

DMP-11,
DMV-11

DMP/V-11 DCLT
CZCLMBO

AH-F599B-MC
FICHE 1 OF 2

OCT 1981
COPYRIGHT © 1981
MADE IN USA



A microfiche card containing a grid of approximately 20 columns and 20 rows of small, illegible data points or text fragments. The content is too small to be transcribed accurately.

DMP-11,
DMV-11

DMP/V-11 DCLT
CZCLMBO

AH-F599B-MC
FICHE 2 OF 2

OCT 1981
COPYRIGHT © 1981
MADE IN USA



1
2

.TITLE CZCLMBO DMP/V-11 DCLT

.REM 8

IDENTIFICATION

PRODUCT CODE: AC-F597B-MC
PRODUCT NAME: CZCLMBO DMP,DMV-11 DATA COMM. LINK TEST
PRODUCT DATE: 26-OCT-81
MAINTAINER: MERRIMACK DIAGNOSTIC ENGINEERING
AUTHOR: BRUCE LUHRS - BRUCE RIBOLINI

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1981 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL
DEC

PDP
DECUS

UNIBUS
DECTAPE

MASSBUS

REVISION HISTORY:

<u>REV</u>	<u>DATE</u>	<u>AUTHOR</u>	<u>REASON</u>
A	14-JAN-81	BRUCE RIBOLINI	ORIGINAL ISSUE, DCLT FOR THE DMP,DMV-11
B	26-OCT-81	ERNIE COOPER	ADD - "SET E=T COMMAND" ADD - ID OF DEVICE REQUESTING DOWNLINELOAD. ADDED NEEDED PATCHES. GENERAL CLEANUP AND ENHANCEMENT OF DOCUMENT.

TABLE OF CONTENