070656                                                  JSR     R4,PREG14
                104372  025410                                                          .WORD   ENTRND-ANCHOR
    6195        104374                  P$POP   R1                              ;CHECK RESULTS
                104374  014501                                                          MOV     -(R5),R1
    6196        104376  001017                  BNE     60$                     ;IF NODE TABLE FULL, RETURN
    6197        104400  012737  061654  002312          MOV     #CMDTY7,KEYWD1          ;ELSE, MOVE 'NODE' INTO KEYWD1
    6198        104406                  PRINTS  #MSG2,KEYWD2,#STRBUF            ;INDICATE IF TARGET OR ASSIST
                104406  012746  002324                                                  MOV     #STRBUF, (SP)
                104412  013746  002314                                                  MOV     KEYWD2, (SP)
                104416  012746  054703                                                  MOV     #MSG2,-(SP)
                104422  012746  000003                                                  MOV     #3, (SP)
                104426  010600                                                          MOV     SP,R0
                104430  104416                                                          TRAP    C$PNTS
                104432  062706  000010                                                  ADD     #10,SP
    6199        104436  000207                  60$:    RTS     PC
    6200
```

```
6201                                    ;
6202                                    ;ACTION ROUTINE TO BUILD NODE TABLE USING THE PERIODIC ADDRESS
6203                                    ;IDENTIFICATION MESSAGES SENT OUT BY NODE ON THE NI
6204                                    ;
6205
6206 104440                            ACTBLD:
6207 104440                                    PRINTS  #MSG1                   ; PRINT  BUILD' COMMAND MESSAGE
     104440  012746  054415                                                             MOV     #MSG1,(SP)
     104444  012746  000001                                                             MOV     #1,(SP)
     104450  010600                                                                     MOV     SP,R0
     104452  104416                                                                     TRAP    C$PNTS
     104454  062706  000004                                                             ADD     #4,SP
6208 104460                                    PRINTS  #MSG11
     104460  012746  054530                                                             MOV     #MSG11,(SP)
     104464  012746  000001                                                             MOV     #1,-(SP)
     104470  010600                                                                     MOV     SP,R0
     104472  104416                                                                     TRAP    C$PNTS
     104474  062706  000004                                                             ADD     #4,SP
6209 104500                                    PRINTS  #MSG12
     104500  012746  054643                                                             MOV     #MSG12,(SP)
     104504  012746  000001                                                             MOV     #1,(SP)
     104510  010600                                                                     MOV     SP,R0
     104512  104416                                                                     TRAP    C$PNTS
     104514  062706  000004                                                             ADD     #4,SP
6210 104520                                    P$PUSH  R2,R3,R4                ; SAVE REGISTERS
     104520  010425                                                                             MOV     R4,(R5)+
     104522  010325                                                                             MOV     R3,(R5)+
     104524  010225                                                                             MOV     R2,(R5)+
6211 104526                                    CALL    SETUP   #ADDMUL         ; WRITE MULTICAST ADDRESS
     104526  012725  000001                                                                     MOV     #ADDMUL,(R5)+
     104532  004437  070656                                                                     JSR     R4,PREG14
     104536  001232                                                                             .WORD   SETUP-ANCHOR
6212 104540                                    P$POP   R2                      ; CHECK FOR ERROR
     104540  014502                                                                             MOV     -(R5),R2
6213 104542  001406                                    BEQ     10$             ; IF OK, CONTINUE
6214 104544                                    ERRHRD  25,EMSG25               ; ELSE REPORT ERROR
     104544  104456                                                                     TRAP    C$ERHRD
     104546  000031                                                                     .WORD   25
     104550  064157                                                                     .WORD   EMSG25
     104552  000000                                                                     .WORD   0
6215 104554  000137  105154                        JMP     100$                    ; LEAVE
6216 104560                            10$:
6217 104560  005237  002200                        INC     BLDFLG                  ; INDICATE THAT WE ARE IN BUILD MODE
6218 104564  005037  050314                        CLR     TEMP                    ; CLEAR 'NO. NODES IN LAST MIN.  COUNTER
6219 104570  005037  050316                        CLR     TEMP1                   ; CLEAR NODE TYPE ARGUMENT (SET TO TARGET)
6220 104574  005037  050320                        CLR     TEMP2                   ; SET INTERVAL COUNTER
6221 104600  012737  000013  050322                MOV     #13,TEMP3               ; SET 'MINS. SINCE LAST NEW NODE  COUNTER
6222 104606  012737  002412  002410                MOV     #NODTBL,SLOT            ; SET SLOT TO BEGINING OF NODE TABLE
6223 104614  012704  003762                        MOV     #TIMERS,R4              ; SET UP FOR 1 MINUTE LOOP
6224 104620                            20$:
6225 104620  012714  000074                        MOV     #60.,(R4)
6226 104624                            30$:
6227 104624                                    BREAK                           ; ALLOW FOR CONTROL C INTERRUPTION
     104624  104422                                                                     TRAP    C$BRK
6228 104626  005714                            TST     (R4)                    ; SEE IF INTERVAL IS UP
6229 104630  001503                            BEQ     70$                     ; IF YES, BRANCH
6230 104632                                    CALL    RECEVE                  ;  ELSE, CHECK FOR RECEPTION OF ID MESSAGE
```

```
                                                                        JSR     R4,PREG14
                                                                        .WORD   RECEVE-ANCHOR
                        P$POP   R2                      ; R2 HOLDS NO OF MESSAGES RECEIVED
                                                                        MOV     (R5),R2
                        BEQ     30$                     ; IF NONE, KEEP LOOKING
                        MOV     RRGNXT,R3               ; IF ONE, GET RECEIVE RING POINTER
                        CALL    GETRNX  #1,#RRGNXT      ; UPDATE POINTER, VALIDATE PREVIOUS DESCRIPTOR
                                                                        MOV     #RRGNXT,(R5)+
                                                                        MOV     #1,(R5)+
                                                                        JSR     R4,PREG14
                                                                        .WORD   GETRNX ANCHOR
                        MOV     LOADD(R3),R3            ; POINT R3 TO MESSAGE BUFFER
                        ADD     #SOURCC,R3              ; POINT R3 TO NODE ADDRESS
                        MOV     #NODTBL,R2              ; POINT R2 TO NODE TABLE
                40$:
                        CALL    CMPADR  R2,R3           ; SEE IF NODE ALREADY ON TABLE
                                                                        MOV     R3,(R5)+
                                                                        MOV     R2,(R5)+
                                                                        JSR     R4,PREG14
                                                                        .WORD   CMPADR ANCHOR
                        P$POP   R1                      ; GET RESULTS OF COMPARE
                                                                        MOV     (R5),R1
                        BEQ     30$                     ; DON'T ADD TO TABLE IF ALREADY THERE
                50$:
                        ADD     #10,R2                  ;  ELSE CHECK NEXT TABLE ENTRY
                        CMP     # 1,(R2)               ;  SEE IF AT END OF TABLE
                        BNE     40$                     ;   IF NO, COMPARE NEXT ENTRY
                        CALL    FINDSL                  ;   IF YES, GET NEXT TABLE SLOT
                                                                        JSR     R4,PREG14
                                                                        .WORD   FINDSL ANCHOR
                        P$POP   R2                      ; SEE IF TABLE FULL
                                                                        MOV     (R5),R2
                        BNE     60$                     ;  IF YES, BRANCH
                        MOV     SLOT,R2                 ;  IF NO, ADD NODE TO TABLE
                        MOV     (R3)+,(R2)+             ;
                        MOV     (R3)+,(R2)+             ;   SIX BYTES WORTH
                        MOV     (R3),(R2)               ;
                        MOVB    TEMP1,3(R2)             ; SET NODE TYPE (TARGET OR ASSIST)
                        COMB    TEMP1                   ;  SWITCH TYPE FOR NEXT TIME
                        BICB    #376,TEMP1              ;
                        INC     TEMP                    ; INCREMENT 'NODES IN LAST MIN.  COUNTER
                        MOV     #13,TEMP3               ; RESET 'MINS. SINCE LAST NODE' COUNTER
                        BR      30$
                60$:
                        PRINTB  #TABFUL,#NOD            ; PRINT "TABLE FULL" MESSAGE
                                                                        MOV     #NOD,(SP)
                                                                        MOV     #TABFUL,(SP)
                                                                        MOV     #2,(SP)
                                                                        MOV     SP,R0
                                                                        TRAP    C$PNTB
                                                                        ADD     #6,SP
                        BR      80$
                70$:
                        DEC     TEMP3                   ; SEE IF 10 MINS SINCE LAST NODE
                        BEQ     80$                     ;  IF YES, EXIT
                        INC     TEMP2                   ; SEE IF TIME IS UP
                        CMP     TEMP2,#40.
```

```
6267 105060  001417                    BEQ     80$                     ; IF YES, EXIT
6268 105062                            PRINTS  #BLDMSG,TEMP,TEMP2      ;  ELSE, PRINT "STILL WORKING  MESSAGE
     105062  013746  050320                                                    MOV     TEMP2,(SP)
     105066  013746  050314                                                    MOV     TEMP,(SP)
     105072  012746  052473                                                    MOV     #BLDMSG,(SP)
     105076  012746  000003                                                    MOV     #3,(SP)
     105102  010600                                                            MOV     SP,R0
     105104  104416                                                            TRAP    C$PNTS
     105106  062706  000010                                                    ADD     #10,SP
6269 105112  005037  050314                    CLR     TEMP
6270 105116  000640                            BR      20$
6271 105120                            80$:
6272 105120                            CALL    SETUP   #KILMUL         ;  KILL THE MULTICAST ADDRESS
     105120  012725  000002                                                    MOV     #KILMUL,(R5)+
     105124  004437  070656                                                    JSR     R4,PREG14
     105130  001232                                                            .WORD   SETUP ANCHOR
6273 105132                            P$POP   R2                      ; CHECK FOR ERROR
     105132  014502                                                            MOV     -(R5),R2
6274 105134  001405                    BEQ     90$                     ;  IF OK CONTINUE
6275 105136                            ERRHRD  34,EMSG25               ;  ELSE, REPORT ERROR
     105136  104456                                                            TRAP    C$ERHRD
     105140  000042                                                            .WORD   34
     105142  064157                                                            .WORD   EMSG25
     105144  000000                                                            .WORD   0
6276 105146  000402                    BR      100$                    ; AND LEAVE
6277 105150                            90$:
6278 105150  004737  113676            JSR     PC,ACTSND               ; PRINT NODE TABLE
6279 105154                            100$:
6280 105154  005037  002200            CLR     BLDFLG                  ; INDICATE THAT WE ARE NO LONGER BUILDING
6281 105160                            P$POP   R2,R3,R4                ; RESTORE REGISTERS
     105160  014502                                                            MOV     -(R5),R2
     105162  014503                                                            MOV     -(R5),R3
     105164  014504                                                            MOV     (R5),R4
6282 105166  000207                    RTS     PC
6283                                   ;
6284                                   ;ACTION ROUTINE TO PRINT OUT THE SUMMARY DATA
6285                                   ;
6286 105170  105037  003220            ACTSUM: CLRB    P$NNUF                  ;CLEAR NOTNUF FLAG
6287 105174                                    P$PUSH  R2,R3,R4
     105174  010425                                                            MOV     R4,(R5)+
     105176  010325                                                            MOV     R3,(R5)+
     105200  010225                                                            MOV     R2,(R5)+
6288 105202  012701  002714                    MOV     #STATBL,R1              ; MOVE ADDRESS OF TABLE TO R1
6289 105206  005711                            TST     (R1)                    ; SEE IF TABLE EMPTY
6290 105210  001013                            BNE     10$                     ;  IF NOT, CONT.
6291 105212                                    PRINTF  #TABFMT,#SUMM           ;  ELSE PRINT 'TABLE EMPTY' MESSAGE
     105212  012746  053422                                                            MOV     #SUMM,(SP)
     105216  012746  053346                                                            MOV     #TABEMT,(SP)
     105222  012746  000002                                                            MOV     #2,(SP)
     105226  010600                                                                    MOV     SP,R0
     105230  104417                                                                    TRAP    C$PNTF
     105232  062706  000006                                                            ADD     #6,SP
6292 105236  000541                            BR      20$                     ;  EXIT
6293 105240  021127  177777    10$:            CMP     (R1),#-1                ; SEE IF AT END OF TABLE
6294 105244  001536                            BEQ     20$                     ;  IF YES, EXIT
6295 105246  005711                            TST     (R1)                    ; SEE IF REST OF TABLE EMPTY
6296 105250  001534                            BEQ     20$                     ;  IF YES, EXIT
```

```
                        CALL    BINHEX  R1,#6,#STRBUF   ; PRINT SUMMARY DATA
                                                                MOV     #STRBUF,(R5)+
                                                                MOV     #6,(R5)+
                                                                MOV     R1,(R5)+
                                                                JSR     R4,PREG14
                                                                .WORD   BINHEX ANCHOR
                        PRINTF  #SUMMS1,#STRBUF         ;  NODE ADDRESS
                                                                MOV     #STRBUF,-(SP)
                                                                MOV     #SUMMS1,-(SP)
                                                                MOV     #2,-(SP)
                                                                MOV     SP,R0
                                                                TRAP    C$PNTF
                                                                ADD     #6,SP
                        PRINTF  #SUMMS2                 ;  FIRST HEADER
                                                                MOV     #SUMMS2,-(SP)
                                                                MOV     #1,-(SP)
                                                                MOV     SP,R0
                                                                TRAP    C$PNTF
                                                                ADD     #4,SP
                        MOV     #26,R2                  ; GET STATISTICS SIZE FOR THIS NODE
                        MOV     #STATBF,R3              ; GET BUFFER ADDRESS FOR STATISTICS
                15$:
                        MOVB    (R1)+,(R3)+             ; BUFFER THE STATISTICS
                        DEC     R2                      ; HAVE ALL BYTES FOR NODE BEEN XFERRED?
                        BNE     15$                     ; BRANCH IF NOT
                        MOV     STATBF+6,R2             ; RX NOT COMPLETE
                        MOV     STATBF+10,R3            ; RX COMPLETE
                        MOV     STATBF+12,R4            ; LENGTH ERRORS
                        PRINTF  #SUMMS3,R2,R3,R4        ; PRINT THEM OUT
                                                                MOV     R4,-(SP)
                                                                MOV     R3,-(SP)
                                                                MOV     R2,-(SP)
                                                                MOV     #SUMMS3,-(SP)
                                                                MOV     #4,-(SP)
                                                                MOV     SP,R0
                                                                TRAP    C$PNTF
                                                                ADD     #12,SP
                        PRINTF  #SUMMS4                 ; SECOND HEADER
                                                                MOV     #SUMMS4,-(SP)
                                                                MOV     #1,-(SP)
                                                                MOV     SP,R0
                                                                TRAP    C$PNTF
                                                                ADD     #4,SP
                        MOV     STATBF+14,R2            ; COMPARE ERRORS
                        CALL    BINDEC  #STATBF+16      ; PUT BYTES COMPARED INTO ASCII STRING
                                                                MOV     #STATBF+16,(R5)+
                                                                JSR     R4,PREG14
                                                                .WORD   BINDEC ANCHOR
                        PRINTF  #SUMMS5,R2,#DECSTR      ; PRINT THEM OUT
                                                                MOV     #DECSTR,-(SP)
                                                                MOV     R2,-(SP)
                                                                MOV     #SUMMS5,-(SP)
                                                                MOV     #3,-(SP)
                                                                MOV     SP,R0
                                                                TRAP    C$PNTF
                                                                ADD     #10,SP
                        CALL    BINDEC  #STATBF+22      ; PUT BYTES TRANSFERED INTO ASCII STRING
```

```
        105502  012725  002706                                                  MOV     #STATBF+22,(R5)+
        105506  004437  070656                                                  JSR     R4,PREG14
        105512  007446                                                          .WORD   BINDEC-ANCHOR
 6315   105514                          PRINTF  #SUMMS6,#DECSTR         ; PRINT
        105514  012746  100526                                                          MOV     #DECSTR,(SP)
        105520  012746  070300                                                          MOV     #SUMMS6,(SP)
        105524  012746  000002                                                          MOV     #2,-(SP)
        105530  010600                                                          MOV     SP,R0
        105532  104417                                                          TRAP    C$PNTF
        105534  062706  000006                                                  ADD     #6,SP
 6316   105540  000637                  BR      10$                     ; PLAY IT AGAIN, SAM
 6317   105542                  20$:    P$POP   R2,R3,R4
        105542  014502                                                  MOV     -(R5),R2
        105544  014503                                                  MOV     -(R5),R3
        105546  014504                                                  MOV     -(R5),R4
 6318   105550  000207                  RTS     PC
 6319                           ;
 6320                           ;ACTION ROUTINE TO INITIATE THE REQUEST ID TEST TO THE SPECIFIED NODE
 6321                           ;
 6322                           ;--*
 6323                           ; FUNCTIONAL DESCRIPTION
 6324                           ;               THIS SUBROUTINE BUILDS AND TRANSMITS REQUEST ID PACKETS
 6325                           ;               TO THE NODE SPECIFIED BY THE OPERATOR IN THE COMMAND LINE.
 6326                           ;               THE SYSTEM ID INFO OF THE SPECIFIED NODE IS THEN DISPLAYED.
 6327                           ;               IF THE NODE DOES NOT RESPOND BEFORE 60 SECONDS HAVE PASSED
 6328                           ;               AN ERROR IS REPORTED TO THE OPERATOR.
 6329                           ;
 6330                           ; INPUTS -      IMPLICIT - THE SPECIFIED NODE ADDRESS IS LOCATED IN ADRBUF.
 6331                           ;
 6332                           ; OUTPUTS -     SYSTEM ID INFO OR ERROR MESSAGE PRINTED TO OPERATOR.
 6333                           ;
 6334                           ; CALLING PROCEDURE - JSR PC, ACTIDT
 6335                           ;
 6336                           ; SIDE EFFECTS - XRGNXT POINTER IS UPDATED BY A CALL TO BLDREQ SUB.
 6337                           ;
 6338                           ; REGISTER USAGE - R1 POINTS TO $WDMO FOR WRITE MODE OPERATIONS.
 6339                           ;                  R2 IS SCRATCH.
 6340                           ;                  R3 POINTS TO THE RECEIVED MESSAGE BUFFER.
 6341                           ;                  R4 POINTS TO TIMOUT TIMER
 6342                           ;
 6343                           ;-*
 6344   105552                  ACTIDT:
 6345   105552  105737  003222          TSTB    P$AERR                  ;SEE IF ADDRESS ENTERED WAS VALID
 6346   105556  001402                  BEQ     10$                     ; IF NOT, EXIT ACTION ROUTINE
 6347   105560  000137  106466          JMP     130$
 6348   105564                  10$:
 6349   105564  105037  003220          CLRB    P$NNUF                  ;CLEAR NOTNUF FLAG
 6350   105570                          CALL    CMPADR  #ADRBUF,#ILLADR ; SEE IF ILLEGAL ADDRESS
        105570  012725  002656                                                  MOV     #ILLADR,(R5)+
        105574  012725  002316                                                  MOV     #ADRBUF,(R5)+
        105600  004437  070656                                                  JSR     R4,PREG14
        105604  026004                                                          .WORD   CMPADR ANCHOR
 6351   105606                          P$POP   R1
        105606  014501                                                  MOV     -(R5),R1
 6352   105610  001012                  BNE     20$                     ;  IF NO, CONTINUE
 6353   105612                  15$:
 6354   105612                          PRINTF  #ILADMS                 ;  ELSE PRINT ILLEGAL ADDRESS MESSAGE
```

```
                105612  012746  052565                                          MOV     @ILADMS,(SP)
                105616  012746  000001                                          MOV     @1,(SP)
                105622  010600                                                  MOV     SP,R0
                105624  104417                                                  TRAP    C$PNTF
                105626  062706  000004                                          ADD     @4,SP
 6355           105632  000137  106466          JMP     130$
 6356           105636                  20$:
 6357           105636          P$PUSH  R1,R2,R3,R4             ; SAVE REGISTERS
                105636  010425                                                  MOV     R4,(R5)+
                105640  010325                                                  MOV     R3,(R5)+
                105642  010225                                                  MOV     R2,(R5)+
                105644  010125                                                  MOV     R1,(R5)+
 6358           105646          CALL    CMPADR  @ADRBUF,@PHYADR ; SEE IF ADDRESS IS OWN (HOST NODE)
                105646  012725  004010                                          MOV     @PHYADR,(R5)+
                105652  012725  002316                                          MOV     @ADRBUF,(R5)+
                105656  004437  070656                                          JSR     R4,PREG14
                105662  026004                                                  .WORD   CMPADR ANCHOR
 6359           105664          P$POP   R1                      ;
                105664  014501                                                  MOV      (R5),R1
 6360           105666  001006          BNE     25$
 6361           105670          ERRHRD  64,EMSG64               ; REPORT THAT WE CAN T USE OUR OWN ADDRESS
                105670  104456                                                  TRAP    C$ERHRD
                105672  000100                                                  .WORD   64
                105674  066615                                                  .WORD   EMSG64
                105676  000000                                                  .WORD   0
 6362           105700  000137  106456          JMP     120$                    ; RESTORE REGISTERS AND EXIT
 6363           105704                  25$:
 6364           105704  012737  177776  050320  MOV     @-2,TEMP2               ; SET COUNTER FOR NO. OF TIMES TRIED
 6365           105712          CALL    BLDREQ                  ; BUILD REQUEST ID MESSAGE PACKET
                105712  004437  070656                                          JSR     R4,PREG14
                105716  006330                                                  .WORD   BLDREQ-ANCHOR
 6366           105720          CALL    XMIT    @XMTDAT         ; TRANSMIT REQUEST
                105720  012725  000001                                          MOV     @XMTDAT,(R5)+
                105724  004437  070656                                          JSR     R4,PREG14
                105730  002326                                                  .WORD   XMIT-ANCHOR
 6367           105732          P$POP   R2                      ; GET RESULTS, R2 = SUCCESS/FAILURE
                105732  014502                                                  MOV     -(R5),R2
 6368           105734  001402          BEQ     30$             ;  IF OK BRANCH
 6369           105736  000137  106430  JMP     90$             ;  ELSE JUMP TO 90$
 6370           105742                  30$:
 6371           105742  005737  050274  TST     RETRYS          ; SEE IF FAILED DUE TO EXCESSIVE COLLISIONS
 6372           105746  001412          BEQ     40$             ;  IF NO, CONT.
 6373           105750          PRINTF  @RTRYER                 ;  YES, PRINT 'EXCESSIVE COLLISIONS' MESSAGE
                105750  012746  052411                                          MOV     @RTRYER,(SP)
                105754  012746  000001                                          MOV     @1,(SP)
                105760  010600                                                  MOV     SP,R0
                105762  104417                                                  TRAP    C$PNTF
                105764  062706  000004                                          ADD     @4,SP
 6374           105770  000137  106456          JMP     120$                    ;  EXIT
 6375           105774                  40$:
 6376           105774  012704  003762  MOV     @TIMERS,R4              ; SET UP FOR 10 SECOND TIMOUT
 6377           106000  012714  000012  MOV     @10.,(R4)
 6378           106004                  50$:
 6379           106004          BREAK
                106004  104422                                                  TRAP    C$BRK
 6380           106006  005714          TST     (R4)                    ; SEE IF TIME HAS EXPIRED
 6381           106010  001427          BEQ     60$                     ;  IF YES, BRANCH
```

```
6382 106012                                    CALL    RECEVE                  ; CHECK FOR ANSWER
     106012  004437  070656                                                             JSR     R4,PREG14
     106016  003044                                                                     .WORD   RECEVE-ANCHOR
6383 106020                                    P$POP   R2                      ;  R2 HOLDS NO. OF BUFFERS RECEIVED
     106020  014502                                                                     MOV     -(R5),R2
6384 106022  001770                                    BEQ     50$             ;  IF NO BUFFERS RECEIVED, LOOP
6385 106024  013703  024226                           MOV     RRGNXT,R3       ; GET RECEIVE RING POINTER
6386 106030  016303  000004                           MOV     LOADD(R3),R3    ; POINT R3 TO MESSAGE BUFFER
6387 106034  026327  000022  051115                   CMP     SIRCPT(R3),#"MR ; SEE IF MESSAGE RECEIVED IS IN REPLY TO ONE SENT
6388 106042  001426                                    BEQ     70$             ;  IF YES, BRANCH TO 70$
6389 106044  005237  050320                           INC     TEMP2           ;  INCREMENT RETRY COUNTER
6390 106050                                    CALL    GETRNX  #1,#RRGNXT      ; THROW OUT THE PACKET
     106050  012725  024226                                                             MOV     #RRGNXT,(R5)+
     106054  012725  000001                                                             MOV     #1,(R5)+
     106060  004437  070656                                                             JSR     R4,PREG14
     106064  006450                                                                     .WORD   GETRNX-ANCHOR
6391 106066  001346                                    BNE     50$             ;  IF NO, LOOK FOR CORRECT REPLY MESSAGE
6392 106070                                60$:
6393 106070                                    PRINTF  #EMSG22                 ;  ELSE, REPORT ERROR
     106070  012746  064050                                                                     MOV     #EMSG22, (SP)
     106074  012746  000001                                                                     MOV     #1, (SP)
     106100  010600                                                                             MOV     SP,R0
     106102  104417                                                                             TRAP    C$PNTF
     106104  062706  000004                                                                     ADD     #4,SP
6394 106110  005237  050032                           INC     S.NREC          ;  UPDATE SUMMARY DATA
6395 106114  000137  106440                           JMP     100$            ;  AND EXIT
6396 106120                                70$:
6397 106120  005237  050030                           INC     S.REC           ; INCREMENT 'RECEIVED MESSAGES  COUNTER
6398 106124  062737  000056  050042                   ADD     #46.,S.XFER     ; UPDATE 'BYTES TRANSFERED  COUNTER
6399 106132  010302                                    MOV     R3,R2           ; PUT POINTER INTO R2
6400 106134  062702  000042                           ADD     #SIADDR,R2      ; POINT R2 TO ADDRESS
6401 106140                                80$:
6402 106140                                    CALL    BINHEX R2,#6,#STRBUF    ; PUT ADDRESS INTO STRBUF
     106140  012725  002324                                                             MOV     #STRBUF,(R5)+
     106144  012725  000006                                                             MOV     #6,(R5)+
     106150  010225                                                                     MOV     R2,(R5)+
     106152  004437  070656                                                             JSR     R4,PREG14
     106156  005150                                                                     .WORD   BINHEX ANCHOR
6403 106160                                    PRINTF  #SIMSG1,#STRBUF         ; PRINT REMOTE NODE ADDRESS
     106160  012746  002324                                                                     MOV     #STRBUF, (SP)
     106164  012746  067266                                                                     MOV     #SIMSG1, (SP)
     106170  012746  000002                                                                     MOV     #2, (SP)
     106174  010600                                                                             MOV     SP,R0
     106176  104417                                                                             TRAP    C$PNTF
     106200  062706  000006                                                                     ADD     #6,SP
6404 106204  016302  000022                           MOV     SIRCPT(R3),R2   ; GET RECEIPT NUMBER
6405 106210                                    PRINTF  #SIMSG2,R2              ; PRINT RECEIPT NUMBER
     106210  010246                                                                     MOV     R2,-(SP)
     106212  012746  067325                                                             MOV     #SIMSG2,-(SP)
     106216  012746  000002                                                             MOV     #2,-(SP)
     106222  010600                                                                     MOV     SP,R0
     106224  104417                                                                     TRAP    C$PNTF
     106226  062706  000006                                                             ADD     #6,SP
6406 106232  116302  000027                           MOVB    SIVERS(R3),R2   ; GET VERSION NO. AND PRINT
6407 106236                                    PRINTF  #SIMSG3,R2
     106236  010246                                                                     MOV     R2, (SP)
     106240  012746  067360                                                             MOV     #SIMSG3, (SP)
```

```
                106244  012746  000002                                          MOV     #2,-(SP)
                106250  010600                                                  MOV     SP,R0
                106252  104417                                                  TRAP    C$PNTF
                106254  062706  000006                                          ADD     #6,SP
6408            106260  116302  000030          MOVB    SIECO(R3),R2            ; GET ECO NO. AND PRINT
6409            106264                          PRINTF  #SIMSG4,R2
                106264  010246                                                  MOV     R2,-(SP)
                106266  012746  067420                                          MOV     #SIMSG4,-(SP)
                106272  012746  000002                                          MOV     #2, (SP)
                106276  010600                                                  MOV     SP,R0
                106300  104417                                                  TRAP    C$PNTF
                106302  062706  000006                                          ADD     #6,SP
6410            106306  116302  000031          MOVB    SIUECO(R3),R2           ; GET USER ECO NO. AND PRINT
6411            106312                          PRINTF  #SIMSG5,R2
                106312  010246                                                  MOV     R2,-(SP)
                106314  012746  067441                                          MOV     #SIMSG5,-(SP)
                106320  012746  000002                                          MOV     #2, (SP)
                106324  010600                                                  MOV     SP,R0
                106326  104417                                                  TRAP    C$PNTF
                106330  062706  000006                                          ADD     #6,SP
6412            106334  116302  000035          MOVB    SIFNCT(R3),R2           ; GET FUNCTON CODE AND PRINT
6413            106340                          PRINTF  #SIMSG6,R2
                106340  010246                                                  MOV     R2, (SP)
                106342  012746  067467                                          MOV     #SIMSG6, (SP)
                106346  012746  000002                                          MOV     #2,-(SP)
                106352  010600                                                  MOV     SP,R0
                106354  104417                                                  TRAP    C$PNTF
                106356  062706  000006                                          ADD     #6,SP
6414            106362  116302  000053          MOVB    SIDEV(R3),R2            ; GET DEVICE TYPE AND PRINT
6415            106366                          PRINTF  #SIMSG7,R2
                106366  010246                                                  MOV     R2,-(SP)
                106370  012746  067515                                          MOV     #SIMSG7, (SP)
                106374  012746  000002                                          MOV     #2,-(SP)
                106400  010600                                                  MOV     SP,R0
                106402  104417                                                  TRAP    C$PNTF
                106404  062706  000006                                          ADD     #6,SP
6416            106410                          CALL    GETRNX  #1,#RRGNXT      ; UPDATE POINTER, VALIDATE PREVIOUS DESCRIPTOR
                106410  012725  024226                                                  MOV     #RRGNXT,(R5)+
                106414  012725  000001                                                  MOV     #1,(R5)+
                106420  004437  070656                                                  JSR     R4,PREG14
                106424  006450                                                          .WORD   GETRNX ANCHOR
6417            106426  000404                  BR      100$                    ; EXIT
6418            106430                  90$:
6419            106430                          ERRDF   24,EMSG24,ERR1          ; ERROR     CAN'T TRANSMIT PACKETS
                106430  104455                                                          TRAP    C$ERDF
                106432  000030                                                          .WORD   24
                106434  064103                                                          .WORD   EMSG24
                106436  070306                                                          .WORD   ERR1
6420            106440                  100$:
6421            106440                          CALL    WRITES  #1,#ADRBUF      ; UPDATE SUMMARY TABLE
                106440  012725  002316                                                  MOV     #ADRBUF,(R5)+
                106444  012725  000001                                                  MOV     #1,(R5)+
                106450  004437  070656                                                  JSR     R4,PREG14
                106454  007176                                                          .WORD   WRITES-ANCHOR
6422            106456                  110$:
6423            106456                  120$:
6424            106456                          P$POP   R1,R2,R3,R4             ; RESTORE REGISTERS
```

```
                                                                                MOV     (R5),R1
                                                                                MOV     -(R5),R2
                                                                                MOV     (R5),R3
                                                                                MOV     -(R5),R4
                        130$:
                                RTS     PC
                        ;
                        ;ACTION ROUTINE TO CHECK FOR ADDITION PARAMETER CHANGE INPUTS
                        ;AND PRINT OUT NEW PARAMETER INFO WHEN ALL INPUT ARE PROCESSED
                        ;
                        ACTMSG:
                                TSTB    (R4)                    ;CHECK FOR ADDITIONAL INPUT
                                BEQ     10$                     ;BR IF NO
                                BR      20$                     ;IF YES RETURN
                        10$:
                                MOV     #CMDTY6,KEYWD1
                                MOV     P$TYPE,R1               ;GET MESSAGE TYPE ASCII STRING ADDRESS
                                ASL     R1                      ;INTO R1
                                ADD     #MSGTAB,R1
                                PRINTF  #MSGPRM                 ;PRINT 'MESSAGE' COMMAND MESSAGE
                                                                                MOV     #MSGPRM,-(SP)
                                                                                MOV     #1,(SP)
                                                                                MOV     SP,R0
                                                                                TRAP    C$PNTF
                                                                                ADD     #4,SP
                                PRINTF  #MSG4,(R1),P$SIZE,P$CPYS        ;PRINT MSG PARAMETERS
                                                                                MOV     P$CPYS,-(SP)
                                                                                MOV     P$SIZE,-(SP)
                                                                                MOV     (R1),-(SP)
                                                                                MOV     #MSG4,(SP)
                                                                                MOV     #4,(SP)
                                                                                MOV     SP,R0
                                                                                TRAP    C$PNTF
                                                                                ADD     #12,SP
                                CLRB    P$NNUF                  ;CLEAR NOTNUF FLAG
                        20$:
                                RTS     PC
                        ;
                        ;ACTION ROUTINE TO RETURN CONTROL TO THE SUPERVISOR
                        ;
                        ACTEXT:
                                MOV     #CEXIT,CFLAG            ;SET EXIT FLAG
                                RTS     PC
                        ;
                        ;ACTION ROUTINE TO TAKE NI NODE ADDRESS FROM INPUT STRING BUFFER
                        ;AND STORE IT IN THE BUFFER CALLED ADRBUF
                        ;
                        ACTXAD:
                                CALL    EDPACK CBOADR,#ADRBUF,#6        ;PUT NODE ADDRESS INTO ADRBUF
                                                                                MOV     #6,(R5)+
                                                                                MOV     #ADRBUF,(R5)+
                                                                                MOV     CBOADR,(R5)+
                                                                                JSR     R4,PREG14
                                                                                .WORD   EDPACK-ANCHOR
                                P$POP   P$AERR                  ;SET ADDRESS+12 CHAR. GOOD/BAD FLAG
                                                                                MOV     (R5),P$AERR
                                TSTB    P$AERR                  ;IF GOOD, RETURN
```

```
 6459 106640  001412                          BEQ     10$
 6460 106642                                  PRINTF  @CADRER                 ;ELSE, PRINT ERROR MESSAGE
      106642  012746  052746                                                          MOV     @CADRER,-(SP)
      106646  012746  000001                                                          MOV     @1,-(SP)
      106652  010600                                                                 MOV     SP,R0
      106654  104417                                                                 TRAP    C$PNTF
      106656  062706  000004                                                          ADD     @4,SP
 6461 106662  105037  003220                  CLRB    P$NNUF                  ; AND CLEAR 'NOT ENOUGH' FLAG
 6462 106666                          10$:
 6463 106666  000207                          RTS     PC
 6464                                 ;
 6465                                 ;ACTION ROUTINE TO STORE POINTER TO BEGINING OF OPERATOR INPUT ADDRESS
 6466                                 ;IN COMMAND INPUT BUFFER
 6467                                 ;
 6468 106670  010437  002370          ACTSR4: MOV     R4,CBOADR               ;SAVE STRING POINTER
 6469 106674  000207                  10$:    RTS     PC
 6470                                 ;
 6471                                 ;ACTION ROUTINE TO SET MESSAGE TYPE = ALPHA FLAG
 6472                                 ;
 6473
 6474 106676  012737  000000  002372  ACTALP: MOV     @ALPHA,P$TYPE           ;SET MESSAGE TYPE
 6475 106704  000207                          RTS     PC
 6476                                 ;
 6477                                 ;ACTION ROUTINE TO SET MESSAGE TYPE = ALL ONES FLAG
 6478                                 ;
 6479 106706  012737  000001  002372  ACTONE: MOV     @ONES,P$TYPE            ;SET MESSAGE TYPE
 6480 106714  000207                          RTS     PC
 6481                                 ;
 6482                                 ;ACTION ROUTINE TO SET MESSAGE TYPE = ALL ZEROS FLAG
 6483                                 ;
 6484 106716  012737  000002  002372  ACTZRO: MOV     @ZEROS,P$TYPE           ;SET MESSAGE TYPE
 6485 106724  000207                          RTS     PC
 6486                                 ;
 6487                                 ;ACTION ROUTINE TO SET MESSAGE TYPE = ALTERNATING ONES FLAG
 6488                                 ;
 6489 106726  012737  000003  002372  ACT1AL: MOV     @ONEALT,P$TYPE          ;SET MESSAGE TYPE
 6490 106734  000207                          RTS     PC
 6491                                 ;
 6492                                 ;ACTION ROUTINE TO SET MESSAGE TYPE = ALTERNATING ZEROS FLAG
 6493                                 ;
 6494 106736  012737  000004  002372  ACTOAL: MOV     @ZROALT,P$TYPE          ;SET MESSAGE TYPE
 6495 106744  000207                          RTS     PC
 6496                                 ;
 6497                                 ;ACTION ROUTINE TO SET MESSAGE TYPE = CCITT FLAG
 6498                                 ;
 6499 106746  012737  000005  002372  ACTCTT: MOV     @CCITT,P$TYPE           ;SET MESSAGE TYPE
 6500 106754  000207                          RTS     PC
 6501                                 ;
 6502                                 ;ACTION ROUTINE TO SET MESSAGE TYPE = OPERATOR SELECTED INPUT
 6503                                 ;
 6504 106756  105037  003223          ACTOPR: CLRB    P$MERR                  ;CLEAR MESSAGE ERROR FLAG
 6505 106762  004737  101656                  JSR     PC,TRVADR               ;PARSE THROUGH INPUT STRING
 6506 106766  105737  003221                  TSTB    P$GDBD                  ;TEST GOOD/BAD FLAG
 6507 106772  001403                          BEQ     10$                     ;IF GOOD, BR 10$
 6508 106774  105037  003221                  CLRB    P$GDBD                  ;CLEAR FLAG
 6509 107000  000415                          BR      20$                     ;SET CTARGT FLAG AND RETURN
 6510 107002  022737  000006  003734  10$:    CMP     @OPRSEL,CFLAG           ;CHECK TO SEE IF STRING VALID
```

```
6511 107010  001011                          BNE     20$                     ;IF NOT OK, RETURN WITH ERROR FLAG SET
6512 107012  012737  000006  002372          MOV     #OPRSEL,P$TYPE          ;SET MESSAGE TYPE
6513 107020                                  CALL    SELMSG CBOADR           ;PUT OPERATOR SELECTED STRING INTO BUFFER
     107020  013725  002370                                                          MOV     CBUADR,(R5)+
     107024  004437  070656                                                          JSR     R4,PREG14
     107030  025352                                                                  .WORD   SELMSG-ANCHOR
6514 107032  000423                          BR      40$                     ;RETURN
6515 107034  022737  000000  003734  20$:    CMP     #CTARGT,CFLAG           ;SEE IF CTARGT FLAG SET, IF YES ERROR
6516 107042  001011                          BNE     30$                     ;IF NOT SET, BR 30$
6517 107044                                  PRINTF  #UNBOND                 ;ELSE PRINT UNBOUNDED INPUT STRING ERR. MSG.
     107044  012746  053756                                                          MOV     #UNBOND,-(SP)
     107050  012746  000001                                                          MOV     #1, (SP)
     107054  010600                                                                  MOV     SP,R0
     107056  104417                                                                  TRAP    C$PNTF
     107060  062706  000004                                                          ADD     #4,SP
6518 107064  000406                          BR      40$                     ;RETURN
6519 107066  105737  003223          30$:    TSTB    P$MERR                  ;IF P$MERR FLAG SET, UNBOUNDED STRING
6520 107072  001003                          BNE     40$                     ;WAS ENTERED, ERROR ALREADY HANDLED
6521 107074  112737  177777  003221          MOVB    #-1,P$GDBD              ;SET ERROR FLAG AND RETURN
6522 107102  000207                  40$:    RTS     PC                      ;RETURN
6523                                         ;
6524                                         ;ACTION ROUTINE TO CHECK FOR MORE INPUT AFTER MESSAGE TYPE HAS BEEN
6525                                         ;ALTERED
6526                                         ;
6527 107104  004737  106470          ACTTYP: JSR     PC,ACTMSG               ;CHECK FOR ADDITIONAL COMMANDS
6528 107110  000207                          RTS     PC
6529                                         ;
6530                                         ;ACTION ROUTINE TO INPUT MESSAGE SIZE PARAMETER, CHECK TO SEE IF
6531                                         ;IT IS WITHIN LEGAL LIMITS, CHANGE PARAMETER AND THEN RETURN TO
6532                                         ;SEE IF MORE INPUT EXISTA
6533                                         ;
6534 107112  023727  003214  000040  ACTSZE: CMP     P$NUM,#32.              ;CHECK FOR VALID SIZE RANGE
6535 107120  003410                          BLE     10$
6536 107122  022737  001001  003214          CMP     #513.,P$NUM
6537 107130  003404                          BLE     10$                     ;IF VALID CONTINUE
6538 107132  013737  003214  002374          MOV     P$NUM,P$SIZE            ;SET MESSAGE SIZE
6539 107140  000410                          BR      20$
6540 107142                          10$:    PRINTF  #SIZLMT                 ;PRINT SIZE LIMITS EXCEEDED MESSAGE
     107142  012746  053575                                                          MOV     #SIZLMT,-(SP)
     107146  012746  000001                                                          MOV     #1, (SP)
     107152  010600                                                                  MOV     SP,R0
     107154  104417                                                                  TRAP    C$PNTF
     107156  062706  000004                                                          ADD     #4,SP
6541 107162  004737  106470          20$:    JSR     PC,ACTMSG               ;CHECK FOR ADDITIONAL COMMANDS
6542 107166  000207                          RTS     PC
6543                                         ;
6544                                         ;ACTION ROUTINE TO INPUT COPIES PARAMETER, CHECK TO SEE IF IT IS
6545                                         ;WITHIN LEGAL LIMITS, CHANGE PARAMETER AND THEN RETURN TO SEE IF
6546                                         ;MORE INPUT PARAMETERS EXIST
6547                                         ;
6548 107170  023727  003214  000000  ACTCPY: CMP     P$NUM,#0                ;CHECK FOR VALID COPIES RANGE
6549 107176  003410                          BLE     10$
6550 107200  022737  000400  003214          CMP     #256.,P$NUM
6551 107206  003404                          BLE     10$                     ;IF VALID, CONTINUE
6552 107210  013737  003214  002376          MOV     P$NUM,P$CPYS            ;SET MESSAGE COPIES
6553 107216  000410                          BR      20$
6554 107220                          10$:    PRINTF  #CPYLMT                 ;PRINT COPY LIMIT EXCEEDED MESSAGE
```

```
        107220  012746  053511                                                  MOV     #CPYLMT,(SP)
        107224  012746  000001                                                  MOV     #1,(SP)
        107230  010600                                                          MOV     SP,R0
        107232  104417                                                          TRAP    C$PNTF
        107234  062706  000004                                                  ADD     #4,SP
  6555  107240  004737  106470          20$:    JSR     PC,ACTMSG               ;CHECK FOR ADDITIONAL COMMANDS
  6556  107244  000207                  20$:    RTS     PC
  6557                                  ;
  6558                                  ;ACTION ROUTINE TO CLEAR NODE SPECIFIED BY PHYSICAL ADDRESS FROM NODE TABLE
  6559                                  ;
  6560  107246  105037  003220          ACTNAD: CLRB    P$NNUF                  ;CLEAR NOTNUF FLAG
  6561  107252  105737  003222                  TSTB    P$AERR                  ;SEE IF ADDRESS ENTERED WAS VALID
  6562  107256  001051                          BNE     40$                     ; IF NOT, EXIT ACTION ROUTINE
  6563  107260                                  P$PUSH  R2,R3                   ;SAVE R2 AND R3
        107260  010325                                                                  MOV     R3,(R5)+
        107262  010225                                                                  MOV     R2,(R5)+
  6564  107264  012702  002316                  MOV     #ADRBUF,R2              ;MOVE ADDRESS OF ADDRESS INTO R2
  6565  107270  012703  002412                  MOV     #NODTBL,R3              ;MOVE ADDRESS OF NODE TABLE INTO R3
  6566  107274                          10$:    CALL    CMPADR R2,R3            ;SEE IF ADDRESSES MATCH
        107274  010325                                                                  MOV     R3,(R5)+
        107276  010225                                                                  MOV     R2,(R5)+
        107300  004437  070656                                                          JSR     R4,PREG14
        107304  026004                                                                  .WORD   CMPADR-ANCHOR
  6567  107306                                  P$POP   R1                                      MOV      (R5),R1
        107306  014501
  6568  107310  001416                          BEQ     20$                     ;IF YES, BR 20$
  6569  107312  062703  000010                  ADD     #10,R3                  ;ELSE POINT R3 TO NEXT ENTRY
  6570  107316  022713  177777                  CMP     #-1,(R3)                ;SEE IF END OF TABLE
  6571  107322  001364                          BNE     10$                     ;IF NOT, COMPARE NEXT ENTRY
  6572  107324                                  PRINTF  #NOCMPR                 ;ELSE, PRINT ADDRESS DOESN'T COMPARE MSG.
        107324  012746  053664                                                          MOV     #NOCMPR,-(SP)
        107330  012746  000001                                                          MOV     #1,-(SP)
        107334  010600                                                                  MOV     SP,R0
        107336  104417                                                                  TRAP    C$PNTF
        107340  062706  000004                                                          ADD     #4,SP
  6573  107344  000414                          BR      30$                     ;RETURN
  6574  107346  005023          20$:    CLR     (R3)+                   ;ELSE, CLEAR NODE FROM TABLE
  6575  107350  005023                  CLR     (R3)+
  6576  107352  005023                  CLR     (R3)+
  6577  107354  005013                  CLR     (R3)
  6578  107356                          PRINTF  #ADRDEL                 ;PRINT NODE DELETED FROM TABLE MESSAGE
        107356  012746  054044                                                  MOV     #ADRDEL,-(SP)
        107362  012746  000001                                                  MOV     #1,-(SP)
        107366  010600                                                          MOV     SP,R0
        107370  104417                                                          TRAP    C$PNTF
        107372  062706  000004                                                  ADD     #4,SP
  6579  107376                  30$:    P$POP   R2,R3                   ;RESTORE R2 AND R3
        107376  014502                                                                  MOV     -(R5),R2
        107400  014503                                                                  MOV     -(R5),R3
  6580  107402  000207                  40$:    RTS     PC                      ;RETURN
  6581                                  ;
  6582                                  ;ACTION ROUTINE TO CLEAR NODE TABLE
  6583                                  ;
  6584  107404                          ACTNAL: P$PUSH  R2,R3                   ;SAVE R2,R3
        107404  010325                                                                  MOV     R3,(R5)+
        107406  010225                                                                  MOV     R2,(R5)+
  6585  107410  012703  000050                  MOV     #TBLLEN,R3              ;SET INCR. COUNTER TO 40
```

```
 6586 107414  012702  002412                 MOV     #NODTBL,R2              ;MOVE NODE TABLE ADDRESS INTO R2
 6587 107420  005022                 10$:    CLR     (R2)+                   ;CLEAR BYTE IN NODE TABEL
 6588 107422  005303                         DEC     R3                      ;DECREMENT COUNTER
 6589 107424  001375                         BNE     10$                     ;CONTINUE UNTIL DONE
 6590 107426                                 PRINTF  #TABCLR,#NOD            ;PRINT NODE TABLE CLEARED MESSAGE
      107426  012746  053415                                                                MOV     #NOD, (SP)
      107432  012746  054220                                                                MOV     #TABCLR,-(SP)
      107436  012746  000002                                                                MOV     #2,-(SP)
      107442  010600                                                                       MOV     SP,R0
      107444  104417                                                                       TRAP    C$PNTF
      107446  062706  000006                                                                ADD     #6,SP
 6591 107452  105037  003220                 CLRB    P$NNUF                  ;CLEAR NOTNUF FLAG
 6592 107456                                 P$POP   R2,R3                   ;RESTORE R2 AND R3
      107456  014502                                                                       MOV     -(R5),R2
      107460  014503                                                                       MOV     -(R5),R3
 6593 107462  000207                         RTS     PC
 6594                                ;
 6595                                ;ACTION ROUTINE TO RUN SPECIFIED TEST
 6596                                ;
 6597 107464                         ACTRUN:
 6598 107464  042737  100000  000000         BIC     #BIT15,FLAG             ; CLEAR OPERATOR FLAG
 6599 107472  105037  003220                 CLRB    P$NNUF                  ; CLEAR 'NOT ENOUGH' FLAG
 6600 107476  013737  003214  002400         MOV     P$NUM,P$PASS
 6601 107504  022737  000032  002312 10$:    CMP     #CRNALL,KEYWD1          ; SEE IF 'ALL  TEST
 6602 107512  001004                         BNE     20$                     ;  IF NO, CONTINUE
 6603 107514                                 CALL    RUNALL                  ;  IF YES, DO ALLNODE
      107514  004437  070656                                                                JSR     R4,PREG14
      107520  016726                                                                        .WORD   RUNALL-ANCHOR
 6604 107522  000423                         BR      50$
 6605 107524  022737  000033  002312 20$:    CMP     #CLUPPR,KEYWD1          ; IS IT 'LOOPPAIR' TEST
 6606 107532  001004                         BNE     30$                     ;  IF NO, CONTINUE
 6607 107534                                 CALL    RUNLUP                  ;  IF YES, DO LOOPPAIR
      107534  004437  070656                                                                JSR     R4,PREG14
      107540  020004                                                                        .WORD   RUNLUP ANCHOR
 6608 107542  000413                         BR      50$
 6609 107544  022737  000043  002312 30$:    CMP     #CDIR,KEYWD1            ; IS IT 'DIRECT  TEST
 6610 107552  001004                         BNE     40$                     ;  IF NO, CONTINUE
 6611 107554                                 CALL    RUNDIR                  ;  IF YES, DO DIRECT
      107554  004437  070656                                                                JSR     R4,PREG14
      107560  017352                                                                        .WORD   RUNDIR ANCHOR
 6612 107562  000403                         BR      50$
 6613 107564                         40$:    CALL    RUNPAT                  ; ELSE, ITS 'PATTERN  TEST
      107564  004437  070656                                                                JSR     R4,PREG14
      107570  021446                                                                        .WORD   RUNPAT ANCHOR
 6614 107572  023727  002400  177777 50$:    CMP     P$PASS,#-1              ; SEE IF PASS SET FOR INDEFINATE
 6615 107600  001741                         BEQ     10$                     ;  IF YES, LOOP
 6616 107602  005337  002400                 DEC     P$PASS                  ; HAVE WE DONE ALL PASSES?
 6617 107606  001336                         BNE     10$                     ;  IF NO, LOOP
 6618 107610  000207                         RTS     PC
 6619                                ;
 6620                                ;ACTION ROUTINE TO SET 'RUN ALL' FLAG
 6621                                ;
 6622 107612  012737  000032  002312 ACTRNA: MOV     #CRNALL,KEYWD1          ; SET FLAG
 6623 107620  000207                 RUNALL: RTS     PC
 6624 107622                                 CALL    DIRCOM                  ; RUN LOOPDIRECT TEST
      107622  004437  070656                                                                JSR     R4,PREG14
      107626  017372                                                                        .WORD   DIRCOM ANCHOR
```

```
6625 107630                                    P$POP   R1                      ;  CHECK RESULTS
     107630  014501                                                                    MOV     (R5),R1
6626 107632  001415                            BEQ     20$                     ;  IF OK, BRANCH
6627 107634  022701  000001                    CMP     #1,R1                   ;   ELSE, WAS TABLE EMPTY?
6628 107640  001410                            BEQ     10$                     ;    IF YES, DON'T PRINT ABORT MESSAGE
6629 107642                                    PRINTS  #PASABT                 ;  ELSE ABORT TEST AND PRINT MESSAGE
     107642  012746  060672                                                            MOV     #PASABT,-(SP)
     107646  012746  000001                                                            MOV     #1,-(SP)
     107652  010600                                                                    MOV     SP,R0
     107654  104416                                                                    TRAP    C$PNTS
     107656  062706  000004                                                            ADD     #4,SP
6630 107662  000137  110234            10$:    JMP     100$
6631 107666  012737  002412  002410  20$:     MOV     #NODTBL,SLOT            ; MOVE NODE TABLE ADDRESS TO SLOT
6632 107674                                    CALL    FULSLT                  ; FIND FIRST ENTRY
     107674  004437  070656                                                            JSR     R4,PREG14
     107700  025556                                                                    .WORD   FULSLT ANCHOR
6633 107702  013701  002410                    MOV     SLOT,R1                 ;  AND PUT TARGET ADDRESS INTO R1
6634 107706  013737  002376  050326  30$:     MOV     P$CPYS,CPYCNT           ; SET UP LOOP FOR NO. OF COPIES
6635 107714  062737  000010  002410            ADD     #10,SLOT                ; UPDATE SLOT
6636 107722                                    CALL    FULSLT                  ; GET NEXT ASSIST NODE FROM TABLE
     107722  004437  070656                                                            JSR     R4,PREG14
     107726  025556                                                                    .WORD   FULSLT ANCHOR
6637 107730  013702  002410                    MOV     SLOT,R2
6638 107734  022737  177777  002410            CMP     #-1,SLOT                ;  SEE IF AT END OF TABLE
6639 107742  001517                            BEQ     90$                     ;  IF YES, BR
6640 107744                            40$:    CALL    BLDFAS  R1,SLOT         ; BUILD FULL ASSIST MESSAGE
     107744  013725  002410                                                            MOV     SLOT,(R5)+
     107750  010125                                                                    MOV     R1,(R5)+
     107752  004437  070656                                                            JSR     R4,PREG14
     107756  005502                                                                    .WORD   BLDFAS ANCHOR
6641 107760                                    CALL    XMIT    #XMTDAT         ; TRANSMIT MESSAGE
     107760  012725  000001                                                            MOV     #XMTDAT,(R5)+
     107764  004437  070656                                                            JSR     R4,PREG14
     107770  002326                                                                    .WORD   XMIT ANCHOR
6642 107772                                    P$POP   R3                      ;  CHECK RESULTS
     107772  014503                                                                    MOV     (R5),R3
6643 107774  001404                            BEQ     50$                     ;  IF OK, CONTINUE
6644 107776                                    ERRHRD  37,EMSG24               ;  PRINT ERROR MESSAGE
     107776  104456                                                                    TRAP    C$ERHRD
     110000  000045                                                                    .WORD   37
     110002  064103                                                                    .WORD   EMSG24
     110004  000000                                                                    .WORD   0
6645 110006                            50$:    CALL    BINHEX  R1,#6,#STRBUF   ;  PRINT ERROR MESSAGE
     110006  012725  002324                                                            MOV     #STRBUF,(R5)+
     110012  012725  000006                                                            MOV     #6,(R5)+
     110016  010125                                                                    MOV     R1,(R5)+
     110020  004437  070656                                                            JSR     R4,PREG14
     110024  005150                                                                    .WORD   BINHEX ANCHOR
6646 110026                                    CALL    BINHEX  R2,#6,#STRBU1   ;
     110026  012725  002346                                                            MOV     #STRBU1,(R5)+
     110032  012725  000006                                                            MOV     #6,(R5)+
     110036  010225                                                                    MOV     R2,(R5)+
     110040  004437  070656                                                            JSR     R4,PREG14
     110044  005150                                                                    .WORD   BINHEX ANCHOR
6647 110046                                    PRINTB  #TSTMS4,#ARGTY7,#STRBUF,#ARGTY6,#STRBU1 ;    ASSIST NODE =
     110046  012746  002346                                                            MOV     #STRBU1,-(SP)
     110052  012746  061747                                                            MOV     #ARGTY6,-(SP)
```

```
        110056  012746  002324                                                  MOV     @STRBUF,(SP)
        110062  012746  061757                                                  MOV     @ARGTY7,(SP)
        110066  012746  060770                                                  MOV     @TSTMS4,(SP)
        110072  012746  000005                                                  MOV     #5.(SP)
        110076  010600                                                          MOV     SP,R0
        110100  104414                                                          TRAP    C$PNTB
        110102  062706  000014                                                  ADD     #14,SP
6648    110106                          CALL    RUNCOM          ; DO RECEIVE LOOP
        110106  004437  070656                                                  JSR     R4,PREG14
        110112  021104                                                          .WORD   RUNCOM ANCHOR
6649    110114                          P$POP   R4              ; CHECK RESULTS
        110114  014504                                                          MOV     (R5),R4
6650    110116  001405                  BEQ     70$             ;  IF OK, LOOP SOME MORE
6651    110120                  60$:    ERRHRD  28,EMSG42,ERR3  ;  ELSE PRINT ERROR MESSAGE
        110120  104456                                                          TRAP    C$ERHRD
        110122  000034                                                          .WORD   28
        110124  064764                                                          .WORD   EMSG42
        110126  070432                                                          .WORD   ERR3
6652    110130  000410                  BR      80$
6653    110132                  70$:    PRINTB  #OKFU
        110132  012746  061157                                                  MOV     #OKFU,(SP)
        110136  012746  000001                                                  MOV     #1,(SP)
        110142  010600                                                          MOV     SP,R0
        110144  104414                                                          TRAP    C$PNTB
        110146  062706  000004                                                  ADD     #4,SP
6654    110152  005337  050326  80$:    DEC     CPYCNT          ; DECREMENT 'COPIES' COUNTER
6655    110156  001272                  BNE     40$             ;  IF MORE TO DO, LOOP
6656    110160                          CALL    WRITES #2,R1,SLOT       ; ELSE, UPDATE SUMMARY TABLE
        110160  013725  002410                                                  MOV     SLOT,(R5)+
        110164  010125                                                          MOV     R1,(R5)+
        110166  012725  000002                                                  MOV     #2,(R5)+
        110172  004437  070656                                                  JSR     R4,PREG14
        110176  007176                                                          .WORD   WRITES-ANCHOR
6657    110200  000642                  BR      30$
6658    110202  062701  000010  90$:    ADD     #10,R1                  ; POINT R1 TO NEXT TARGET NODE
6659    110206  010137  002410          MOV     R1,SLOT                 ; UPDATE SLOT
6660    110212                          CALL    FULSLT                  ; GET ADDRESS FROM TABLE
        110212  004437  070656                                                  JSR     R4,PREG14
        110216  025556                                                          .WORD   FULSLT-ANCHOR
6661    110220  013701  002410          MOV     SLOT,R1
6662    110224  022737  177777  002410  CMP     #-1,SLOT                ; SEE IF END OF TABLE
6663    110232  001225                  BNE     30$                     ;  IF NO, CONTINUE ELSE, FINISHED
6664    110234                  100$:   RETURN
        110234  000207                                                          RTS     PC
6665                                    ;
6666                                    ;ACTION ROUTINE TO SET  RUN LOOP DIRECT' FLAG
6667                                    ;
6668
6669    110236                  ACTDIR:
6670    110236  012737  000043  002312          MOV     #CDIR,KEYWD1            ; SET FLAG
6671    110244  000207                  RTS     PC
6672
6673    110246                  RUNDIR: CALL    DIRCOM                  ; CALL COMMON CODE
        110246  004437  070656                                                  JSR     R4,PREG14
        110252  017372                                                          .WORD   DIRCOM ANCHOR
6674    110254                          P$POP   R1
        110254  014501                                                          MOV     (R5),R1
```

```
6675 110256  022701  000001                    CMP     #1,R1                   ; WAS TABLE EMPTY?
6676 110262  001400                            BEQ     10$                     ;  IF YES, DON'T PRINT
6677 110264                             10$:    RETURN
     110264  000207                                                                    RTS     PC
6678 110266  005001                     DIRCOM: CLR     R1                      ; CLEAR RESULTS REGISTER
6679 110270  012737  002412  002410            MOV     #NODTBL,SLOT            ; MOVE NODE TABLE ADDRESS TO SLOT
6680 110276                                     CALL    FULSLT                  ; SEE IF TABLE EMPTY
     110276  004437  070656                                                            JSR     R4,PREG14
     110302  025556                                                                    .WORD   FULSLT-ANCHOR
6681 110304  022737  177777  002410            CMP     #-1,SLOT
6682 110312  001015                            BNE     10$                     ;  IF NO CONTINUE
6683 110314                                     PRINTF  #TABEMT,#NOD            ;  ELSE, PRINT "TABLE EMPTY" MESSAGE
     110314  012746  053415                                                            MOV     #NOD,(SP)
     110320  012746  053346                                                            MOV     #TABEMT,-(SP)
     110324  012746  000002                                                            MOV     #2,-(SP)
     110330  010600                                                                    MOV     SP,R0
     110332  104417                                                                    TRAP    C$PNTF
     110334  062706  000006                                                            ADD     #6,SP
6684 110340  012701  000001                    MOV     #1,R1                   ;   PUT 'TABLE EMPTY' INDICATOR IN R1
6685 110344  000547                            BR      100$
6686 110346  012737  002412  002410    10$:    MOV     #NODTBL,SLOT
6687 110354  013737  002376  050326    20$:    MOV     P$CPYS,CPYCNT           ; SET UP FOR NO. OF COPIES
6688 110362                                     CALL    FULSLT                  ; GET NEXT NODE IN TABLE
     110362  004437  070656                                                            JSR     R4,PREG14
     110366  025556                                                                    .WORD   FULSLT-ANCHOR
6689 110370  022737  177777  002410            CMP     #-1,SLOT                ; SEE IF AT END OF TABLE
6690 110376  001532                            BEQ     100$                    ;  IF YES, EXIT
6691 110400                                     CALL    BINHEX  SLOT,#6,#STRBUF ; PRINT ADDRESS BEING TESTED
     110400  012725  002324                                                            MOV     #STRBUF,(R5)+
     110404  012725  000006                                                            MOV     #6,(R5)+
     110410  013725  002410                                                            MOV     SLOT,(R5)+
     110414  004437  070656                                                            JSR     R4,PREG14
     110420  005150                                                                    .WORD   BINHEX-ANCHOR
6692 110422                             30$:    PRINTB  #TSTMS2,#DIRECT,#STRBUF ;  NODE ADDRESS
     110422  012746  002324                                                            MOV     #STRBUF,-(SP)
     110426  012746  061337                                                            MOV     #DIRECT,(SP)
     110432  012746  060735                                                            MOV     #TSTMS2,-(SP)
     110436  012746  000003                                                            MOV     #3,(SP)
     110442  010600                                                                    MOV     SP,R0
     110444  104414                                                                    TRAP    C$PNTB
     110446  062706  000010                                                            ADD     #10,SP
6693 110452  022737  000005  002312            CMP     #CPATRN,KEYWD1
6694 110460  001016                            BNE     40$
6695 110462  013701  002372                    MOV     P$TYPE,R1
6696 110466  006301                            ASL     R1
6697 110470  062701  003324                    ADD     #MSGTAB,R1
6698 110474                                     PRINTB  #MESPA1,(R1)
     110474  011146                                                                    MOV     (R1),(SP)
     110476  012746  061271                                                            MOV     #MESPA1,(SP)
     110502  012746  000002                                                            MOV     #2,(SP)
     110506  010600                                                                    MOV     SP,R0
     110510  104414                                                                    TRAP    C$PNTB
     110512  062706  000006                                                            ADD     #6,SP
6699 110516                             40$:    CALL    BLDLD   SLOT            ; CALL BUILD LOOPDIRECT SUBROUTINE
     110516  013725  002410                                                                    MOV     SLOT,(R5)+
     110522  004437  070656                                                                    JSR     R4,PREG14
     110526  005236                                                                            .WORD   BLDLD ANCHOR
```

```
                                                        CALL    XMIT    @XMTDAT         ; TRANSMIT LOOPDIRECT MESSAGES
                                                                                                MOV     @XMTDAT,(R5)+
                                                                                                JSR     R4,PREG14
                                                                                                .WORD   XMIT-ANCHOR
                                                        P$POP   R2                      ; GET RESULTS, R2 = SUCCESS/FAILURE
                                                                                                MOV     (R5),R2
                                                        BEQ     60$                     ; IF OK, EXIT
                                                50$:    ERRHRD  26,EMSG24               ;  ELSE PRINT ERROR MESSAGE
                                                                                                TRAP    C$ERHRD
                                                                                                .WORD   26
                                                                                                .WORD   EMSG24
                                                                                                .WORD   0
                                                        BR      20$
                                                60$:    CALL    RUNCOM                  ; DO RECEIVE LOOP
                                                                                                JSR     R4,PREG14
                                                                                                .WORD   RUNCOM-ANCHOR
                                                        P$POP   R4                      ; GET RESULTS
                                                                                                MOV     (R5),R4
                                                        BEQ     70$                     ;  IF NO ERRORS, CONTINUE
                                                        ERRHRD  27,EMSG34,ERR2
                                                                                                TRAP    C$ERHRD
                                                                                                .WORD   27
                                                                                                .WORD   EMSG34
                                                                                                .WORD   ERR2
                                                        MOV     #1,R1                   ; PUT ERROR INDICATOR INTO R1
                                                        BR      80$
                                                70$:    PRINTB  @OK                     ; RESPONSE OK
                                                                                                MOV     @OK,(SP)
                                                                                                MOV     #1,-(SP)
                                                                                                MOV     SP,R0
                                                                                                TRAP    C$PNTB
                                                                                                ADD     #4,SP
                                                80$:    DEC     CPYCNT                  ; DECREMENT 'COPIES' COUNTER
                                                        BNE     30$                     ;  IF MORE TO DO, LOOP
                                                        CALL    WRITES #1,SLOT          ; ELSE,UPDATE SUMMARY TABLE
                                                                                                MOV     SLOT,(R5)+
                                                                                                MOV     #1,(R5)+
                                                                                                JSR     R4,PREG14
                                                                                                .WORD   WRITES ANCHOR
                                                90$:    ADD     #10,SLOT                ; INCREMENT TO NEXT NODE TABLE ENTRY
                                                        BR      20$
                                                100$:   RETURN  R1
                                                                                                MOV     R1,(R5)+
                                                                                                RTS     PC
                                                ;
                                                ;ACTION ROUTINE TO SET 'RUN LOOPPAIR' FLAG
                                                ;
                                                ACTRNL: MOV     @CLUPPR,KEYWD1          ; SET FLAG
                                                        RTS     PC

                                                RUNLUP:
                                                        CALL    CHECK
                                                                                                JSR     R4,PREG14
                                                                                                .WORD   CHECK ANCHOR
                                                        CLR     TEMP1                   ; CLEAR 'HEADER PRINTED' FLAG
                                                        MOV     @NODTBL,SLOT            ; MOVE NODE TABLE ADDRESS TO SLOT
                                                        CALL    FULSLT                  ; SEE IF TABLE EMPTY
```

```
                110720  004437  070656                                                          JSR     R4,PREG14
                110724  025556                                                                  .WORD   FULSLT ANCHOR
6729            110726  022737  177777  002410          CMP     #1,SLOT                 ;
6730            110734  001014                          BNE     9$                      ;  IF NO, CONTINUE
6731            110736                                  PRINTF  #TABEMT,#NOD            ;  ELSE, PRINT 'TABLE EMPTY" MESSAGE
                110736  012746  053415                                                          MOV     #NOD,-(SP)
                110742  012746  053346                                                          MOV     #TABEMT,-(SP)
                110746  012746  000002                                                          MOV     #2,-(SP)
                110752  010600                                                                  MOV     SP,R0
                110754  104417                                                                  TRAP    C$PNTF
                110756  062706  000006                                                          ADD     #6,SP
6732            110762  000137  111776                  JMP     30$
6733            110766  012737  002412  002410  9$:     MOV     #NODTBL,SLOT            ; MOVE NODE TABLE ADDRESS TO SLOT
6734            110774  013737  002376  050326  10$:    MOV     P$CPYS,CPYCNT           ; SET UP FOR NO. OF COPIES
6735            111002                                  CALL    FULSLT                  ; GET NEXT NODE IN TABLE
                111002  004437  070656                                                          JSR     R4,PREG14
                111006  025556                                                                  .WORD   FULSLT ANCHOR
6736            111010  022737  177777  002410          CMP     #1,SLOT                 ; SEE IF AT END OF TABLE
6737            111016  001002                          BNE     11$                     ;  IF YES, EXIT
6738            111020  000137  111776                  JMP     30$
6739            111024  013701  002410          11$:    MOV     SLOT,R1                 ; MOVE SLOT TO R1
6740            111030  126127  000007  000000          CMPB    7(R1),#CTARGT           ; SEE IF TARGET NODE
6741            111036  001422                          BEQ     18$                     ;  IF YES, BRANCH
6742            111040                          17$:    PRINTF  #EMSG32                 ;  ELSE PRINT ERROR MESSAGE
                111040  012746  064245                                                          MOV     #EMSG32,-(SP)
                111044  012746  000001                                                          MOV     #1,-(SP)
                111050  010600                                                                  MOV     SP,R0
                111052  104417                                                                  TRAP    C$PNTF
                111054  062706  000004                                                          ADD     #4,SP
6743            111060                                  PRINTF  #PASABT
                111060  012746  060672                                                          MOV     #PASABT,-(SP)
                111064  012746  000001                                                          MOV     #1,-(SP)
                111070  010600                                                                  MOV     SP,R0
                111072  104417                                                                  TRAP    C$PNTF
                111074  062706  000004                                                          ADD     #4,SP
6744            111100  000137  111776                  JMP     30$                     ;   EXIT
6745            111104  010102                  18$:    MOV     R1,R2                   ; POINT R1 TO TARGET NODE
6746            111106  062702  000010          112$:   ADD     #10,R2                  ;  AND R2 TO ASSIST NODE
6747            111112  021227  000000                  CMP     (R2),#0
6748            111116  001773                          BEQ     112$
6749            111120  022712  177777                  CMP     #-1,(R2)
6750            111124  001002                          BNE     110$
6751            111126  000137  111776                  JMP     30$
6752            111132  126227  000007  000001  110$:   CMPB    7(R2),#CASIST           ; IS R2 POINTING TO AN ASSIST NODE?
6753            111140  001337                          BNE     17$                     ;  IF NOT, ERROR, ELSE CONTINUE
6754            111142                          20$:    CALL    BLDAST  R2,R1           ; BUILD TRANSMIT ASSIST MESSAGE
                111142  010125                                                                  MOV     R1,(R5)+
                111144  010225                                                                  MOV     R2,(R5)+
                111146  004437  070656                                                          JSR     R4,PREG14
                111152  006030                                                                  .WORD   BLDAST-ANCHOR
6755            111154                                  CALL    XMIT                    ; TRANSMIT MESSAGE
                111154  004437  070656                                                          JSR     R4,PREG14
                111160  002326                                                                  .WORD   XMIT ANCHOR
6756            111162                                  P$POP   R4                      ; GET RESULTS, R2 = SUCCESS/FAILURE
                111162  014504                                                                  MOV     (R5),R4
6757            111164  001406                          BEQ     22$                     ; IF OK, EXIT
6758            111166                          21$:    ERRHRD  26,EMSG24               ;  ELSE PRINT ERROR MESSAGE
```

```
                111166  104456                                                          TRAP    C$ERHRD
                111170  000032                                                          .WORD   26
                111172  064103                                                          .WORD   EMSG24
                111174  000000                                                          .WORD   0
 6759 111176    000137  111712                  JMP     28$
 6760 111202                            22$:    CALL    BINHEX  R1,06,@STRBUF    ;  PRINT ERROR MESSAGE
                111202  012725  002324                                                  MOV     @STRBUF,(R5)+
                111206  012725  000006                                                  MOV     #6,(R5)+
                111212  010125                                                          MOV     R1,(R5)+
                111214  004437  070656                                                  JSR     R4,PREG14
                111220  005150                                                          .WORD   BINHEX ANCHOR
 6761 111222                            CALL    B_NHEX  R2,06,@STRBU1    ;
                111222  012725  002346                                                  MOV     @STRBU1,(R5)+
                111226  012725  000006                                                  MOV     #6,(R5)+
                111232  010225                                                          MOV     R2,(R5)+
                111234  004437  070656                                                  JSR     R4,PREG14
                111240  005150                                                          .WORD   BINHEX-ANCHOR
 6762 111242                            PRINTB  @TSTMS4,@ARGTY7,@STRBUF,@ARGTY6,@STRBU1 ;   ASSIST NODE =
                111242  012746  002346                                                  MOV     @STRBU1,-(SP)
                111246  012746  061747                                                  MOV     @ARGTY6,-(SP)
                111252  012746  002324                                                  MOV     @STRBUF,-(SP)
                111256  012746  061757                                                  MOV     @ARGTY7,-(SP)
                111262  012746  060770                                                  MOV     @TSTMS4,-(SP)
                111266  012746  000005                                                  MOV     #5,-(SP)
                111272  010600                                                          MOV     SP,R0
                111274  104414                                                          TRAP    C$PNTB
                111276  062706  000014                                                  ADD     #14,SP
 6763 111302                            CALL    RUNCOM                  ; DO RECIEVE LOOP
                111302  004437  070656                                                  JSR     R4,PREG14
                111306  021104                                                          .WORD   RUNCOM ANCHOR
 6764 111310                            P$POP   R3                      ; CHECK RESULTS
                111310  014503                                                          MOV     -(R5),R3
 6765 111312    001405                  BEQ     23$                     ;  IF OK, CONT.
 6766 111314                            ERRHRD  28,EMSG40,ERR3
                111314  104456                                                          TRAP    C$ERHRD
                111316  000034                                                          .WORD   28
                111320  064561                                                          .WORD   EMSG40
                111322  070432                                                          .WORD   ERR3
 6767 111324    000410                  BR      101$
 6768 111326                            23$:    PRINTB  #OKRE
                111326  012746  061046                                                  MOV     #OKRE,-(SP)
                111332  012746  000001                                                  MOV     #1,-(SP)
                111336  010600                                                          MOV     SP,R0
                111340  104414                                                          TRAP    C$PNTB
                111342  062706  000004                                                  ADD     #4,SP
 6769 111346                            101$:   CALL    BLDAST  R1,R2           ; BUILD RECEIVE ASSIST MESSAGE
                111346  010225                                                          MOV     R2,(R5)+
                111350  010125                                                          MOV     R1,(R5)+
                111352  004437  070656                                                  JSR     R4,PREG14
                111356  006030                                                          .WORD   BLDAST ANCHOR
 6770 111360                            CALL    XMIT                    ; TRANSMIT MESSAGE
                111360  004437  070656                                                  JSR     R4,PREG14
                111364  002326                                                          .WORD   XMIT ANCHOR
 6771 111366                            P$POP   R4                      ; CHECK RESULTS
                111366  014504                                                          MOV     -(R5),R4
 6772 111370    001276                  BNE     21$                     ; IF OK CONTINUE, ELSE REPORT ERROR
 6773 111372                            CALL    BINHEX  R1,06,@STRBUF   ;  PRINT ERROR MESSAGE
```

```
                111372  012725  002324                                                          MOV     #STRBUF,(R5)+
                111376  012725  000006                                                          MOV     #6,(R5)+
                111402  010125                                                                  MOV     R1,(R5)+
                111404  004437  070656                                                          JSR     R4,PREG14
                111410  005150                                                                  .WORD   BINHEX-ANCHOR
6774            111412                  CALL    BINHEX  R2,#6,#STRBU1   ;
                111412  012725  002346                                                          MOV     #STRBU1,(R5)+
                111416  012725  000006                                                          MOV     #6,(R5)+
                111422  010225                                                                  MOV     R2,(R5)+
                111424  004437  070656                                                          JSR     R4,PREG14
                111430  005150                                                                  .WORD   BINHEX-ANCHOR
6775            111432                  PRINTB  #TSTMS4,#ARGTY7,#STRBUF,#ARGTY6,#STRBU1 ;    ASSIST NODE *
                111432  012746  002346                                                                  MOV     #STRBU1,-(SP)
                111436  012746  061747                                                                  MOV     #ARGTY6,-(SP)
                111442  012746  002324                                                                  MOV     #STRBUF,-(SP)
                111446  012746  061757                                                                  MOV     #ARGTY7,-(SP)
                111452  012746  060770                                                                  MOV     #TSTMS4,-(SP)
                111456  012746  000005                                                                  MOV     #5,-(SP)
                111462  010600                                                                          MOV     SP,R0
                111464  104414                                                                          TRAP    C$PNTB
                111466  062706  000014                                                                  ADD     #14,SP
6776            111472                  CALL    RUNCOM                  ; DO RECEIVE LOOP
                111472  004437  070656                                                                  JSR     R4,PREG14
                111476  021104                                                                          .WORD   RUNCOM-ANCHOR
6777            111500                  P$POP   R3                      ;  GET RESULTS
                111500  014503                                                                          MOV     (R5),R3
6778            111502  001405                  BEQ     25$             ;  IF OK, CONT.
6779            111504                  ERRHRD  28,EMSG41,ERR3
                111504  104456                                                                          TRAP    C$ERHRD
                111506  000034                                                                          .WORD   28
                111510  064662                                                                          .WORD   EMSG41
                111512  070432                                                                          .WORD   ERR3
6780            111514  000410                  BR      102$
6781            111516          25$:    PRINTB  #OKTR
                111516  012746  061112                                                                  MOV     #OKTR,(SP)
                111522  012746  000001                                                                  MOV     #1,-(SP)
                111526  010600                                                                          MOV     SP,R0
                111530  104414                                                                          TRAP    C$PNTB
                111532  062706  000004                                                                  ADD     #4,SP
6782            111536          102$:   CALL    BLDFAS  R1,R2           ; BUILD FULL ASSIST MESSAGE
                111536  010225                                                                          MOV     R2,(R5)+
                111540  010125                                                                          MOV     R1,(R5)+
                111542  004437  070656                                                                  JSR     R4,PREG14
                111546  005502                                                                          .WORD   BLDFAS-ANCHOR
6783            111550                  CALL    XMIT                    ; TRANSMIT MESSAGE
                111550  004437  070656                                                                  JSR     R4,PREG14
                111554  002326                                                                          .WORD   XMIT-ANCHOR
6784            111556                  P$POP   R4                      ; CHECK RESULTS
                111556  014504                                                                          MOV     -(R5),R4
6785            111560  001402                  BEQ     26$             ; IF OK CONTINUE, ELSE REPORT ERROR
6786            111562  000137  111166        JMP     21$
6787            111566          26$:    CALL    BINHEX  R1,#6,#STRBUF   ;  PRINT ERROR MESSAGE
                111566  012725  002324                                                                  MOV     #STRBUF,(R5)+
                111572  012725  000006                                                                  MOV     #6,(R5)+
                111576  010125                                                                          MOV     R1,(R5)+
                111600  004437  070656                                                                  JSR     R4,PREG14
                111604  005150                                                                          .WORD   BINHEX ANCHOR
```

```
 6788 111606                            CALL    BINHEX  R2,#6,#STRBU1   ;
      111606  012725  002346                                                    MOV     #STRBU1,(R5)+
      111612  012725  000006                                                    MOV     #6,(R5)+
      111616  010225                                                            MOV     R2,(R5)+
      111620  004437  070656                                                    JSR     R4,PREG14
      111624  005150                                                           .WORD   BINHEX-ANCHOR
 6789 111626                            PRINTB  #TSTMS4,#ARGTY7,#STRBUF,#ARGTY6,#STRBU1 ;   ASSIST NODE =
      111626  012746  002346                                                    MOV     #STRBU1,-(SP)
      111632  012746  061747                                                    MOV     #ARGTY6,-(SP)
      111636  012746  002324                                                    MOV     #STRBUF,-(SP)
      111642  012746  061757                                                    MOV     #ARGTY7,-(SP)
      111646  012746  060770                                                    MOV     #TSTMS4,-(SP)
      111652  012746  000005                                                    MOV     #5,(SP)
      111656  010600                                                            MOV     SP,R0
      111660  104414                                                            TRAP    C$PNTB
      111662  062706  000014                                                    ADD     #14,SP
 6790 111666                            CALL    RUNCOM                  ; DO RECEIVE LOOP
      111666  004437  070656                                                            JSR     R4,PREG14
      111672  021104                                                                   .WORD   RUNCOM-ANCHOR
 6791 111674                            P$POP   R3                      ; CHECK RESULTS
      111674  014503                                                                    MOV     (R5),R3
 6792 111676  001405                    BEQ     28$                     ;  IF NO ERRORS, CONT
 6793 111700                            ERRHRD  28,EMSG42,ERR3
      111700  104456                                                            TRAP    C$ERHRD
      111702  000034                                                           .WORD   28
      111704  064764                                                           .WORD   EMSG42
      111706  070432                                                           .WORD   ERR3
 6794 111710  000410                    BR      103$
 6795 111712                    28$:    PRINTB  #OKFU
      111712  012746  061157                                                    MOV     #OKFU,(SP)
      111716  012746  000001                                                    MOV     #1,-(SP)
      111722  010600                                                            MOV     SP,R0
      111724  104414                                                            TRAP    C$PNTB
      111726  062706  000004                                                    ADD     #4,SP
 6796 111732  005337  050326    103$:   DEC     CPYCNT                  ; DECREMENT 'COPIES  COUNTER
 6797 111736  001402                    BEQ     29$                     ;  IF MORE TO DO, LOOP
 6798 111740  000137  111142            JMP     20$
 6799 111744                    29$:    CALL    WRITES #2,R1,R2         ; ELSE,UPDATE SUMMARY TABLE
      111744  010225                                                                    MOV     R2,(R5)+
      111746  010125                                                                    MOV     R1,(R5)+
      111750  012725  000002                                                            MOV     #2,(R5)+
      111754  004437  070656                                                            JSR     R4,PREG14
      111760  007176                                                                   .WORD   WRITES-ANCHOR
 6800 111762  062702  000010            ADD     #10,R2
 6801 111766  010237  002410            MOV     R2,SLOT
 6802 111772  000137  110774            JMP     10$
 6803 111776                    30$:    RETURN
      111776  000207                                                                    RTS     PC
 6804 112000  005737  050274    RUNCOM: TST     RETRYS                  ; SEE IF FAILED DUE TO EXCESSIVE COLLISIONS
 6805 112004  001120                    BNE     30$                     ;  IF YES, BR, ELSE CONT.
 6806 112006  012704  003762            MOV     #TIMERS,R4              ; SET UP FOR 10 SECOND TIMOUT
 6807 112012  012714  000012            MOV     #10.,(R4)
 6808 112016                    10$:    BREAK
      112016  104422                                                            TRAP    C$BRK
 6809 112020  005714                    TST     (R4)                    ; SEE IF TIME HAS EXPIRED
 6810 112022  001517                    BEQ     40$                     ;  IF YES, BRANCH
 6811 112024                            CALL    RECEVE                  ; CHECK FOR ANSWER
```

```
        112024  004437  070656                                                  JSR     R4,PREG14
        112030  003044                                                          .WORD   RECEVE-ANCHOR
  6812  112032                          P$POP   R1                      ; R2 HOLDS NO. OF BUFFERS RECEIVED
        112032  014501                                                          MOV     -(R5),R1
  6813  112034  001770                  BEQ     10$                     ; IF NO BUFFERS RECEIVED, LOOP
  6814  112036  063737  050324  050042  ADD     XFER,S.XFER             ; UPDATE BYTES TRANSFERED SUM. COUNTER
  6815  112044  005237  050030          INC     S.REC                   ; UPDATE PACKETS RECEIVED SUM. COUNTER
  6816  112050  013703  024226          MOV     RRGNXT,R3               ; GET RECEIVE RING POINTER
  6817  112054  116301  000012          MOVB    STAT2(R3),R1            ; GET RCV BYTE COUNT BITS 0 7
  6818  112060  116302  000011          MOVB    STAT1+1(R3),R2          ; GET RCV BYTE COUNT BITS 8-10
  6819  112064  042701  177400          BIC     #177400,R1              ; CLEAR ALL UNWANTED BITS
  6820  112070  042702  000370          BIC     #370,R2                 ; CLEAR ALL UNWANTED BITS
  6821  112074  000302                  SWAB    R2                      ; PUT 8-10 INTO HIGH BYTE
  6822  112076  050201                  BIS     R2,R1                   ; MAKE BYTE COUNT COMPLETE
  6823  112100  062701  000074          ADD     #60.,R1                 ; ADD BYTES LOST WHILE COMPARING ADDRESSES
  6824  112104  020137  050332          CMP     R1,BUFLEN               ; CHECK FOR LENGTH ERROR
  6825  112110  001423                  BEQ     20$                     ;  IF OK, BR
  6826  112112  005237  050034          INC     S.LEN                   ;  ELSE, UPDATE LENGTH ERRORS COUNTER
  6827  112116  012737  061506  002314  MOV     #LENGTH,KEYWD2          ;  MOVE 'LENGTH' TO ERROR INDICATOR
  6828  112124  012702  177777          MOV     #-1,R2                  ;  INDICATE ERROR TO R2
  6829  112130                          PRINTX  #LGERMS,BUFLEN,R1       ;  PRINT LENGTH ERROR MESSAGE
        112130  010146                                                          MOV     R1,-(SP)
        112132  013746  050332                                                  MOV     BUFLEN,-(SP)
        112136  012746  067745                                                  MOV     #LGERMS,-(SP)
        112142  012746  000003                                                  MOV     #3.,(SP)
        112146  010600                                                          MOV     SP,R0
        112150  104415                                                          TRAP    C$PNTX
        112152  062706  000010                                                  ADD     #10,SP
  6830  112156  000452                  BR      60$                     ;  AND EXIT
  6831  112160  016303  000004  20$:    MOV     LOADD(R3),R3            ; POINT R3 TO MESSAGE BUFFER
  6832  112164  066303  000016          ADD     16(R3),R3               ; POINT R3 TO DATA AFTER SKIP COUNT
  6833  112170  062703  000030          ADD     #30,R3                  ; POINT R3 TO FIRST DATA BYTE
  6834  112174  063737  002374  050040  ADD     P$SIZE,S.BYTE           ; UPDATE BYTES COMPARED SUMMARY COUNTER
  6835  112202                          CALL    DATCMP P$SIZE,CMPBUF,R3 ; CHECK FOR DATA COMPARE ERRORS
        112202  010325                                                          MOV     R3,(R5)+
        112204  013725  050334                                                  MOV     CMPBUF,(R5)+
        112210  013725  002374                                                  MOV     P$SIZE,(R5)+
        112214  004437  070656                                                  JSR     R4,PREG14
        112220  007040                                                          .WORD   DATCMP ANCHOR
  6836  112222                          P$POP   R3                      ; CHECK RESULTS
        112222  014503                                                          MOV     -(R5),R3
  6837  112224  001426                  BEQ     50$                     ; BRANCH IF NO ERRORS
  6838  112226  060337  050036          ADD     R3,S.COMP               ;   UPDATE COMPARE ERRORS SUMMARY COUNTER
  6839  112232  012737  061515  002314  MOV     #COMPAR,KEYWD2          ;   MOVE 'COMPARE' TO ERROR INDICATOR
  6840  112240  012702  177777          MOV     #-1,R2                  ;   INDICATE ERROR TO R2
  6841  112244  000417                  BR      60$
  6842
  6843  112246  012737  061462  002314  30$:    MOV     #RETRY,KEYWD2           ; MOVE 'EXCESSIVE COLLISIONS' TO ERROR INCICATOR
  6844  112254  012702  177777          MOV     #-1,R2                  ;  INDICATE ERROR IN R2
  6845  112260  000411                  BR      60$
  6846
  6847  112262  005237  050032  40$:    INC     S.NREC                  ; UPDATE MESSAGES NOT RECEIVED COUNTER
  6848  112266  012737  061446  002314  MOV     #NORESP,KEYWD2          ; MOVE 'NO RESPONCE' TO ERROR INDICATOR
  6849  112274  012702  177777          MOV     #-1,R2                  ; INDICATE ERROR TO R2
  6850  112300  000412                  BR      65$                     ; LEAVE
  6851  112302                  50$:
  6852  112302  005002                  CLR     R2                      ; INDICATE SUCCESS
```

```
6853 112304                            60$:
6854 112304  013703  024226                   MOV     RRGNXT,R3               ; GET DESCRIPTOR
6855 112310                                   CALL    GETRNX  #1,@RRGNXT      ; UPDATE POINTER, VALIDATE PREVIOUS DESCRIPTOR
     112310  012725  024226                                                           MOV     @RRGNXT,(R5)+
     112314  012725  000001                                                           MOV     #1,(R5)+
     112320  004437  070656                                                           JSR     R4,PREG14
     112324  006450                                                                   .WORD   GETRNX ANCHOR
6856 112326                            65$:
6857 112326                                   RETURN  R2                      ; RETURN
     112326  010225                                                           MOV     R2,(R5)+
     112330  000207                                                           RTS     PC
6858
6859                                   ;
6860                                   ;ACTION ROUTINE TO SET 'RUN PATTERN' FLAG
6861                                   ;
6862
6863 112332  012737  000005  002312   ACTPAT: MOV     #CPATRN,KEYWD1          ;SET FLAG
6864 112340  000207                           RTS     PC
6865
6866
6867 112342                            RUNPAT: P$PUSH  P$TYPE                  ; SAVE TYPE PARAMETER
     112342  013725  002372                                                           MOV     P$TYPE,(R5)+
6868 112346  005037  002372                   CLR     P$TYPE                  ; SET TYPE TO FIRST TYPE
6869 112352                            10$:    CALL    DIRCOM                  ; SEND MESSAGES
     112352  004437  070656                                                           JSR     R4,PREG14
     112356  017372                                                                   .WORD   DIRCOM ANCHOR
6870 112360                                   P$POP   R1                      ; GET RESULTS TO KEEP STACK IN ORDER
     112360  014501                                                           MOV     (R5),R1
6871 112362  001403                           BEQ     20$                     ; IF OK, CONT
6872 112364  022701  000001                   CMP     #1,R1                   ;  ELSE, WAS TABLE EMPTY
6873 112370  001406                           BEQ     30$                     ;  IF YES, RETURN
6874 112372  005237  002372           20$:    INC     P$TYPE                  ; SET TO NEXT TYPE
6875 112376  022737  000005  002372         CMP     #5,P$TYPE               ; SEE IF DONE ALL OF THEM
6876 112404  002362                           BGE     10$                     ;  IF NOT, DO MORE
6877 112406                            30$:    P$POP   P$TYPE                  ; RESTORE MESSAGE TYPE
     112406  014537  002372                                                           MOV     (R5),P$TYPE
6878 112412                                   RETURN
     112412  000207                                                           RTS     PC
6879
6880                                   ;
6881                                   ;ACTION ROUTINE TO SHOW THE CURRENT MESSAGE PARAMETERS
6882                                   ;
6883
6884 112414  013701  002372           ACTSMS: MOV     P$TYPE,R1               ;GET MESSAGE TYPE INTO R1
6885 112420  006301                           ASL     R1                      ;MULTIPLY BY 2
6886 112422  062701  003324                   ADD     #MSGTAB,R1              ;ADD MESSAGE TABLE OFFSET
6887 112426                                   PRINTF  #MSGPRM                 ;PRINT MESSAGE PARAMETER MESSAGE
     112426  012746  054345                                                                   MOV     #MSGPRM, (SP)
     112432  012746  000001                                                                   MOV     #1, (SP)
     112436  010600                                                                           MOV     SP,R0
     112440  104417                                                                           TRAP    C$PNTF
     112442  062706  000004                                                                   ADD     #4,SP
6888 112446                                   PRINTF  #MSG4,(R1),P$SIZE,P$CPYS        ;PRINT PARAMETERS
     112446  013746  002376                                                                   MOV     P$CPYS,-(SP)
     112452  013746  002374                                                                   MOV     P$SIZE,-(SP)
     112456  011146                                                                           MOV     (R1),-(SP)
     112460  012746  055002                                                                   MOV     #MSG4,-(SP)
```

```
                                                                                        MOV     #4.,(SP)
                                                                                        MOV     SP,R0
                                                                                        TRAP    C$PNTF
                                                                                        ADD     #12,SP
 6889 112500  105037  003220                CLRB    P$NNUF
 6890 112504  000207                        RTS     PC
 6891
 6892                                ;
 6893                                ;ACTION ROUTINE TO CLEAR THE CURRENT MESSAGE PARAMETERS AND
 6894                                ;RESET THEM TO THE DEFAULT VALUE
 6895                                ;
 6896
 6897 112506  012737  000000 002372 ACTCMS: MOV     #ALPHA,P$TYPE           ;RESET TYPE
 6898 112514  012737  001000 002374         MOV     #512.,P$SIZE            ;RESET SIZE
 6899 112522  012737  000001 002376         MOV     #1,P$CPYS               ;RESET COPIES
 6900 112530                                PRINTF  #CLRMSG                 ;PRINT MESSAGE PARAMETERS RESET MESSAGE
                                                                                        MOV     #CLRMSG,-(SP)
                                                                                        MOV     #1,-(SP)
                                                                                        MOV     SP,R0
                                                                                        TRAP    C$PNTF
                                                                                        ADD     #4,SP
 6901 112550                                PRINTF  #MSG4,MSGTAB,P$SIZE,P$CPYS      ;PRINT PARAMETERS
                                                                                        MOV     P$CPYS,-(SP)
                                                                                        MOV     P$SIZE,-(SP)
                                                                                        MOV     MSGTAB,-(SP)
                                                                                        MOV     #MSG4,-(SP)
                                                                                        MOV     #4,-(SP)
                                                                                        MOV     SP,R0
                                                                                        TRAP    C$PNTF
                                                                                        ADD     #12,SP
 6902 112604  105037  003220                CLRB    P$NNUF                  ;CLEAR NOTNUF FLAG
 6903 112610  000207                        RTS     PC
 6904
 6905
 6906                                ;
 6907                                ;ACTION ROUTINE TO SET SHOW COUNTERS FLAG
 6908                                ;
 6909
 6910 112612                        ACTCNT:
 6911 112612                                P$PUSH  R1,R2,R3                ; SAVE REGISTERS
                                                                                        MOV     R3 (R5)+
                                                                                        MOV     R2,(R5)+
                                                                                        MOV     R1,(R5)+
 6912 112620  012701  050144                MOV     #CNTRS,R1               ; GET ADRESS OF COUNTERS
 6913 112624  012702  050044                MOV     #UCB12,R2               ; GET ADDRESS OF COUNTER BUFFER
 6914 112630  016103  000000                MOV     C.SIZ(R1),R3            ; GET SIZE OF BUFFER
 6915 112634                        10$:
 6916 112634  112122                        MOVB    (R1)+,(R2)+             ; MOV COUNTERS TO BUFFER
 6917 112636  005303                        DEC     R3                      ; SUBTRACT ONE FROM THE COUNT
 6918 112640  001375                        BNE     10$                     ; BRANCH UNTIL ALL ARE DONE
 6919 112642                                P$POP   R1,R2,R3                ; RESTORE THE REGISTERS
                                                                                        MOV     -(R5),R1
                                                                                        MOV     -(R5),R2
                                                                                        MOV     -(R5),R3
 6920 112650                                CALL    BINHEX  #PHYADR,#6,#STRBUF      ;GET ADDRESS INTO ASCII
                                                                                        MOV     #STRBUF,(R5)+
                                                                                        MOV     #6,(R5)+
```

```
        112660  012725  004010                                  MOV     @PHYADR,(R5)+
        112664  004437  070656                                  JSR     R4,PREG14
        112670  005150                                          .WORD   BINHEX-ANCHOR
   6921 112672                          PRINTF  @CNTR00,@STRBUF
        112672  012746  002324                                          MOV     @STRBUF,-(SP)
        112676  012746  062047                                          MOV     @CNTR00,-(SP)
        112702  012746  000002                                          MOV     @2,-(SP)
        112706  010600                                                  MOV     SP,R0
        112710  104417                                                  TRAP    C$PNTF
        112712  062706  000006                                          ADD     @6,SP
   6922 112716                          PRINTF  @CNTR01,UCB12+2
        112716  013746  050046                                          MOV     UCB12+2,-(SP)
        112722  012746  062127                                          MOV     @CNTR01,(SP)
        112726  012746  000002                                          MOV     @2,-(SP)
        112732  010600                                                  MOV     SP,R0
        112734  104417                                                  TRAP    C$PNTF
        112736  062706  000006                                          ADD     @6,SP
   6923 112742                          CALL    BINDEC  @UCB12+4
        112742  012725  050050                                  MOV     @UCB12+4,(R5)+
        112746  004437  070656                                  JSR     R4,PREG14
        112752  007446                                          .WORD   BINDEC-ANCHOR
   6924 112754                          PRINTF  @CNTR02,@DECSTR
        112754  012746  100526                                          MOV     @DECSTR,-(SP)
        112760  012746  062176                                          MOV     @CNTR02,-(SP)
        112764  012746  000002                                          MOV     @2,-(SP)
        112770  010600                                                  MOV     SP,R0
        112772  104417                                                  TRAP    C$PNTF
        112774  062706  000006                                          ADD     @6,SP
   6925 113000                          CALL    BINDEC  @UCB12+10
        113000  012725  050054                                  MOV     @UCB12+10,(R5)+
        113004  004437  070656                                  JSR     R4,PREG14
        113010  007446                                          .WORD   BINDEC-ANCHOR
   6926 113012                          PRINTF  @CNTR03,@DECSTR
        113012  012746  100526                                          MOV     @DECSTR,-(SP)
        113016  012746  062232                                          MOV     @CNTR03,-(SP)
        113022  012746  000002                                          MOV     @2,-(SP)
        113026  010600                                                  MOV     SP,R0
        113030  104417                                                  TRAP    C$PNTF
        113032  062706  000006                                          ADD     @6,SP
   6927 113036                          PRINTF  @CNTR04,UCB12+14
        113036  013746  050060                                          MOV     UCB12+14,-(SP)
        113042  012746  062277                                          MOV     @CNTR04,-(SP)
        113046  012746  000002                                          MOV     @2,-(SP)
        113052  010600                                                  MOV     SP,R0
        113054  104417                                                  TRAP    C$PNTF
        113056  062706  000006                                          ADD     @6,SP
   6928 113062                          PRINTF  @CNTR05,UCB12+16
        113062  013746  050062                                          MOV     UCB12+16,-(SP)
        113066  012746  062354                                          MOV     @CNTR05,-(SP)
        113072  012746  000002                                          MOV     @2,-(SP)
        113076  010600                                                  MOV     SP,R0
        113100  104417                                                  TRAP    C$PNTF
        113102  062706  000006                                          ADD     @6,SP
   6929 113106                          CALL    BINDEC  @UCB12+20
        113106  012725  050064                                  MOV     @UCB12+20,(R5)+
        113112  004437  070656                                  JSR     R4,PREG14
        113116  007446                                          .WORD   BINDEC-ANCHOR
```

```
6930 113120                                    PRINTF  @CNTR06,@DECSTR
     113120  012746  100526                                    MOV     @DECSTR,(SP)
     113124  012746  062424                                    MOV     @CNTR06,-(SP)
     113130  012746  000002                                    MOV     #2,(SP)
     113134  010600                                            MOV     SP,R0
     113136  104417                                            TRAP    C$PNTF
     113140  062706  000006                                    ADD     #6,SP
6931 113144                                    CALL    BINDEC  @UCB12.24
     113144  012725  050070                            MOV     @UCB12.24,(R5)+
     113150  004437  070656                            JSR     R4,PREG14
     113154  007446                                    .WORD   BINDEC-ANCHOR
6932 113156                                    PRINTF  @CNTR07,@DECSTR
     113156  012746  100526                                    MOV     @DECSTR,-(SP)
     113162  012746  062463                                    MOV     @CNTR07,-(SP)
     113166  012746  000002                                    MOV     #2,-(SP)
     113172  010600                                            MOV     SP,R0
     113174  104417                                            TRAP    C$PNTF
     113176  062706  000006                                    ADD     #6,SP
6933 113202                                    PRINTF  @CNTR08,UCB12.30
     113202  013746  050074                                    MOV     UCB12.30,-(SP)
     113206  012746  062533                                    MOV     @CNTR08,(SP)
     113212  012746  000002                                    MOV     #2,(SP)
     113216  010600                                            MOV     SP,R0
     113220  104417                                            TRAP    C$PNTF
     113222  062706  000006                                    ADD     #6,SP
6934 113226                                    PRINTF  @CNTR09,UCB12.32
     113226  013746  050076                                    MOV     UCB12.32,(SP)
     113232  012746  062606                                    MOV     @CNTR09,(SP)
     113236  012746  000002                                    MOV     #2,(SP)
     113242  010600                                            MOV     SP,R0
     113244  104417                                            TRAP    C$PNTF
     113246  062706  000006                                    ADD     #6,SP
6935 113252                                    CALL    BINDEC  @UCB12.34
     113252  012725  050100                            MOV     @UCB12.34,(R5)+
     113256  004437  070656                            JSR     R4,PREG14
     113262  007446                                    .WORD   BINDEC ANCHOR
6936 113264                                    PRINTF  @CNTR10,@DECSTR
     113264  012746  100526                                    MOV     @DECSTR,(SP)
     113270  012746  062657                                    MOV     @CNTR10,(SP)
     113274  012746  000002                                    MOV     #2,-(SP)
     113300  010600                                            MOV     SP,R0
     113302  104417                                            TRAP    C$PNTF
     113304  062706  000006                                    ADD     #6,SP
6937 113310                                    CALL    BINDEC  @UCB12.40
     113310  012725  050104                            MOV     @UCB12.40,(R5)+
     113314  004437  070656                            JSR     R4,PREG14
     113320  007446                                    .WORD   BINDEC-ANCHOR
6938 113322                                    PRINTF  @CNTR11,@DECSTR
     113322  012746  100526                                    MOV     @DECSTR,(SP)
     113326  012746  062716                                    MOV     @CNTR11,(SP)
     113332  012746  000002                                    MOV     #2,(SP)
     113336  010600                                            MOV     SP,R0
     113340  104417                                            TRAP    C$PNTF
     113342  062706  000006                                    ADD     #6,SP
6939 113346                                    CALL    BINDEC  @UCB12.44
     113346  012725  050110                            MOV     @UCB12.44,(R5)+
     113352  004437  070656                            JSR     R4,PREG14
```

```
                113356  007446                  PRINTF  @CNTR12,@DECSTR         .WORD   BINDEC-ANCHOR
                                                                                
                                                                                        MOV     @DECSTR,-(SP)
                                                                                        MOV     @CNTR12,-(SP)
                                                                                        MOV     @2,-(SP)
                                                                                        MOV     SP,R0
                                                                                        TRAP    C$PNTF
                                                                                        ADD     @6,SP
     6941       113404                          CALL    BINDEC  @UCB12+50       MOV     @UCB12+50,(R5)+
                113404  012725  050114                                          JSR     R4,PREG14
                113410  004437  070656                                          .WORD   BINDEC-ANCHOR
                113414  007446
     6942       113416                          PRINTF  @CNTR13,@DECSTR
                113416  012746  100526                                                  MOV     @DECSTR,-(SP)
                113422  012746  063034                                                  MOV     @CNTR13,-(SP)
                113426  012746  000002                                                  MOV     @2,-(SP)
                113432  010600                                                          MOV     SP,R0
                113434  104417                                                          TRAP    C$PNTF
                113436  062706  000006                                                  ADD     @6,SP
     6943       113442                          CALL    BINDEC  @UCB12+54       MOV     @UCB12+54,(R5)+
                113442  012725  050120                                          JSR     R4,PREG14
                113446  004437  070656                                          .WORD   BINDEC-ANCHOR
                113452  007446
     6944       113454                          PRINTF  @CNTR14,@DECSTR
                113454  012746  100526                                                  MOV     @DECSTR,-(SP)
                113460  012746  063101                                                  MOV     @CNTR14,-(SP)
                113464  012746  000002                                                  MOV     @2,-(SP)
                113470  010600                                                          MOV     SP,R0
                113472  104417                                                          TRAP    C$PNTF
                113474  062706  000006                                                  ADD     @6,SP
     6945       113500                          CALL    BINDEC  @UCB12+60       MOV     @UCB12+60,(R5)+
                113500  012725  050124                                          JSR     R4,PREG14
                113504  004437  070656                                          .WORD   BINDEC-ANCHOR
                113510  007446
     6946       113512                          PRINTF  @CNTR15,@DECSTR
                113512  012746  100526                                                  MOV     @DECSTR,-(SP)
                113516  012746  063135                                                  MOV     @CNTR15,-(SP)
                113522  012746  000002                                                  MOV     @2,-(SP)
                113526  010600                                                          MOV     SP,R0
                113530  104417                                                          TRAP    C$PNTF
                113532  062706  000006                                                  ADD     @6,SP
     6947       113536                          CALL    BINDEC  @UCB12+64       MOV     @UCB12+64,(R5)+
                113536  012725  050130                                          JSR     R4,PREG14
                113542  004437  070656                                          .WORD   BINDEC-ANCHOR
                113546  007446
     6948       113550                          PRINTF  @CNTR16,@DECSTR
                113550  012746  100526                                                  MOV     @DECSTR,-(SP)
                113554  012746  063177                                                  MOV     @CNTR16,-(SP)
                113560  012746  000002                                                  MOV     @2,-(SP)
                113564  010600                                                          MOV     SP,R0
                113566  104417                                                          TRAP    C$PNTF
                113570  062706  000006                                                  ADD     @6,SP
     6949       113574                          PRINTF  @CNTR17,UCB12+70
                113574  013746  050134                                                  MOV     UCB12+70,-(SP)
                113600  012746  063245                                                  MOV     @CNTR17,-(SP)
                113604  012746  000002                                                  MOV     @2,-(SP)
                113610  010600                                                          MOV     SP,R0
```

```
                113612  104417                                                          TRAP    C$PNTF
                113614  062706  000006                                                  ADD     #6,SP
6950            113620                          PRINTF  #CNTR18,UCB12+72
                113620  013746  050136                                                  MOV     UCB12+72,-(SP)
                113624  012746  063320                                                  MOV     #CNTR18,-(SP)
                113630  012746  000002                                                  MOV     #2,-(SP)
                113634  010600                                                          MOV     SP,R0
                113636  104417                                                          TRAP    C$PNTF
                113640  062706  000006                                                  ADD     #6,SP
6951            113644                          PRINTF  #CNTR19,UCB12+74
                113644  013746  050140                                                  MOV     UCB12+74,-(SP)
                113650  012746  063365                                                  MOV     #CNTR19,-(SP)
                113654  012746  000002                                                  MOV     #2,-(SP)
                113660  010600                                                          MOV     SP,R0
                113662  104417                                                          TRAP    C$PNTF
                113664  062706  000006                                                  ADD     #6,SP
6952            113670  105037  003220                  CLRB    P$NNUF
6953            113674  000207                          RTS     PC
6954
6955
6956                                    ;
6957                                    ;ACTION ROUTINE TO PRINT OUT THE NODE TABLE
6958                                    ;
6959
6960            113676  105037  003220          ACTSND: CLRB    P$NNUF
6961            113702  012737  002412  002410          MOV     #NODTBL,SLOT            ;MOVE NODE TABLE ADDRESS INTO SLOT
6962            113710                          CALL    FULSLT                          ;SEE IF TABLE EMPTY
                113710  004437  070656                                                          JSR     R4,PREG14
                113714  025556                                                                  .WORD   FULSLT-ANCHOR
6963            113716  022737  177777  002410          CMP     #-1,SLOT                ;IF YES, DON'T PRINT HEADER
6964            113724  001437                          BEQ     20$
6965            113726                          PRINTF  #NTBHDR                         ;PRINT NODE TABLE HEADER
                113726  012746  053176                                                          MOV     #NTBHDR,-(SP)
                113732  012746  000001                                                          MOV     #1,-(SP)
                113736  010600                                                                  MOV     SP,R0
                113740  104417                                                                  TRAP    C$PNTF
                113742  062706  000004                                                          ADD     #4,SP
6966            113746                  10$:    CALL    FULSLT                          ;FIND LOCATION IN TABLE WITH AN ADDRESS
                113746  004437  070656                                                  JSR     R4,PREG14
                113752  025556                                                          .WORD   FULSLT ANCHOR
6967            113754  022737  177777  002410          CMP     #-1,SLOT                ;CHECK IF AT END OF TABLE
6968            113762  001432                          BEQ     30$                     ;IF YES, RETURN
6969            113764                          CALL    BINHEX SLOT,#6,#STRBUF          ;ELSE, PUT ASCII ADDRESS INTO BUFFER
                113764  012725  002324                                                  MOV     #STRBUF,(R5)+
                113770  012725  000006                                                  MOV     #6,(R5)+
                113774  013725  002410                                                  MOV     SLOT,(R5)+
                114000  004437  070656                                                  JSR     R4,PREG14
                114004  005150                                                          .WORD   BINHEX-ANCHOR
6970            114006                          CALL    PRTNOD                          ;PRINT NODE TABLE ENTRY
                114006  004437  070656                                                  JSR     R4,PREG14
                114012  026040                                                          .WORD   PRTNOD-ANCHOR
6971            114014  062737  000010  002410          ADD     #8.,SLOT                ;INCR. SLOT TO POINT TO NEXT TABLE ENTRY
6972            114022  000751                          BR      10$                     ;CONTINUE UNTIL ALL ENTRIES PRINTED
6973            114024                  20$:    PRINTF  #TABEMT,#NOD
                114024  012746  053415                                                  MOV     #NOD,-(SP)
                114030  012746  053346                                                  MOV     #TABEMT,-(SP)
                114034  012746  000002                                                  MOV     #2,-(SP)
```

```
                                                                                        MOV     SP,R0
                                                                                        TRAP    C$PNTF
                                                                                        ADD     #6,SP
6974 114050 000207          30$:    RTS     PC                      ;RETURN
6975
6976                                ;
6977                                ;ACTION ROUTINE TO CLEAR A NODE SPECIFIED BY NODE LOGICAL NAME
6978                                ;FROM THE NODE TABLE
6979                                ;
6980
6981 114052                 ACTCNL: P$PUSH  R2                      ;SAVE R2
     114052 010225                                                              MOV     R2,(R5)+
6982 114054 013702 003214          MOV     P$NUM,R2                ;PUT NODE LOGICAL NUMBER INTO R2
6983 114060 006302                 ASL     R2                      ;MULTIPLY BY 3
6984 114062 006302                 ASL     R2                      ;NODE TABLE ADDRESS =
6985 114064 006302                 ASL     R2                      ;  (LOG. NO. X 8) + #NODTBL
6986 114066 062702 002412          ADD     #NODTBL,R2              ;ADD OFFSET
6987 114072 005022                 CLR     (R2)+                   ;CLEAR ENTRY (8 BYTES)
6988 114074 005022                 CLR     (R2)+
6989 114076 005022                 CLR     (R2)+
6990 114100 005012                 CLR     (R2)
6991 114102                        P$POP   R2                      ;RESTORE R2
     114102 014502                                                              MOV     -(R5),R2
6992 114104 105037 003220          CLRB    P$NNUF                  ;CLEAR NOTNUF FLAG
6993 114110                        PRINTF  #LOGDEL,P$NUM           ;PRINT MESSAGE INDICATING DELETION
     114110 013746 003214                                                       MOV     P$NUM,(SP)
     114114 012746 054132                                                       MOV     #LOGDEL,(SP)
     114120 012746 000002                                                       MOV     #2,(SP)
     114124 010600                                                              MOV     SP,R0
     114126 104417                                                              TRAP    C$PNTF
     114130 062706 000006                                                       ADD     #6,SP
6994 114134 000207                 RTS     PC                      ;RETURN
6995                                ;
6996                                ;ACTION ROUTINE TO SAVE NODE TABLE
6997                                ;
6998
6999 114136                 ACTSAV: P$PUSH  R2,R3                   ;SAVE R2 AND R3
     114136 010325                                                              MOV     R3,(R5)+
     114140 010225                                                              MOV     R2,(R5)+
7000 114142 012702 002412          MOV     #NODTBL,R2              ;SET REGISTERS FOR COPYING
7001 114146 012703 002536          MOV     #SAVTBL,R3              ;R2 = FROM, R3 = TO
7002 114152                        PRINTF  #UNSMSG,#SAVED          ;PRINT 'TABLE SAVED' MESSAGE
     114152 012746 054324                                                       MOV     #SAVED,(SP)
     114156 012746 054265                                                       MOV     #UNSMSG,(SP)
     114162 012746 000002                                                       MOV     #2,(SP)
     114166 010600                                                              MOV     SP,R0
     114170 104417                                                              TRAP    C$PNTF
     114172 062706 000006                                                       ADD     #6,SP
7003 114176 000137 114260          JMP     SAVCOM
7004
7005                                ;
7006                                ;ACTION ROUTINE TO UNSAVE NODE TABLE
7007                                ;
7008
7009 114202 105037 003220   ACTUNS: CLRB    P$NNUF                  ;CLEAR 'NOT ENOUGH' FLAG
7010 114206                        P$PUSH  R2,R3                   ;SAVE R2 AND R3
     114206 010325                                                              MOV     R3,(R5)+
```

CLI ACTION TABLE AND ROUTINES

```
                                                                                        MOV     R2,(R5)+
     114210  010225
7011 114212  121427  000057                 CMPB    (R4),#57
7012 114216  001002                         BNE     10$
7013 114220  000137  114272                 JMP     QUIT
7014 114224  012703  002412         10$:    MOV     #NODTBL,R3          ;SET REGISTERS FOR COPYING
7015 114230  012702  002536                 MOV     #SAVTBL,R2          ;R2 = FROM, R3 = TO
7016 114234                                 PRINTF  #UNSMSG,#RESTOR     ;PRINT 'TABLE RESTORED' MESSAGE
     114234  012746  054333                                                    MOV     #RESTOR,-(SP)
     114240  012746  054265                                                    MOV     #UNSMSG,-(SP)
     114244  012746  000002                                                    MOV     #2,-(SP)
     114250  010600                                                            MOV     SP,R0
     114252  104417                                                            TRAP    C$PNTF
     114254  062706  000006                                                    ADD     #6,SP
7017 114260  012701  000050         SAVCOM: MOV     #TBLLEN,R1          ;MOVE TABLE LENGTH TO R1
7018 114264  012223                 10$:    MOV     (R2)+,(R3)+         ;MOVE WORD
7019 114266  005301                         DEC     R1                  ;DECREMENT COUNTER
7020 114270  001375                         BNE     10$                 ;IF MORE, LOOP
7021 114272                         QUIT:   P$POP   R2,R3               ; ELSE, RESTORE COUNTERS
     114272  014502                                                            MOV     -(R5),R2
     114274  014503                                                            MOV     -(R5),R3
7022 114276  105037  003220                 CLRB    P$NNUF              ; CLEAR 'NOT ENOUGH' FLAG
7023 114302  000207                         RTS     PC
7024
7025                                 ;
7026                                 ;ACTION ROUTINE TO CLEAR SUMMARY TABLE
7027                                 ;
7028
7029 114304  105037  003220         ACTCSU: CLRB    P$NNUF              ;CLEAR 'NOT ENOUGH' COUNTER
7030 114310                                 P$PUSH  R2                  ;SAVE R2
     114310  010225                                                            MOV     R2,(R5)+
7031 114312  012701  000132                 MOV     #STBLEN,R1          ;MOVE TABLE LENGTH TO R1
7032 114316  012702  002714                 MOV     #STATBL,R2          ;MOVE SUMMARY TABLE ADDRESS TO R2
7033 114322  005022                 10$:    CLR     (R2)+               ;CLEAR FIRST WORD
7034 114324  005301                         DEC     R1                  ;SEE IF FINISHED
7035 114326  001375                         BNE     10$                 ; IF NO, DO MORE
7036 114330                                 PRINTF  #TABCLR,#SUMM       ; ELSE, PRINT 'TABLE CLEARED' MESSAGE
     114330  012746  053422                                                    MOV     #SUMM,-(SP)
     114334  012746  054220                                                    MOV     #TABCLR,-(SP)
     114340  012746  000002                                                    MOV     #2,(SP)
     114344  010600                                                            MOV     SP,R0
     114346  104417                                                            TRAP    C$PNTF
     114350  062706  000006                                                    ADD     #6,SP
7037 114354                                 P$POP   R2                  ; AND RESTORE R2
     114354  014502                                                            MOV     -(R5),R2
7038 114356  000207                         RTS     PC
7039
7040                                 ;
7041                                 ;ACTION ROUTINE TO CHECK FOR PA_S DEFAULT VALUE
7042                                 ;
7043
7044 114360                         ACTDFT:
7045 114360  121427  000040         10$:    CMPB    (R4),#40            ;SEE IF SPACES
7046 114364  001002                         BNE     20$                 ; IF NO, CONT.
7047 114366  005204                         INC     R4                  ; ELSE, POINT TO NEXT CHAR
7048 114370  000773                         BR      10$                 ; AND CHECK AGAIN
7049 114372  121427  000000         20$:    CMPB    (R4),#0             ;SEE IF DEFAULT VALUE
7050 114376  001007                         BNE     30$                 ; IF NO, BR
```

```
114400  012763  000030  000002          MOV     #30,2(R3)               ; IF YES, POINT R3 TO SKIP CHECK PASS COUNT
114406  012737  000001  003214          MOV     #1,P$NUM                ;SET DEFAULT TO 1
114414  000403                          BR      40$                     ;RETURN
114416  012763  000004  000002  30$:    MOV     #4,2(R3)                ;POINT R3 TO CHECK FOR PASS COUNT
114424  000207                  40$:    RTS     PC


                                ;
                                ;ACTION ROUTINE TO READ A FILE FROM EXTERNAL MEDIA ONTO THE NODE TABLE
                                ;

114426                          ACTUSF:
114426                                  P$PUSH  R2                      ;SAVE R2
114426  010225                                                                  MOV     R2,(R5)+
114430  005002                          CLR     R2                      ;INITIALIZE R2 TO NODE TYPE 'TARGET'
114432                                  OPEN    CBOADR                  ;OPEN FILE, NAME=ASCIZ STRING
114432  013700  002370                                                                  MOV     CBOADR,R0
114436  104434                                                                          TRAP    C$OPEN
114440                                  BCOMPLETE       10$             ;RETURN IF SUCCESSFUL
114440  103413                                                                          BCS     10$
114442                                  PRINTF  #OPNERR,CBOADR          ; ELSE PRINT "OPEN ERROR"
114442  013746  002370                                                                  MOV     CBOADR,-(SP)
114446  012746  116072                                                                  MOV     #OPNERR,-(SP)
114452  012746  000002                                                                  MOV     #2,-(SP)
114456  010600                                                                          MOV     SP,R0
114460  104417                                                                          TRAP    C$PNTF
114462  062706  000006                                                                  ADD     #6,SP
114466                                  CLOSE                           ; CLOSE FILE
114466  104435                                                                          TRAP    C$CLOS
114470                          10$:    CALL    RDLIN                   ;READ A LINE AT A TIME
114470  004437  070656                                                          JSR     R4,PREG14
114474  024520                                                                  .WORD   RDLIN ANCHOR
114476  005737  116066                  TST     BAD                     ;SEE IF AN ERROR DURING READ
114502  001064                          BNE     40$                     ; BR ON ERROR TO LEAVE
114504  005737  116070                  TST     EOFF                    ;SEE IF EOF BEFORE PROCESS
114510  001411                          BEQ     20$                     ;IF VALID, PROCESS
114512                                  PRINTF  #EOFFND                 ; ELSE SAY 'END OF FILE' AND LEAVE
114512  012746  116205                                                                  MOV     #EOFFND,-(SP)
114516  012746  000001                                                                  MOV     #1,-(SP)
114522  010600                                                                          MOV     SP,R0
114524  104417                                                                          TRAP    C$PNTF
114526  062706  000004                                                                  ADD     #4,SP
114532  000450                          BR      40$
114534                          20$:    PRINTF  #PLINE,#FILLIN          ;PRINT LINE READ FROM FILE
114534  012746  115542                                                                  MOV     #FILLIN,-(SP)
114540  012746  116154                                                                  MOV     #PLINE,-(SP)
114544  012746  000002                                                                  MOV     #2,-(SP)
114550  010600                                                                          MOV     SP,R0
114552  104417                                                                          TRAP    C$PNTF
114554  062706  000006                                                                  ADD     #6,SP
114560                                  CALL    EDPACK #FILLIN,#ADRBUF,#6       ;PUT ADDRESS INTO BINARY
114560  012725  000006                                                          MOV     #6,(R5)+
114564  012725  002316                                                          MOV     #ADRBUF,(R5)+
114570  012725  115542                                                          MOV     #FILLIN,(R5)+
114574  004437  070656                                                          JSR     R4,PREG14
114600  004600                                                                  .WORD   EDPACK ANCHOR
114602                                  P$POP   R1                      ;CHECK RESULTS
114602  014501                                                                  MOV     -(R5),R1
```

```
7078 114604  001411                         BEQ     30$                     ; IF OK, BR
7079 114606                                 PRINTF  #CADERR                 ; ELSE PRINT ERROR MESSAGE
     114606  012746  053022                                                                     MOV     #CADERR, (SP)
     114612  012746  000001                                                                     MOV     #1, (SP)
     114616  010600                                                                             MOV     SP,R0
     114620  104417                                                                             TRAP    C$PNTF
     114622  062706  000004                                                                     ADD     #4,SP
7080 114626  000412                         BR      40$                     ; AND EXIT
7081 114630  110237  002406         30$:    MOVB    R2,NODTY                ;SET UP NODE TYPE
7082 114634  105102                         COMB    R2                      ;SWITCH TYPE FOR NEXT TIME
7083 114636  142702  000376                 BICB    #376,R2
7084 114642                                 CALL    ENTRND                  ;ENTER IN NODE TABLE
     114642  004437  070656                                                         JSR     R4,PREG14
     114646  025410                                                                 .WORD   ENTRND-ANCHOR
7085 114650                                 P$POP   R1                      ;GET RESULTS
     114650  014501                                                                 MOV     (R5),R1
7086 114652  000706                         BR      10$                     ;READ MORE ADDRESS
7087 114654                         40$:    P$POP   R2                      ;RESTORE R2
     114654  014502                                                                 MOV     -(R5),R2
7088 114656  105037  003220                 CLRB    P$NNUF                  ;CLEAR 'NOT ENOUGH' FLAG
7089 114662                                 PRINTF  #UNSMSG,#RESTOR         ;PRINT 'TALBE RESTORED' MESSAGE
     114662  012746  054333                                                                     MOV     #RESTOR,-(SP)
     114666  012746  054265                                                                     MOV     #UNSMSG, (SP)
     114672  012746  000002                                                                     MOV     #2, (SP)
     114676  010600                                                                             MOV     SP,R0
     114700  104417                                                                             TRAP    C$PNTF
     114702  062706  000006                                                                     ADD     #6,SP
7090 114706                                 RETURN
     114706  000207                                                                 RTS     PC
```

```
                                .SBTTL  RESPONDER TEST
                        ;**
                        ; FUNCTIONAL DESCRIPTION:
                        ;
                        ;       THIS TEST FORWARDS LOOP-SERVER FORWARD REQUEST PACKETS,
                        ;       AND TRANSMITS A CONSOLE ID EVERY 8 TO 9 MINUTES.
                        ;
                        ; CALLING SEQUENCE
                        ;
                        ;       NONE
                        ;
                        ; INPUT
                        ;
                        ;       NONE
                        ;
                        ; IMPLICIT INPUT
                        ;
                        ;       NONE
                        ;
                        ; OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; IMPLICIT OUTPUT
                        ;
                        ;       NONE
                        ;
                        ; COMPLETION CODES
                        ;
                        ;       NONE
                        ;
                        ; SIDE EFFECTS
                        ;
                        ;       NONE
                        ;
                        ; REGISTERS USED
                        ;
                        ;       NONE
                        ;
                        ; DEBUG
                        ;
                        ;       NONE
                        ;--
114710                  ACTRSP:
114710  005237  003202          INC     RSPFLG                  ; INDICATE WE ARE IN LISTEN MODE
114714                  10$:
114714  013701  003750          MOV     TIMMIN,R1               ; GET CURRENT NUMBER OF MINUTES
114720  062701  000010          ADD     #8.,R1                  ; AND ADD 9 MINUTES TO IT
114724                  20$:
114724                          BREAK                           ; ALLOW OPERATOR TO RETAKE CONTROL
114724  104422                                                                  TRAP    C$BRK
114726  020137  003750          CMP     R1,TIMMIN               ; HAS NINE MINUTES GONE BY?
114732  002016                  BGE     30$                     ; BRANCH IF NOT
114734  005001                  CLR     R1                      ; INDICATE THAT TIME HAS GONE BY
114736  012703  004032          MOV     #MCSTAD,R3              ; CONSOLE ID MULTICAST ADDRESS
114742  012704  047674          MOV     #IDTDAT,R4              ; ADDRESS OF CONSOLE ID MESSAGE
114746  012314                  MOV     (R3)+,(R4)              ; SET MULTICAST ADDRESS
```

```
7148 114750  012364  000002                MOV     (R3)+,2(R4)             ; AGAIN
7149 114754  012364  000004                MOV     (R3)+,4(R4)             ; AND LAST TWO BYTES
7150 114760  013702  024220                MOV     XRGCUR,R2               ; GET CURRENT XMIT DESCRIPTOR
7151 114764  000137  115260                JMP     75$                     ; GO XMIT THE CONSOLE ID
7152 114770                          30$:
7153 114770                                CALL    RECEVE                  ; SEE IF THERE IS A BUFFER
     114770  004437  070656                                                        JSR     R4,PREG14
     114774  003044                                                        .WORD   RECEVE-ANCHOR
7154 114776                                P$POP   R2                      ; GET COUNT OF VALID RECEIVES
     114776  014502                                                        MOV     -(R5),R2
7155 115000  001751                        BEQ     20$                     ; BRANCH IF NONE
7156 115002  013702  024226                MOV     RRGNXT,R2               ; ELSE, GET ADDRESS OF DESCRIPTOR
7157 115006  032762  040000  000010        BIT     @MRERRS,STAT1(R2)       ; ARE ANY ERRORS INDICATED
7158 115014  001411                        BEQ     40$                     ; BRANCH IF NOT
7159 115016  032762  010000  000010        BIT     @MDISC,STAT1(R2)        ; ELSE, ARE THEY VALID?
7160 115024  001405                        BEQ     40$                     ; BRANCH IF NOT
7161 115026                                ERRHRD  58,EMSG58               ; ELSE, REPORT ERROR
     115026  104456                                                        TRAP    C$ERHRD
     115030  000072                                                        .WORD   58
     115032  066211                                                        .WORD   EMSG58
     115034  000000                                                        .WORD   0
7162 115036  000555                        BR      110$                    ; DO IT ALL AGAIN
7163 115040                          40$:
7164 115040  016203  000004                MOV     LOADD(R2),R3            ; GET THE DATA BUFFER ADDRESS
7165 115044  016304  000014                MOV     PROTOT(R3),R4           ; GET PROTOCOL TYPE CODE
7166 115050  023704  050306                CMP     PROTOO,R4               ; LOOP SERVER MESSAGE?
7167 115054  001060                        BNE     70$                     ; BRANCH IF NOT
7168 115056  010304                        MOV     R3,R4                   ; POINT TO BEGINNING OF BUFFER
7169 115060  062704  000016                ADD     @LDSKIP,R4              ; BUMP THE POINTER
7170 115064  062404                        ADD     (R4)+,R4                ; POINT TO FUNCTION CODE
7171 115066  022724  000002                CMP     @FORWRD,(R4)+           ; IS THIS A FORWARD MESSAGE?
7172 115072  001405                        BEQ     50$                     ; BRANCH IF YES
7173 115074                                ERRHRD  59,EMSG59               ; ELSE, REPORT UNINVITED REPLY RCVD
     115074  104456                                                        TRAP    C$ERHRD
     115076  000073                                                        .WORD   59
     115100  066267                                                        .WORD   EMSG59
     115102  000000                                                        .WORD   0
7174 115104  000532                        BR      110$                    ; DO IT ALL AGAIN
7175 115106                          50$:
7176 115106  012423                        MOV     (R4)+,(R3)+             ; FILL IN NEW DESTINATION ADDRESS
7177 115110  012423                        MOV     (R4)+,(R3)+             ; AND AGAIN
7178 115112  012423                        MOV     (R4)+,(R3)+             ; AND AGAIN
7179 115114  012704  0040                  MOV     @PHYADR,R4              ; POINT TO OUR PHYSICAL ADDRESS
7180 115120  012423                        MOV     (R4)+,(R3)+             ; FILL IN SOURCE ADDRESS
7181 115122  012423                        MOV     (R4)+,(R3)+             ; AND AGAIN
7182 115124  012423                        MOV     (R4)+,(R3)+             ; AND AGAIN
7183 115126  005723                        TST     (R3)+                   ; POINT PAST TYPE CODE
7184 115130  062713  000010                ADD     @10,(R3)                ; UPDATE THE SKIP COUNT TO NEXT FUNC CODE
7185 115134  016203  000010                MOV     STAT1(R2),R3            ; GET STATUS WORD 2
7186 115140  042703  174377                BIC     @174377,R3              ; CLEAR ALL BUT RCVD BYTE CNT BITS 8 10
7187 115144  116204  000012                MOVB    STAT2(R2),R4            ; GET RCVD BYTE CNT 0 7
7188 115150  042704  177400                BIC     @177400,R4              ; KILL ALL UNWANTED BITS
7189 115154  050403                        BIS     R4,R3                   ; COMBINE THE TWO FOR TOTAL BYTE COUNT
7190 115156  062703  000074                ADD     @60.,R3                 ; ADD MISSING BYTES
7191 115162  006203                        ASR     R3                      ; CONVERT TO A WORD COUNT
7192 115164  005403                        NEG     R3                      ; MAKE IT TWO'S COMPLEMENT
7193 115166  013704  024220                MOV     XRGCUR,R4               ; GET CURRENT XMIT DESCRIPTOR
```

```
 7194 115172  010364  000006                    MOV     R3,WRDCNT(R4)           ; PUT IT INTO THE XMIT DESCRIPTOR
 7195 115176  016404  000004                    MOV     LOADD(R4),R4            ; POINT TO THE XMIT DATA BUFFER
 7196 115202  016202  000004                    MOV     LOADD(R2),R2            ; GET THE RECEIVE DATA BUFFER
 7197 115206                              60$:
 7198 115206  012224                            MOV     (R2)+,(R4)+             ; START FILLING IN THE DATA BUFFER
 7199 115210  005203                            INC     R3                      ; BUMP WORDCOUNT
 7200 115212  001375                            BNE     60$                     ; DO IT UNTIL THE PACKET HAS BEEN .OPIED
 7201 115214  000434                            BR      90$                     ; GO XMIT THE PACKET
 7202 115216                              70$:
 7203 115216  022763  000005  000020            CMP     #IDFUNC,RIFUNC(R3)      ; IS THIS A REQUEST ID CONSOLE T PE?
 7204 115224  001062                            BNE     110$                    ; IF NOT, THROW IT OUT
 7205 115226  012704  047674                    MOV     #IDTDAT,R4              ; POINT TO ID TRANSMIT DATA
 7206 115232  016364  000006  000000            MOV     SOURCC(R3),DESTIN(R4)   ; FILL IN DESTINATION OF ID MESSAGE
 7207 115240  016364  000010  000002            MOV     SOURCC+2(R3),DESTIN+2(R4) ; FILL IN DESTINATION OF ID MESSAGE
 7208 115246  016364  000012  000004            MOV     SOURCC+4(R3),DESTIN+4(R4) ; FILL IN DESTINATION OF ID MESSAGE
 7209 115254  013702  024220                    MOV     XRGCUR,R2               ; GET XMIT DESCRIPTOR
 7210 115260                              75$:
 7211 115260  013703  047770                    MOV     IDTSIZ,R3               ; SET SIZE OF XMISSION
 7212 115264  006203                            ASR     R3                      ; CHANGE TO WORD COUNT
 7213 115266  005403                            NEG     R3                      ; TWOS COMPLEMENT IT
 7214
 7215 115270  010362  000006                    MOV     R3,WRDCNT(R2)           ; SET XFER SIZE
 7216 115274  016202  000004                    MOV     LOADD(R2),R2            ; GET XMIT DATA BUFFER ADDRESS
 7217 115300                              80$:
 7218 115300  012422                            MOV     (R4)+,(R2)+             ; PUT ID DATA INTO XMIT DATA BUFFER
 7219 115302  005203                            INC     R3                      ; BUMP COUNT
 7220 115304  001375                            BNE     80$                     ; DO UNTIL COMPLETE
 7221 115306                              90$:
 7222 115306                                    CALL    XMIT    #XMTDAT         ; TRANSMIT THE PACKET
      115306  012725  000001                                                            MOV     #XMTDAT,(R5)+
      115312  004437  070656                                                            JSR     R4,PREG14
      115316  002326                                                                    .WORD   XMIT-ANCHOR
 7223 115320                                    P$POP   R2                      ; GET STATUS
      115320  014502                                                                    MOV     (R5),R2
 7224 115322  001404                            BEQ     100$                    ; BRANCH IF OKAY
 7225 115324                                    ERRHRD  62,EMSG62
      115324  104456                                                                    TRAP    C$ERHRD
      115326  000076                                                                    .WORD   62
      115330  066451                                                                    .WORD   EMSG62
      115332  000000                                                                    .WORD   0
 7226 115334                              100$:
 7227 115334                                    CALL    GETXNX  #1,#XRGNXT      ; UPDATE XMIT RING POINTERS
      115334  012725  024224                                                            MOV     #XRGNXT,(R5)+
      115340  012725  000001                                                            MOV     #1,(R5)+
      115344  004437  070656                                                            JSR     R4,PREG14
      115350  006534                                                                    .WORD   GETXNX ANCHOR
 7228 115352  005701                            TST     R1                      ; WAS TRANSMISSION A 9 MINUTE SYS ID?
 7229 115354  001006                            BNE     110$                    ; BRANCH IF NOT, RECEIVE POINTER NEEDS UPDATING
 7230 115356  013701  003750                    MOV     TIMMIN,R1               ; ELSE GET MINUTES SINCE START
 7231 115362  062701  000010                    ADD     #8.,R1                  ; AND ADD NINE MINUTES TO IT
 7232 115366  000137  114724                    JMP     20$                     ; NOTHING RECEIVED CAUSED XMIT, DON T UPDATE RCV PN
R
 7233 115372                              110$:
 7234 115372                                    CALL    GETRNX  #1,#RRGNXT      ; UPDATE RECEIVE RING POINTERS
      115372  012725  024226                                                            MOV     #RRGNXT,(R5)+
      115376  012725  000001                                                            MOV     #1,(R5)+
      115402  004437  070656                                                            JSR     R4,PREG14
      115406  006450                                                                    .WORD   GETRNX ANCHOR
```

```
 7235 115410  000137  114724                JMP     20$                ; DO IT ALL AGAIN
 7236
```

READ LINE OF OPENED FILE

```
                                .SBTTL  READ LINE OF OPENED FILE
                                ;
                                ; THIS ROUTINE GETS BYTES FROM AN OPENED FILE UNTIL A CR IS ENCOUNTERED
                                ; "EOF" AND "BAD" FLAGS ARE SET IF END OF-FILE OR ERRORS ARE ENCOUNTERED
                                ;
                                ; NOTE: ASSUMING A ASCII TEXT FILE IS BEING READ, FOR EXAMPLE:
                                ;       AA-00 03-00-01-AB<CR><LF>
                                ;             "
                                ;       AA-00 03-00-01-AB<CR><LF>
                                ;
                                ;       WHAT YOU SEE READ BYTE BY BYTE IS:
                                ;               "A..-AB<CR><LF>A..-AB<CR><LF>..<0><0><0>.....???
                                ;       SO I MADE ASSUMPTION THAT SINCE SEE "0-PADDING" AFTER LAST CHAR TO
                                ;       END-OF-FILEBLOCK, ANY CHARACTER THAT IS NOT "SPACE OR GREATER" OR A
                                ;       <CR> OR <LF> THEN I'LL TAKE THAT AS END-OF-FILE(TEXT), SET EOF FLAG
                                ;       AND LEAVE.
                                ;
                                ;       INPUTS:
                                ;               FILLIN  BUFFER TO HOLD LINE OF BYTES READ FROM OPENED FILE
                                ;                       (CR NOT INCLUDED, 0-BYTE TERMINATED)
                                ;       OUTPUTS:
                                ;               BAD     IF NON-ZERO, ERROR IN READING A BYTE FROM FILE
                                ;               EOFF    IF NON-ZERO, END OF FILE WAS ENCOUNTERED
                                ;               FILLIN  ASCIZ STRING THAT WAS READ AS CHAR-CR-LF STRING
                                ;                        (CR-LF REMOVED)
                                ;
115414  012702  115542        RDLIN:  MOV     #FILLIN,R2              ;POINT R2 TO A LINE BUFFER
115420  005037  116066                CLR     BAD                     ;CLEAR FLAGS
115424  005037  116070                CLR     EOFF
115430                        10$:    GETBYT  (R2)                    ;GO GET A BYTE FROM INPUT FILE
115430  104426                                                                  TRAP    C$GETB
115432  110012                                                                  MOVB    R0,(R2)
115434                                BCOMPLETE       30$             ;BR IF READ-BYTE SUCESSFUL
115434  103414                                                                  BCS     30$
115436                        20$:    PRINTF  #RDERR                  ; ELSE PRINT "READ-ERROR"
115436  012746  116126                                                          MOV     #RDERR, (SP)
115442  012746  000001                                                          MOV     #1,-(SP)
115446  010600                                                                  MOV     SP,R0
115450  104417                                                                  TRAP    C$PNTF
115452  062706  000004                                                          ADD     #4,SP
115456  012737  177777  116066        MOV     #-1,BAD                 ; SET BAD-TRY FLAG AND LEAVE
115464  000416                        BR      50$
                                ;
115466  122712  000015        30$:    CMPB    #15,(R2)                ;IS THE CHARACTER A <CR>
115472  001756                        BEQ     10$                     ; BR IF YES (GO BACK TO GET <LF>)
115474  122712  000012                CMPB    #12,(R2)                ;IS THE CHARACTER A <LF>
115500  001410                        BEQ     50$                     ; BR IF YES (TERMINATE AND LEAVE)
115502  122712  000040                CMPB    #40,(R2)                ;IS IT A "EOF" (END-OF FILE(TEXT))
                                                                        ; (EOF=ANY NON-CHAR>37 EXCEPT CR,LF)
115506  101002                        BHI     40$                     ; BR IF YES
115510  005202                        INC     R2                      ; IF NO, LEAVE CHAR IN BUFFER
115512  000746                        BR      10$                     ; AND GO GET MORE CHARS
                                ;
115514  012737  177777  116070  40$:    MOV     # 1,EOFF               ;IF YES, TERMINATE INPUT BUFF
                                                                        ; AND SET EOF FLAG
115522  105012                50$:    CLRB    (R2)
115524                                RETURN
```

```
READ LINE OF OPENED FILE

     115524  000207                                                          RTS     PC
 7287
 7288 115526                          FILENM: .BLKB   12.             ;BUFFER FOR FILE NAME
 7289 115542                          FILLIN: .BLKB   132.            ;BUFFER FOR SINGLE LINE READ FROM FILE
 7290 115746                          MATCH:  .BLKB   80.             ;BUFFER FOR WORD TO MATCH FROM FILE
 7291 116066  000000                  BAD:    .WORD   0               ;ERROR/NOT-FOUND FLAG WORD
 7292 116070  000000                  EOFF:   .WORD   0               ;END-OF-FILE FLAG (<>0 = EOF)
 7293
 7294 116072     045     116     045  OPNERR: .ASCIZ  /%N%A?UNABLE TO OPEN "%T%A"?/
      116075     101     077     125
      116100     116     101     102
      116103     114     105     040
      116106     124     117     040
      116111     117     120     105
      116114     116     040     042
      116117     045     124     045
      116122     101     042     077
      116125     000
 7295 116126     045     116     045  RDERR:  .ASCIZ  /%N%A?FILE READ ERROR?/
      116131     101     077     106
      116134     111     114     105
      116137     040     122     105
      116142     101     104     040
      116145     105     122     122
      116150     117     122     077
      116153     000
 7296 116154     045     116     045  PLINE:  .ASCIZ  /%N%AFILE LINE WAS:%N%T%N/
      116157     101     106     111
      116162     114     105     040
      116165     114     111     116
      116170     105     040     127
      116173     101     123     072
      116176     045     116     045
      116201     124     045     116
      116204     000
 7297 116205     045     116     045  EOFFND: .ASCIZ  /%N%AEND-OF-FILE FOUND, FILE READ/
      116210     101     105     116
      116213     104     055     117
      116216     106     055     106
      116221     111     114     105
      116224     040     106     117
      116227     125     116     104
      116232     054     040     106
      116235     111     114     105
      116240     040     122     105
      116243     101     104     000
 7298
 7299                                 ; -+
 7300                                 ;       SELMSG                          OPERATOR SELECTED MESSAGE STORAGE
 7301                                 ;
 7302                                 ;       THIS ROUTINE WILL TAKE THE OPERATOR SELECTED MESSAGE FROM THE COMMAND
 7303                                 ;       LINE INPUT STRING BUFFER AND PUT IT INTO A BUFFER AT LOCATION OPSLBF.
 7304                                 ;
 7305                                 ;       INPUTS  -               P1   ADDRESS OF OPERATOR SELECTED MESSAGE IN
 7306                                 ;                                    INPUT STRING
 7307                                 ;       EXPLICIT OUTPUTS -      NONE
 7308                                 ;       IMPLICIT OUTPUTS        THE BUFFER AT OPSLBF WILL CONTAIN THE ASCII
```

```
                                        ;                               OPERATOR SELECTED INPUT STRING FOLLOWED BY
                                        ;                               A NULL CHARACTER
                                        ;       SUBORDINATE ROUTINES    NONE
                                        ;       CALLING PROCEEDURE      CALL SELMSG P1  ;INPUT ADDRESS OF ASCII STRING
                                        ;       REGISTER USAGE -        R1 CONTAINS ADDRESS OF INPUT STRING
                                        ;                               R2 CONTAINS ADDRESS OF OUTPUT STRING
                                        ;
                                        ;  .
                                        SELMSG: P$POP   R1              ;PUT ADDRESS OF OPR. SEL ASCII STRING INTO R1
116246  014501                                                                  MOV     -(R5),R1
                                                MOV     #OPSLBF,R2      ;PUT ADDRESS OF OUTPUT BUFFER INTO R2
                                                CLR     R3              ;CLEAR CHARACTER COUNTER
                                        10$:    TSTB    (R1)            ;CHECK FOR END OF STRING
                                                BEQ     20$             ;GO TO 20$ IF END
                                                MOVB    (R1)+,(R2)+     ;ELSE, MOVE BYTE TO OUTPUT BUFFER
                                                INC     R3              ;COUNT NUMBER OF CHARACTERS IN INPUT BUFFER
                                                JMP     10$             ;GO DO MORE CHARACTERS
                                        20$:    MOVB    #0,(R2)         ;PUT ZERO AT END OF OUTPUT BUFFER
                                                MOV     R3,MSG6C        ;STORE NUMBER OF CHARACTERS FOR USE IN BUF. BUILDING
                                                RETURN
116302  000207                                                                  RTS     PC
                                        ;  .
                                        ;       ENTRND                                  ENTER NODE IN TABLE
                                        ;
                                        ;       THIS ROUTINE ENTERS A NODE INTO THE NODE TABLE
                                        ;
                                        ;       INPUTS                  NONE
                                        ;       EXPLICIT OUTPUTS        P1   ZERO IF SUCCESSFUL,  1 IF TABLE FULL
                                        ;       IMPLICIT OUTPUTS        THE ADDRESS CONTAINED IN ADRBUF WILL BE
                                        ;                               ADDED TO THE NODE TABLE IN THE FIRST
                                        ;                               AVAILABLE SLOT WITH THE NODE TYPE CONTAINED
                                        ;                               IN NODTY (TARGET OR ASSIST)
                                        ;       SUBORDINATE ROUTINES    FINDSL   FIND EMPTY SLOT IN TABLE
                                        ;       CALLING PROCEEDURE      CALL ENTRND
                                        ;                               P$POP P1        ;OUTPUT GOOD/BAD RESULT
                                        ;
                                        ;- .
116304  004437  070656                  ENTRND: CALL FINDSL             ;FIND AVAILABLE SLOT IN TABLE
                                                                                JSR     R4,PREG14
116310  025466                                                                    .WORD   FINDSL ANCHOR
                                                P$POP   R1              ;CHECK IF TABLE FULL
116312  014501                                                                  MOV     (R5),R1
                                                BEQ     10$             ;IF NOT FULL BR TO 10$
                                                P$PUSH  # 1             ;ELSE PUT FULL INDICATION ON STACK
116316  012725  177777                                                          MOV     # 1,(R5)+
                                                BR      30$             ;RETURN
                                        10$:    MOV     #6,R3           ;SET INCR. COUNTER TO 6 (BYTES)
                                                MOV     SLOT,R1         ;MOV ADDRESS OF AVAILABLE SLOT TO R1
                                                MOV     #ADRBUF,R2      ;MOV ADDRESS OF NODE ADDRESS TO R2
                                        20$:    MOVB    (R2)+,(R1)+     ;MOV BYTE OF ADDRESS
                                                DEC     R3              ;DECR. COUNTER
                                                BNE     20$             ;CONTINUE UNTIL 6 BYTES TRANSFERED
                                                INC     R1              ;SET POINTER TO NODE TYPE LOCATION
                                                MOVB    NODTY,(R1)      ;MOVE NODE TYPE INTO TABLE
```

```
READ LINE OF OPENED FILE

7360 116354                             P$PUSH  #0              ;PUT ADDRESS ADDED INDICATION ON STACK
     116354  012725  000000                                                 MOV     #0,(R5)+
7361 116360                     30$:    RETURN                  ;RETURN
     116360  000207                                                         RTS     PC
7362
7363                            ;-*
7364                            ;       FINDSL                          FIND EMPTY SLOT IN NODE TABLE
7365                            ;
7366                            ;       INPUTS                  NONE
7367                            ;       EXPLICIT OUTPUTS        NONE
7368                            ;       IMPLICIT OUTPUTS        THE ADDRESS OF THE FIRST AVAILABLE SLOT IN THE
7369                            ;                               NODE TABLE WILL BE LOCATED IN SLOT.  THE
7370                            ;                               PARAMETER STACK WILL CONTAIN -1 IF THE NODE
7371                            ;                               TABLE IS FULL AND 0 IF AN EMPTY SLOT WAS FOUND
7372                            ;       SUBORDINATE ROUTINES    NONE
7373                            ;       CALLING PROCEDURE       CALL FINDSL
7374                            ;                               P$POP  P1       ; 1 IF FULL/ 0 IF SLOT AVAILABLE
7375                            ;
7376                            ;-*
7377
7378 116362  012702  002412    FINDSL: MOV     #NODTBL,R2              ;MOVE ADDRESS OF NODE TABLE TO R2
7379 116366  022712  000000    10$:    CMP     #0,(R2)                 ;SEE IF SLOT EMPTY
7380 116372  001422                    BEQ     20$                     ;IF YES, BR 20$
7381 116374  062702  000010            ADD     #8.,R2                  ;ELSE MOVE POINTER TO NEXT ENTRY LOC.
7382 116400  022712  177777            CMP     #-1,(R2)                ;SEE IF AT END OF TABLE
7383 116404  001370                    BNE     10$                     ;IF NOT, CONTINUE LOOKING
7384 116406                             PRINTF  #TABFUL,#NOD            ;ELSE, PRINT TABLE FULL MESSAGE
     116406  012746  053415                                                 MOV     #NOD, (SP)
     116412  012746  053274                                                 MOV     #TABFUL, (SP)
     116416  012746  000002                                                 MOV     #2, (SP)
     116422  010600                                                         MOV     SP,R0
     116424  104417                                                         TRAP    C$PNTF
     116426  062706  000006                                                 ADD     #6,SP
7385 116432                             P$PUSH  #-1                     ;PUT TABLE FULL INDICATION ON STACK
     116432  012725  177777                                                 MOV     #-1,(R5)+
7386 116436  000404                    BR      30$                     ;RETURN
7387 116440  010237  002410    20$:    MOV     R2,SLOT                 ;MOVE ADDRESS OF EMPTY LOC. INTO SLOT
7388 116444                             P$PUSH  #0                      ;PUT LOC. FOUND INDICATION ON STACK
     116444  012725  000000                                                 MOV     #0,(R5)+
7389 116450                     30$:    RETURN                          ;RETURN
     116450  000207                                                         RTS     PC
7390
7391                            ;-*
7392                            ;       FULSLT                                  FULL SLOT ROUTINE
7393                            ;
7394                            ;       THIS ROUTINE FINDS A LOCATION IN THE TABLE WHERE A NODE PHYSICAL
7395                            ;       ADDRESS EXISTS.  IT IS USED WHEN PRINTING OUT THE NODE TABLE.
7396                            ;
7397                            ;       INPUTS                  NONE
7398                            ;       EXPLICIT OUTPUTS        NONE
7399                            ;       IMPLICIT OUTPUTS        THE LOCATION SLOT WILL CONTAIN THE PHYSICAL
7400                            ;                               ADDRESS OF A NODE TABLE ENTRY.  SLOT WILL
7401                            ;                               CONTAIN  1 WHEN POINTING TO THE END OF THE
7402                            ;                               NODE TABLE
7403                            ;       SUBORDINATE ROUTINES    NONE
7404                            ;       CALLING PROCEDURE       CALL FULSLT
7405                            ;
```

```
                                                ;   *
                                                
116452  013701  002410                          FULSLT: MOV     SLOT,R1                 ;MOVE SLOT LOCATION TO R1
116456  022711  000000                          10$:    CMP     #0,(R1)                 ;CHECK IF EMPTY
116462  001412                                          BEQ     30$                     ;IF YES, BR 30$
116464  022711  177777                                  CMP     # 1,(R1)                ;SEE IF END OF NODE TABLE
116470  001403                                          BEQ     20$                     ;IF YES, BR 20$
116472  010137  002410                                  MOV     R1,SLOT                 ;ELSE PUT EMPTY LOC. ADDRESS INTO SLOT
116476  000407                                          BR      40$                     ;RETURN
116500  012737  177777  002410                  20$:    MOV     #-1,SLOT                ;PUT  1 INTO SLOT TO SHOW END OF TABLE
116506  000403                                          BR      40$                     ;RETURN
116510  062701  000010                          30$:    ADD     #8.,R1                  ;INCR. POINTER TO NEXT LOCATION
116514  000760                                          BR      10$                     ;CHECK NEXT LOC.
116516                                          40$:    RETURN                          ;RETURN
116516  000207                                                                                  RTS     PC
                                                ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
                                                ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
116520                                          CHECK:
116520  010146                                          MOV     R1,-(SP)
116522  010346                                          MOV     R3,-(SP)
116524  010446                                          MOV     R4,-(SP)
116526  010546                                          MOV     R5,-(SP)

116530  005005                                          CLR     R5
116532  012701  177776                                  MOV     # 2.,R1
116536                                          3$:
116536  062701  000002                                  ADD     #2.,R1
116542  022761  000000  002412                          CMP     #0.,NODTBL(R1)
116550  001772                                          BEQ     3$

116552  062701  000006                                  ADD     #6.,R1
116556  022761  000000  002412                          CMP     #0,NODTBL(R1)
116564  001401                                          BEQ     5$
116566  000413                                          BR      7$
116570                                          5$:
116570  016104  002412                                  MOV     NODTBL(R1),R4
116574  022761  000400  002412                          CMP     #400,NODTBL(R1)
116602  001013                                          BNE     10$

116604  062705  000001                                  ADD     #1,R5
116610  022705  000002                                  CMP     #2,R5
116614  001007                                          BNE     12$
116616                                          7$:
116616  005061  002412                                  CLR     NODTBL(R1)
116622                                                  ERRSOFT 69,EMSG69               ; REPORT IT
116622  104457                                                                                  TRAP    C$ERSOFT
116624  000105                                                                                  .WORD   69
116626  067204                                                                                  .WORD   EMSG69
116630  000000                                                                                  .WORD   0
116632                                          10$:
116632  005005                                          CLR     R5
116634                                          12$:
116634  062701  000002                                  ADD     #2.,R1
116640  022761  177777  002412                          CMP     #-1,NODTBL(R1)
116646  001407                                          BEQ     EXIT1
116650  022761  000000  002412                          CMP     #0.,NODTBL(R1)
116656  001766                                          BEQ     12$
```

```
116660  062701  000006                  ADD     #6.,R1
116664  000741                          BR      5$
116666                          EXIT1:
116666  012605                          MOV     (SP)+,R5
116670  012604                          MOV     (SP)+,R4
116672  012603                          MOV     (SP)+,R3
116674  012601                          MOV     (SP)+,R1
116676                                  RETURN
116676  000207                                                                          RTS     PC
                                ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
                                ;@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
                                ;   +
                                ;       CMPADR                          COMPARE TWO ADDRESSES
                                ;
                                ;       THIS ROUTINE COMPARES TWO SIX BYTE STRINGS
                                ;
                                ;       INPUTS                  P1 - ADDRESS OF FIRST STRING
                                ;                               P2   ADDRESS OF SECOND STRING
                                ;       OUTPUTS                 P3   0 = COMPARISON/ 1 = NO COMPARISON
                                ;
                                ;       CALLING PROCEDURE       CALL    CMPADR P1,P2
                                ;                               P$POP   P3
                                ;
                                ;--+

116700                          CMPADR: P$POP   R2,R3           ;PUT ADDRESS OF STRING TO BE COMPARED IN R2 AND R3
116700  014502                                                                          MOV     (R5),R2
116702  014503                                                                          MOV     (R5),R3
116704  022223                          CMP     (R2)+,(R3)+     ;DO FIRST TWO BYTES COMPARE
116706  001006                          BNE     10$             ; IF NO, EXIT
116710  022223                          CMP     (R2)+,(R3)+     ;DO SECOND TWO BYTES COMPARE
116712  001004                          BNE     10$             ; IF NO, EXIT
116714  021213                          CMP     (R2),(R3)       ;DO LAST TWO BYTES COMPARE
116716  001002                          BNE     10$             ; IF NO, EXIT
116720  005001                          CLR     R1              ;PUT COMPARISON OK INDICATOR IN R1
116722  000402                          BR      20$
116724  012701  177777          10$:    MOV     #-1,R1          ;PUT NO COMPARISON INDICATOR IN R1
116730                          20$:    RETURN  R1
116730  010125                                                                          MOV     R1,(R5)+
116732  000207                                                                          RTS     PC

                                ;-  +
                                ;       PRTNOD                          PRINT NODE TABLE
                                ;
                                ;       INPUTS                  NONE
                                ;       EXPLICIT OUTPUTS        NONE
                                ;       IMPLICIT OUTPUTS        ONE ENTRY IN THE NODE TABLE WILL BE PRINTED
                                ;       SUBORDINATE ROUTINES    NONE
                                ;       CALLING SEQUENCE        CALL PRTNOD
                                ;
116734                          PRTNOD: PRINTF  #NODADR,#STRBUF         ;PRINT NODE ADDRESS
116734  012746  002324                                                                  MOV     #STRBUF,-(SP)
116740  012746  053144                                                                  MOV     #NODADR,-(SP)
116744  012746  000002                                                                  MOV     #2,-(SP)
116750  010600                                                                          MOV     SP,R0
116752  104417                                                                          TRAP    C$PNTF
```

```
                116754  062706  000006                                                          ADD     #6,SP
7505            116760  013702  002410                  MOV     SLOT,R2                 ;MOVE SLOT ADDRESS INTO R2
7506            116764  162702  002412                  SUB     #NODTBL,R2              ;CALCULATE NODE LOGICAL NAME
7507            116770  006202                          ASR     R2                      ;USING:    LOG. NO. *
7508            116772  006202                          ASR     R2                      ;(SLOT #NODTAB)/8
7509            116774  006202                          ASR     R2
7510            116776                                  PRINTF  #LOGNAM,R2              ;PRINT LOGICAL NAME
                116776  010246                                                                  MOV     R2, (SP)
                117000  012746  053154                                                          MOV     #LOGNAM, (SP)
                117004  012746  000002                                                          MOV     #2,-(SP)
                117010  010600                                                                  MOV     SP,R0
                117012  104417                                                                  TRAP    C$PNTF
                117014  062706  000006                                                          ADD     #6,SP
7511            117020  013702  002410                  MOV     SLOT,R2                 ;SEE IF TARGET OR ASSIST NODE
7512            117024  062702  000007                  ADD     #7,R2                   ; INFO CONTAINED IN 7TH BYTE OF ENTRY
7513            117030  111203                          MOVB    (R2),R3                 ;MOVE INTO R3
7514            117032  020327  000001                  CMP     R3,#CASIST              ;SEE IF ASSIST NODE
7515            117036  001404                          BEQ     10$                     ; IF YES, BR 10$
7516            117040  012737  061757  002314          MOV     #ARGTY7,KEYWD2          ; ELSE MOVE 'TARGET' INTO KEYWD2
7517            117046  000403                          BR      20$                     ; CONTINUE
7518            117050  012737  061747  002314  10$:     MOV     #ARGTY6,KEYWD2          ; MOVE 'ASSIST  INTO KEYWD2
7519            117056                          20$:     PRINTF  #NODTYP,KEYWD2          ;PRINT NODE TYPE
                117056  013746  002314                                                          MOV     KEYWD2, (SP)
                117062  012746  053167                                                          MOV     #NODTYP, (SP)
                117066  012746  000002                                                          MOV     #2,-(SP)
                117072  010600                                                                  MOV     SP,R0
                117074  104417                                                                  TRAP    C$PNTF
                117076  062706  000006                                                          ADD     #6,SP
7520            117102                                  RETURN                          ;RETURN
                117102  000207                                                                  RTS     PC
7521            117104                                  EXIT    TST
                117104  104432                                                                  TRAP    C$EXIT
                117106  000002                                                                  .WORD   L10015 .
7522
7524                                    ;*****************************************************************************
7525                                    ;       INSERT LOCAL STORAGE THAT IS USED ONLY
7526                                    ;       DURING THIS TEST.
7527                                    ;*****************************************************************************
7528
7529                                    ;*****************************************************************************
7530                                    ;       INSERT MESSAGES THAT ARE USED ONLY
7531                                    ;       DURING THIS TEST.
7532                                    ;*****************************************************************************
7534
7535                                            .EVEN
7536
7537             117110                         ENDTST
                 117110                                                                 L10015:
                 117110  104401                                                                  TRAP    C$ETST
7538
7540                                    ;*****************************************************************************
7541                                    ;       BEGIN THE REMAINING TESTS ON NEW PAGES.
7542                                    ;*****************************************************************************
```

```
                                .SBTTL  HARDWARE PARAMETER CODING SECTION

                                ;++
                                ; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
                                ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
                                ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
                                ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
                                ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
                                ; WITH THE OPERATOR.
                                ;--

117112                                  BGNHRD
117112  000015                                                          .WORD L10016-L$HARD/2
117114                                                                L$HARD::

                                ;*************************************************************************
                                ;       INSERT HARDWARE PARAMETER INTERPRETIVE CODE HERE.  THIS CODE
                                ;       IS USED BY THE SUPERVISOR TO INTERROGATE THE OPERATOR FOR
                                ;       DEVICE INFORMATION TO PUT IN THE P-TABLE.  THIS CODE IS USED
                                ;       IN CONJUNCTION WITH THE DEFAULT P-TABLE TEMPLATE.  THE MACROS
                                ;       USED IN THIS SECTION ARE "GPRMD", "GPRMA" AND "GPRML".
                                ;*************************************************************************

117114                                  GPRMA   ASKCSR,0,0,160000,177776,YES            ; GET CSR ADDRESS
117114  000031                                                                  .WORD   T$CODE
117116  117146                                                                  .WORD   ASKCSR
117120  160000                                                                  .WORD   T$LOLIM
117122  177776                                                                  .WORD   T$HILIM
117124                                  GPRMA   ASKVEC,2,0,0,776,YES                    ; GET VECTOR ADDRESS
117124  001031                                                                  .WORD   T$CODE
117126  117201                                                                  .WORD   ASKVEC
117130  000000                                                                  .WORD   T$LOLIM
117132  000776                                                                  .WORD   T$HILIM
117134                                  GPRMD   ASKPRI,4,0,340,0,7,YES                  ; GET PRIORITY LEVEL
117134  002032                                                                  .WORD   T$CODE
117136  117235                                                                  .WORD   ASKPRI
117140  000340                                                                  .WORD   340
117142  000000                                                                  .WORD   T$LOLIM
117144  000007                                                                  .WORD   T$HILIM

117146                                  ENDHRD
                                                                                .EVEN
117146                                                                        L10016:

                                ;*************************************************************************
                                ;       INSERT MESSAGES THAT ARE USED ONLY
                                ;       DURING THE HARDWARE PARAMETER CODING SECTION.
                                ;*************************************************************************

117146  127     110     101     ASKCSR: .ASCIZ  /WHAT IS THE PCSRO ADDRESS?/
117151  124     040     111
117154  123     040     124
117157  110     105     040
117162  120     103     123
117165  122     117     040
117170  101     104     104
```

```
                117173      122     105     123
                117176      123     077     000
     7583       117201      127     110     101     ASKVEC: .ASCIZ  /WHAT IS THE VECTOR ADDRESS?/
                117204      124     040     111
                117207      123     040     124
                117212      110     105     040
                117215      126     105     103
                117220      124     117     122
                117223      040     101     104
                117226      104     122     105
                117231      123     123     077
                117234      000
     7584       117235      127     110     101     ASKPRI: .ASCIZ  /WHAT IS THE PRIORITY LEVEL?/
                117240      124     040     111
                117243      123     040     124
                117246      110     105     040
                117251      120     122     111
                117254      117     122     111
                117257      124     131     040
                117262      114     105     126
                117265      105     114     077
                117270      000
     7585                                               .EVEN
     7586
```

```
                        .SBTTL  SOFTWARE PARAMETER CODING SECTION

                        ;++
                        ; THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
                        ; THAT ARE USED BY THE SUPERVISOR TO BUILD P TABLES.  THE
                        ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
                        ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
                        ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
                        ; WITH THE OPERATOR.
                        ;

117272                          BGNSFT
117272  000000                                                          .WORD L10017-L$SOFT/2
117274                                                                L$SOFT::

                        ;*****************************************************************************
                        ;       INSERT SOFTWARE PARAMETER INTERPRETIVE CODING HERE.  THIS CODE
                        ;       IS USED BY THE SUPERVISOR TO INTERROGATE THE OPERATOR FOR
                        ;       SOFTWARE INFORMATION WHICH WILL BE PLACED IN THE SOFTWARE
                        ;       TABLE.  THIS SECTION IS OPTIONAL.
                        ;*****************************************************************************

                                .EVEN

                                ENDSFT
                                                                        .EVEN
117274                                                                L10017:

                        ;*****************************************************************************
                        ;     INSERT MESSAGES THAT ARE USED ONLY
                        ;     DURING THE SOFTWARE PARAMETER CODING SECTION.
                        ;*****************************************************************************

117274                  $PATCH::
117274                          .BLKW   5       ;; N.M.  (CHANGED FROM 100 TO 5)

                        ;*****************************************************************************
                        ;       THIS IS A PATCH AREA THAT SHOULD BE INCLUDED IN ALL DIAGNOSTICS.
                        ;       ADJUST THE SIZE TO FIT YOUR OWN PREFERENCES.
                        ;*****************************************************************************

117306                          LASTAD
                                                                        .EVEN
117306  000000                                                          .WORD   0
117310  000000                                                          .WORD   0
117312                  L$LAST::
117312                          .BLKW   5       ;; N.M. (CHANGED FROM 40 TO 5)
```

```
7635
7636
7638                            ;*****************************************************************************
7639                            ;       HARDCODED P TABLES MAY BE PLACED HERE BY USING THE SETUP MACROS.
7640                            ;       THIS SECTION IS OPTIONAL AND SHOULD BE REMOVED IF IT IS NOT BEING
7641                            ;       USED.  CHANGE THE POINTER MACRO ARGUMENT TO REFLECT THE REMOVAL.
7642                            ;
7643                            ;       THE P-TABLES ARE DELIMITED BY THE "BGNSETUP" AND "ENDSETUP" MACROS.
7644                            ;       THE "BGNSETUP" MACRO HAS ONE ARGUMENT WHICH IS THE NUMBER OF
7645                            ;       P-TABLE ENTRIES.   EACH ENTRY IS DELIMITED BY THE 'BGNPTAB  AND
7646                            ;       "ENDPTAB" MACROS.  NEITHER OF THESE MACROS REQUIRE AN ARGUMENT.
7647                            ;*****************************************************************************
7649
7650                            ;       BGNSETUP        1
7651                            ;       BGNPTAB
7652                            ;       .WORD   0
7653                            ;       ENDPTAB
7654                            ;       ENDSETUP
7655                            ;
7656                            ;
7657                            ; END OF THIS MODULE
7658                            ;-
7659            000001                          .END
```

SYMBOL TABLE

```
ACTALP  106676        B      = 000010        CCLNAD= 000004 G      CMPER2  067677        C$CEFG= 000045
ACTBLD  104440        BAD      116066        CCLNAL= 000010 G      CMPSTR  075702        C$CLCK= 000062
ACTCMS  112506        BCOUNT  050266 G       CCLSUM= 000042        CNDADR= 000030        C$CLEA= 000012
ACTCNL  114052        BD     = 000010 G      CCNTR = 000036        CNDLOG= 000037        C$CLOS= 000035
ACTCNT  112612        BINDEC  100342 G       CCPYS = 000027        CNODAL= 000031        C$CLP1= 000006
ACTCPY  107170        BINHEX  076044          CDEFLT= 000044        CNODE = 000015        C$CVEC= 000036
ACTCSU  114304        BIT0   = 000001 G       CDIR  = 000043        CNTEND  050242        C$DCLN= 000044
ACTCTT  106746        BIT00  = 000001 G       CEXADR= 000013        CNTRS   050144        C$DODU= 000051
ACTDFT  114360        BIT01  = 000002 G       CEXIT = 000020 G      CNTR00  062047        C$DRPT= 000024
ACTDIR  110236        BIT02  = 000004 G       CFLAG   003734        CNTR01  062127        C$DU   = 000053
ACTEXT  106576        BIT03  = 000010 G       CFUNCT= 000040        CNTR02  062176        C$EDIT= 000003
ACTHLP  104072        BIT04  = 000020 G       CHECK   116520        CNTR03  062232        C$ERDF= 000055
ACTIDT  105552        BIT05  = 000040 G       CITSUM  050244        CNTR04  062277        C$ERHR= 000056
ACTMSG  106470        BIT06  = 000100 G       CITTST  050242        CNTR05  062354        C$ERRO= 000060
ACTNAD  107246        BIT07  = 000200 G       CITVEC= 000004 G      CNTR06  062424        C$ERSF= 000054
ACTNAL  107404        BIT08  = 000400 G       CITWEN= 024100 G      CNTR07  062463        C$ERSO= 000057
ACTNOD  104130        BIT09  = 001000 G       CITWOR= 014100 G      CNTR08  062533        C$ESCA= 000010
ACTNUF  104062        BIT1   = 000002 G       CLIACT  103726        CNTR09  062606        C$ESEG= 000005
ACTNUL  104070        BIT10  = 002000 G       CLIALN= 000007        CNTR10  062657        C$ESUB= 000003
ACTONE  106706        BIT11  = 004000 G       CLIALP= 000006        CNTR11  062716        C$ETST= 000001
ACTOPR  106756        BIT12  = 010000 G       CLIBIF= 000003        CNTR12  062766        C$EXIT= 000032
ACTPAT  112332        BIT13  = 020000 G       CLIBR = 000002        CNTR13  063034        C$GETB= 000026
ACTRNA  107612        BIT14  = 040000 G       CLIBRX  052222        CNTR14  063101        C$GETW= 000027
ACTRNL  110670        BIT15  = 100000 G       CLIDEC= 000011        CNTR15  063135        C$GMAN= 000043
ACTRSP  114710        BIT2   = 000004 G       CLIERM  052113        CNTR16  063177        C$GPHR= 000042
ACTRUN  107464        BIT3   = 000010 G       CLIERR= 000000        CNTR17  063245        C$GPLO= 000030
ACTSAV  114136        BIT4   = 000020 G       CLIEXI= 000001        CNTR18  063320        C$GPRI= 000040
ACTSMS  112414        BIT5   = 000040 G       CLINBG  052175        CNTR19  063365        C$INIT= 000011
ACTSND  113676        BIT6   = 000100 G       CLINUF  052144        COMPAR  061515        C$INLP= 000020
ACTSR4  106670        BIT7   = 000200 G       CLINUM= 000005        CONES  = 000017        C$MANI= 000050
ACTSUM  105170        BIT8   = 000400 G       CLIOCT= 000010        COPRSL= 000024        C$MEM  = 000031
ACTSZE  107112        BIT9   = 001000 G       CLISPA= 000004        COUNT   050304 G      C$MSG  = 000023
ACTTYP  107104        BLDAST  076724 G        CLISTR= 000012        CPATRN= 000005        C$OPEN= 000034
ACTUNS  114202        BLDBUF  077652 G        CLITRE  050516        CPYCNT  050326 G      C$PNTB= 000014
ACTUSF  114426        BLDFAS  076376 G        CLI$PM  052104        CPYLMT  053511        C$PNTF= 000017
ACTXAD  106606        BLDFLG  002200          CLKBR   003740        CR     = 000015 G     C$PNTS= 000016
ACTZRO  106716        BLDLD   076132 G        CLKCSR  003736        CRNALL= 000032        C$PNTX= 000015
ACTOAL  106736        BLDMSG  052473          CLKEN   003746        CRUN   = 000004        C$QIO  = 000377
ACT1AL  106726        BLDREQ  077224 G        CLKHZ   003744        CSAVE  = 000006        C$RDBU= 000007
ADDMUL= 000001 G      BMPCNT  074632          CLKINT  070530 G      CSAVR4= 000014        C$REFG= 000047
ADR    = 000020 G     BMPEXI  075356          CLKSET  070504        CSHCTR= 000002 G      C$RESE= 000033
ADRBUF  002316        BMPMUL  075164          CLKVEC  003742        CSHMSG= 000034        C$REVI= 000003
ADRDEL  054044        BMPRCV= 000000 G        CLRMSG  053432        CSIZE  = 000026        C$RFLA= 000021
ALLNOD  061315        BMPREC  075054          CLUPPR= 000033        CSR    = 000016 G     C$RPT  = 000025
ALPHA  = 000000 G     BMPXMT= 000001 G        CMDBUF  002202        CSRBUF  050246        C$SEFG= 000046
ANCHOR  070674        BOE    = 000400 G       CMDTY1  061610        CSR0    050004 G      C$SPRI= 000041
ARGTY1  061702        BRDADR  004016          CMDTY2  061615        CSR1    050020 G      C$SVEC= 000037
ARGTY2  061710        BUFLEN  050332 G        CMDTY3  061625        CTARGT= 000000 G      C$TPRI= 000013
ARGTY3  061721        BUILD  = 000003         CMDTY4  061633        CTYPE  = 000025       C.COLL= 000074 G
ARGTY4  061732        CADERR  053022          CMDTY5  061640        CUNSAV= 000041        C.CRC  = 000001 G
ARGTY5  061743        CADRER  052746          CMDTY6  061644        CUNSVF= 000045        C.FRAM= 000002 G
ARGTY6  061747        CALPHA= 000016          CMDTY7  061654        CZEROS= 000020        C.LCAR= 000002 G
ARGTY7  061757        CASIST= 000001 G        CMDTY8  061661        C$AU   = 000052       C.MLEN= 000004 G
ASKCSR  117146        CBOADR  002370          CMDTY9  061667        C$AUTO= 000061        C.MREC= 000010 G
ASKPRI  117235        CCCITT= 000023          CMPADR  116700        C$BRK  = 000022       C.MXMT= 000040 G
ASKVEC  117201        CCITT  = 000005 G       CMPBUF  050334 G      C$BSEG= 000004        C.OPEN= 000010 G
ASSEMB= 000010        CCLMSG= 000035          CMPER1  067602        C$BSUB= 000002        C.PREC= 000004 G
```

SYMBOL TABLE

```
C.PXMD= 000054 G    EMSG37  064451      FAFCT4= 000050 G    G$RADO= 000020      ILLADR  002656
C.PXMT= 000034 G    EMSG38  064515      FASIST  050454 G    G$XFER= 000004      INIADR= 000000 G
C.PXM2= 000050 G    EMSG4   003532 G    FASKIP= 000016 G    G$YES = 000010      INICLN  103344
C.PXM3= 000044 G    EMSG40  064561      FATFLG  050252 G    HDMSG1  055046      INIEXI  103354
C.RDAT= 000020 G    EMSG41  064662      FILENM  115526      HEADER= 000016 G    INIT    102126
C.RERB= 000014 G    EMSG42  064764      FILLIN  115542      HELP  = 000001      INIT1   102154
C.RERR= 000016 G    EMSG43  065062      FINDSL  116362      HELP1   055120      ISR   = 000100 G
C.RLEX= 000032 G    EMSG44  065141      FLAG  = 000000 G    HELP10  056141      IXE   = 004000 G
C.RLIN= 000030 G    EMSG45  065237      FLAGWO= 000000 G    HELP11  056232      I$AU  = 000041
C.RMDB= 000024 G    EMSG46  065303      FLAG1   050250 G    HELP12  056435      I$AUTO= 000041
C.RTRY= 000001 G    EMSG47  065366      FORWRD= 000002 G    HELP13  056542      I$CLN = 000041
C.RUNT= 000010 G    EMSG48  065446      FRDADR= 000004 G    HELP14  056641      I$DU  = 000041
C.SECS= 000002 G    EMSG5   003632 G    FREMEM  050024 G    HELP15  056740      I$HRD = 000041
C.SHRT= 000004 G    EMSG50  065544      FRESIZ  050022 G    HELP16  057043      I$INIT= 000041
C.SIZ = 000000 G    EMSG51  065600      FULAST  061353      HELP17  057132      I$MOD = 000041
C.XABB= 000070 G    EMSG52  065633      FULSLT  116452      HELP18  057235      I$MSG = 000041
C.XABT= 000072 G    EMSG53  065721      F$AU  = 000015      HELP19  057305      I$PROT= 000040
C.XDAT= 000060 G    EMSG54  065773      F$AUTO= 00002C      HELP2   055221      I$PTAB= 000041
C.XMDB= 000064 G    EMSG55  066063      F$BGN = 00004C      HELP20  057410      I$PWR = 000041
COALT = 000022      EMSG56  066116      F$CLEA= 000007      HELP21  057466      I$RPT = 000041
C1ALT = 000021      EMSG57  066152      F$DU  = 000016      HELP22  057551      I$SEG = 000041
DATCMP  077734 G    EMSG58  066211      F$END = 000041      HELP23  057652      I$SETU= 000041
DECSTR  100526 G    EMSG59  066267      F$HARD= 000004      HELP24  057752      I$SFT = 000041
DEPADR  003774 G    EMSG60  066344      F$HW  = 000013      HELP25  060102      I$SRV = 000041
DESC  = 000002 G    EMSG61  066412      F$INIT= 000006      HELP26  060166      I$SUB = 000041
DESTIN= 000000 G    EMSG62  066451      F$JMP = 000050      HELP27  060272      I$TST = 000041
DFPTBL  002170 G    EMSG63  066526      F$MOD = 000000      HELP28  060374      J$JMP = 000167
DIAGMC= 000000      EMSG64  066615      F$MSG = 000011      HELP29  060513      KEYWD1  002312
DIRCOM  110266      EMSG65  066705      F$PROT= 000021      HELP3   055314      KEYWD2  002314
DIRECT  061337      EMSG66  066777      F$PWR = 000017      HELP30  060563      KILMUL= 000002 G
DNIFLG  050262 G    EMSG67  067071      F$RPT = 000012      HELP4   055365      LAPCNT  050336
EDPACK  075474      EMSG68  067121      F$SEG = 000003      HELP5   055436      LCLKEN= 000100 G
EF.CON= 000036 G    EMSG69  067204      F$SOFT= 000005      HELP6   055536      LDADR1= 000022 G
EF.NEW= 000035 G    ENTRND  116304      F$SRV = 000010      HELP7   055651      LDADR2= 000032 G
EF.PWR= 000034 G    EOFF    116070      F$SUB = 000002      HELP8   055762      LDFCT1= 000020 G
EF.RES= 000037 G    EOFFND  116205      F$SW  = 000014      HELP9    56052      LDFCT2= 000030 G
EF.STA= 000040 G    ERRBLK  052102 G    F$TEST= 000001      HEXBIN  075724      LDRESP  052310
EL    = 001000 G    ERRFLG  050270 G    GETCL   103544      HEXC    076022      LDSKIP= 000016 G
EMSG0   003526 G    ERRMSG  052100 G    GETCOM  077442      HIADD = 000002 G    LENGTH  061506
EMSG01  063436      ERRNBR  052076 G    GETRNX  077344 G    HIRCV = 000006 G    LF    = 000012 G
EMSG02  063467      ERROR   071002 G    GETXNX  077430 G    HIXMT = 000012 G    LGERMS  067745
EMSG03  063527      ERRTYP  052074 G    G$CNTO= 000200      HLPEND  003322      LOADD = 000004 G
EMSG04  063563      ERR1    070306 G    G$DELM= 000372      HLPTAB  003224      LOCDST  075360
EMSG08  063605      ERR2    070344 G    G$DISP= 000003      HLP115  056330      LOE   = 040000 G
EMSG1   003527 G    ERR3    070432 G    G$EXCP= 000400      HN      075700      LOGDEL  054132
EMSG10  063670      EVL   = 000004 G    G$HILI= 000002      HOE   = 100000 G    LOGNAM  053154
EMSG11  063735      EXIT  = 000011      G$LOLI= 000001      HXFORM  075566      LOPDIR  050346 G
EMSG14  063774      EXIT1   116666      G$NO  = 000000      HXN     075564      LORCV = 000004 G
EMSG2   003530 G    E$END = 002100      G$OFFS= 000400      IBE   = 010000 G    LORTGT  004040
EMSG22  064050      E$LOAD= 000035      G$OFSI= 000376      IDENT = 000010      LOT   = 000010 G
EMSG24  064103      FAADR1= 000022 G    G$PRMA= 000001      IDFUNC= 000005 G    LOXMT = 000010 G
EMSG25  064157      FAADR2= 000032 G    G$PRMD= 000002      IDTDAT  047674      LOXTGT  004060
EMSG3   003531 G    FAADR3= 000042 G    G$PRML= 000000      IDTSIZ  047770      LPADR   004024 G
EMSG32  064245      FAADR4= 000052 G    G$RADA= 000140      IDU   = 000040 G    LSRTRY  002402
EMSG34  064322      FAFCT1= 000020 G    G$RADB= 000000      IER   = 020000 G    LST     076042
EMSG35  064367      FAFCT2= 000030 G    G$RADD= 000040      ILADMS  052565      LUPAIR  061326
EMSG36  064424      FAFCT3= 000040 G    G$RADL= 000120      ILADM1  052651      L$ACP   002110 G
```

SYMBOL TABLE

L$APT	002036 G	L10004	070502	MSGTY4	061564	NOD111	051414	NOD22	050700
L$AU	103536 G	L10005	070654	MSGTY5	061571	NOD112	051420	NOD23	050702
L$AUT	002070 G	L10006	102116	MSGTY6	061577	NOD113	051424	NOD24	050704
L$AUTO	103370 G	L10010	103366	MSG0C	003342	NOD114	051440	NOD25	050720
L$CCP	002106 G	L10011	103370	MSG00	003376 G	NOD115	051444	NOD26	050724
L$CLEA	103372 G	L10012	103526	MSG01	003526 G	NOD116	051460	NOD27	050744
L$CO	002032 G	L10013	103534	MSG02	003527 G	NOD117	051464	NOD3	050530
L$DEPO	002011 G	L10014	103542	MSG03	003530 G	NOD12	050616	NOD30	050750
L$DESC	002130 G	L10015	117110	MSG04	003531 G	NOD120	051500	NOD31	050766
L$DESP	002076 G	L10016	117146	MSG05	003532 G	NOD121	051504	NOD32	050772
L$DEVP	002060 G	L10017	117274	MSG1	054415	NOD122	051520	NOD33	051010
L$DISP	002164 G	L5060	052242	MSG1C	003344	NOD123	051524	NOD34	051014
L$DLY	002116 G	MABORT=	004000 G	MSG11	054530	NOD124	051540	NOD35	051030
L$DTP	002040 G	MATCH	115746	MSG12	054643	NOD125	051544	NOD36	051032
L$DTYP	002034 G	MBOOT =	000010 G	MSG2	054703	NOD126	051560	NOD37	051052
L$DU	103530 G	MCARRI=	003000 G	MSG2C	003346	NOD127	051564	NOD4	050544
L$DUT	002072 G	MCHADR=	040000 G	MSG3	054741	NOD13	050622	NOD40	051056
L$DVTY	002122 G	MCRC =	000002 G	MSG3C	003350	NOD130	051600	NOD41	051074
L$EF	002052 G	MCSTAD	004032	MSG4	055002	NOD131	051604	NOD42	051076
L$ENVI	002044 G	MDISC =	010000 G	MSG4C	003352	NOD132	051610	NOD43	051100
L$ERRT	052074 G	MELOOP=	001000 G	MSG5C	003354	NOD133	051614	NOD44	051102
L$ETP	002102 G	MEOM =	020000 G	MSG6C	003356	NOD134	051620	NOD45	051106
L$EXP1	002046 G	MESPAT	061220	MSHORT=	000010 G	NOD135	051634	NOD46	051112
L$EXP4	002064 G	MESPA1	061271	MVALID=	100000 G	NOD136	051640	NOD47	051116
L$EXP5	002066 G	MFAIL =	000400 G	MXERRS=	040000 G	NOD137	051644	NOD5	050546
L$HARD	117114 G	MFERRS=	040000 G	MXLAST=	100000 G	NOD14	050626	NOD50	051122
L$HIME	002120 G	MFLAST=	100000 G	MXMTIN=	000200 G	NOD140	051650	NOD51	051124
L$HPCP	002016 G	MFRAM =	000004 G	MXSETP=	010000 G	NOD141	051666	NOD52	051130
L$HPTP	002022 G	MFUSED=	040000 G	MXUSED=	040000 G	NOD142	051672	NOD53	051144
L$HW	002170 G	MHIBYT=	000100 G	NCRTRY	002404	NOD143	051676	NOD54	051150
L$ICP	002104 G	MILOOP=	000400 G	NETADD=	000000 G	NOD144	051702	NOD55	051166
L$INIT	102126 G	MINTEN=	000100 G	NETAD0	050000 G	NOD145	051706	NOD56	051172
L$LADP	002026 G	MINVRC=	000040 G	NETAD1	050014 G	NOD146	051712	NOD57	051212
L$LAST	117312 G	MINVXM=	000020 G	NEW	103334	NOD147	051732	NOD6	050562
L$LOAD	002100 G	MLOBYT=	000200 G	NIRCNT	050256 G	NOD15	050640	NOD60	051216
L$LUN	002074 G	MLOSS =	010000 G	NOADR	004002 G	NOD150	051736	NOD61	051222
L$MREV	002050 G	MNOCAR=	002000 G	NOCLK	061766	NOD151	051750	NOD62	051224
L$NAME	002000 G	MNXM =	000004 G	NOCMPR	053664	NOD152	051754	NOD63	051230
L$PRIO	002042 G	MOVF =	000001 G	NOD	053415	NOD153	051772	NOD64	051244
L$PROT	102120 G	MRCVEN=	000001 G	NODADR	053144	NOD154	051776	NOD65	051250
L$PRT	002112 G	MRCVIN=	100000 G	NODE =	000002	NOD155	052014	NOD66	051254
L$REPP	002062 G	MRERRS=	040000 G	NODTBL	002412	NOD156	052020	NOD67	051260
L$REV	002010 G	MRESET=	000002 G	NODTY	002406	NOD157	052034	NOD7	050566
L$RPT	102106 G	MRLAST=	100000 G	NODTYP	053167	NOD16	050644	NOD70	051272
L$SOFT	117274 G	MRLONG=	020000 G	NOD0	050516	NOD160	052040	NOD71	051276
L$SPC	002056 G	MRSETP=	020000 G	NOD1	050522	NOD161	052044	NOD72	051302
L$SPCP	002020 G	MRUNT =	004000 G	NOD10	050602	NOD162	052046	NOD73	051306
L$SPTP	002024 G	MRUSED=	040000 G	NOD100	051344	NOD163	052052	NOD74	051312
L$STA	002030 G	MSANIT=	002000 G	NOD101	051362	NOD164	052056	NOD75	051316
L$SW	002200 G	MSGAD	003360 G	NOD102	051366	NOD165	052060	NOD76	051322
L$TEST	002114 G	MSGCNT	003342 G	NOD103	051370	NOD166	052064	NOD77	051340
L$TIML	002014 G	MSGPRM	054345	NOD104	051372	NOD167	052070	NORESP	061446
L$UNIT	002012 G	MSGTAB	003324	NOD105	051376	NOD17	050662	NOTNUF=	000012
L10000	002176	MSGTY0	061536	NOD106	051402	NOD170	052072	NTBHDR	053176
L10001	002200	MSGTY1	061544	NOD107	051406	NOD2	050526	NULL =	000000
L10002	070342	MSGTY2	061551	NOD11	050604	NOD20	050664	NULSTR	053076
L10003	070430	MSGTY3	061557	NOD110	051412	NOD21	050676	N10$	050522

SYMBOL TABLE

```
N100$   051124          N30$    051014          P$EXIT  100670          RRGSRT  024216 G        SLFIST  050312
N101$   051130          N31$    051032          P$GDBD  003221          RRG001  025674 G        SLOT    002410 G
N102$   051150          N32$    051056          P$MERR  003223          RRG002  026774 G        SMLSIZ= 000100 G
N104$   051172          N33$    051076          P$NNUF  003220          RRG003  030074 G        SOURCC= 000006 G
N106$   051216          N80$    051102          P$NUM   003214          RRG004  031174 G        SOURCE  075472
N110$   051222          N81$    051106          P$PASS  002400          RRG005  032274 G        SRBL  = 000010 G
N12$    050530          N82$    051112          P$RADX  003216          RRG006  033374 G        START   102234
N120$   051224          N90$    051116          P$SIZE  002374          RRG007  034474 G        STATBF  002664
N121$   051230          N95$    051122          P$TREE  003206          RRG010  035574 G        STATBL  002714
N122$   051250          OK      061025          P$TRV   100544          RRG011  036574 G        STAT1 = 000010 G
N123$   051276          OKFU    061157          P$TR5   100554          RRG012  037774 G        STAT2 = 000012 G
N124$   051312          OKRE    061046          P$TYPE  002372          RRG013  041074 G        STAT3 = 000010 G
N126$   051316          OKTR    061112          QNAAD0  047772 G        RRG014  042174 G        STAT4 = 000012 G
N130$   051322          ONEALT= 000003 G        QNAAD1  050006 G        RRG015  043274 G        STBLEN= 000132 G
N132$   051344          ONES  = 000001 G        QNAEXI  072122          RRG016  044374 G        STDR  = 000016 G
N134$   051366          OPNERR  116072          QNAINI  071114 G        RRG017  045474 G        STRBUF  002324
N135$   051370          OPRSEL= 000006 G        QNAIN1  071534          RRG020  046574 G        STRBU1  002346
N14$    050546          OPSLBF  003632          QNAISR  073064          RRING   024260 G        SUMM    053422
N140$   051372          O$APTS= 000000          QNAPR0  047776 G        RRINGH  024544 G        SUMMRY= 000007
N141$   051376          O$AU  = 000000          QNAPR1  050012 G        RRNGTB  024172 G        SUMMS1  070043
N142$   051402          O$BGNR= 000001          QNAVC0  047774 G        RSPFLG  003202          SUMMS2  070064
N143$   051412          O$BGNS= 000000          QNAVC1  050010 G        RSPOND= 000046          SUMMS3  070150
N16$    050566          O$DU  = 000000          QUIT    114272          RTRYER  052411          SUMMS4  070177
N160$   051414          O$ERRT= 000000          RASIST  050422 G        RUNALL  107622          SUMMS5  070262
N161$   051420          O$GNSW= 000000          RBFCNT  050264 G        RUNCOM  112000          SUMMS6  070300
N162$   051464          O$POIN= 000001          RBUFB   004100          RUNDIR  110246          SVCGBL= 000000
N163$   051504          O$SETU= 000000          RBUFNR= 000151 G        RUNLUP  110700          SVCINS= 000001
N164$   051524          PART    100542 G        RCVBUF  050300 G        RUNPAT  112342          SVCSUB= 000001
N165$   051544          PASABT  060672          RCVERR  050276 G        SAVCOM  114260          SVCTAG= 000001
N166$   051564          PATTRN  061426          RCVTGT  023772          SAVED   054324          SVCTST= 000001
N167$   051604          PCCALL  050330 G        RDERR   116126          SAVTBL  002536          S$LSYM= 010000
N168$   051614          PCEFLG  050254 G        RDESAL= 002354 G        SCOLL = 000004 G        S.BYTE  050040 G
N170$   051620          PCLKCT= 001600 G        RDESC   021200          SELMSG  116246          S.COMP  050036 G
N175$   051650          PCLKEN= 000111 G        RDESSZ= 000014 G        SETADD  072346          S.LEN   050034 G
N178$   051702          PCMSG   067542 G        RDLIN   115414          SETEXI  072450          S.NREC  050032 G
N18$    050604          PHYADR  004010 G        RECAST  061407          SETFIL  073006          S.REC   050030 G
N180$   051706          PLINE   116154          RECERR  052345          SETINI  072212          S.XFER  050042 G
N181$   051712          PNT   = 001000 G        RECEVE  073740          SETKLD  072410          TABCLR  054220
N182$   051736          PREG14  070656          REPLY = 000001 G        SETUP   072126          TABEMT  053346
N183$   051754          PRI   = 002000 G        REQID   050340 G        SETVAL  072720          TABFUL  053274
N184$   051776          PRI00 = 000000 G        RESTOR  054333          SETWRT  072454          TASIST  050370 G
N185$   052014          PRI01 = 000040 G        RESTRT  103234          SFPTBL  002200 G        TBLLEN= 000050 G
N186$   052020          PRI02 = 000100 G        RETRY   061462          SIADDR= 000042 G        TEMP    050314 G
N190$   052044          PRI03 = 000140 G        RETRYS  050274 G        SIDEV = 000053 G        TEMP1   050316 G
N20$    050622          PRI04 = 000200 G        RIFUNC= 000020 G        SICSR = 000030 G        TEMP2   050320 G
N200$   052046          PRI05 = 000240 G        RLSTHI  003772 G        SIFNCT= 000035 G        TEMP3   050322 G
N201$   052052          PRI06 = 000300 G        RLSTLO  003770 G        SIMSG1  067266          TENPWR  100456
N210$   052060          PRI07 = 000340 G        RNGSIZ  024256          SIMSG2  067325          TGTADR  023570
N212$   052064          PROTOT= 000014 G        RNXPRV  077360 G        SIMSG3  067360          TGTSIZ  023770
N215$   052072          PROT00  050306 G        ROMAD1  003764 G        SIMSG4  067420          TIMERS  003762
N22$    050644          PROT02  050310 G        ROMSIZ  050302 G        SIMSG5  067441          TIMER1  003756
N23$    050664          PRTNOD  116734          RPKLEN= 001100 G        SIMSG6  067467          TIMER2  003760
N24$    050700          P$ACT   003210          RRGCHN  024560 G        SIMSG7  067515          TIMMIN  003750
N25$    050704          P$AERR  003222          RRGCUR  024222 G        SIRCPT= 000022 G        TIMOUT  050272 G
N26$    050724          P$BUFA  003204          RRGLST  024232 G        SIUECO= 000031 G        TIMSEC  003752
N28$    050750          P$CNT   003212          RRGNXT  024226 G        SIVERS= 000027 G        TIMTCK  003754
N29$    050772          P$CPYS  002376          RRGPRV  024234 G        SIZLMT  053575          TRAST   061367
```

SYMBOL TABLE

```
TRVACT  100672          T$LOLI= 000000          T$$SW  = 010001          VINVRC= 000005 G        WRDCNT= 000006 G
TRVADR  101656          T$LSYM= 010000          T$$TES= 010015          VINVXM= 000004 G        WRITES  100072
TRVALN  101472          T$LTNO= 000001          T1      103544 G        VLOBYT= 000007 G        WRTEXI  072736
TRVALP  101426          T$NEST= 177777          UAM    = 000200 G       VLOSS = 000014 G        XC     = 010000 G
TRVBIF  100776          T$NSO = 000005          UCB12   050044          VNOCAR= 000013 G        XFER    050324 G
TRVBR   100766          T$PTNU= 000000          UNBOND  053756          VNXM  = 000002 G        XFLAG   050260 G
TRVBRC  100712          T$SAVL= 177777          UNIT    050026 G        VOVF  = 000000 G        XMIT    073222 G
TRVDEC  101072          T$SEGL= 177777          UNSMSG  054265          VRBL  = 000000 G        XMTDAT= 000001 G
TRVERR  100730          T$SUBN= 000000          V      = 100000 G       VRCVEN= 000000 G        XMTSET= 000000 G
TRVEXI  100750          T$TAGL= 177777          VABORT= 000013 G        VRCVIN= 000017 G        XPKLEN= 001100 G
TRVNMA  101112          T$TAGN= 010020          VBOOT = 000003 G        VRERRS= 000016 G        XRGCUR  024220 G
TRVNOB  100722          T$TEMP= 000005          VCARRI= 000015 G        VRESET= 000001 G        XRGINV  024252
TRVNUM  101104          T$TEST= 000001          VCHADR= 000016 G        VRLAST= 000017 G        XRGLST  024230 G
TRVOCT  101104          T$TSTM= 177777          VCOLLO= 000004 G        VRLONG= 000015 G        XRGNXT  024224 G
TRVSPA  101020          T$TSTS= 000001          VCRC  = 000001 G        VRSETP= 000015 G        XRGSRT  024214 G
TRVSTR  101560          T$$AU = 010014          VDISC = 000014 G        VRUNT = 000013 G        XRG001  024574 G
TSTMS1  060715          T$$AUT= 010011          VECTOR= 000014 G        VRUSED= 000016 G        XRING   024236 G
TSTMS2  060735          T$$CLE= 010012          VELOOP= 000011 G        VSANIT= 000012 G        XRINGH  024252
TSTMS3  060755          T$$DU = 010013          VEOM  = 000015 G        VSHORT= 000003 G        XRNGTB  024212 G
TSTMS4  060770          T$$HAR= 010016          VFAIL = 000010 G        VTDR  = 000000 G        X$     = 000171
T$ARGC= 000002          T$$HW = 010000          VFERRS= 000016 G        VVALID= 000017 G        X$ALWA= 000000
T$CODE= 002032          T$$INI= 010010          VFLAST= 000017 G        VXERRS= 000016 G        X$FALS= 000040
T$ERRN= 000105          T$$MSG= 010004          VFRAM = 000002 G        VXLAST= 000015 G        X$OFFS= 000400
T$EXCP= 000000          T$$PRO= 010007          VFUSED= 000016 G        VXMTIN= 000007 G        X$TRUE= 000020
T$FLAG= 000040          T$$RPT= 010006          VHIBYT= 000006 G        VXSETP= 000014 G        ZEROS  = 000002 G
T$GMAN= 000000          T$$SOF= 010017          VILOOP= 000010 G        VXUSED= 000016 G        ZROALT= 000004 G
T$HILI= 000007          T$$SRV= 010005          VINTEN= 000006 G        WAIT    070720 G        $PATCH  117274 G
T$LAST= 000001

. ABS.  117324      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  32432 WORDS  ( 127 PAGES)
DYNAMIC MEMORY:  20060 WORDS  ( 77 PAGES)
ELAPSED TIME:  00:09:03
CVNIAA.BIN,CVNIAA.LST/-SP=SYC34R.MLB,CVNIAA.P11
.
```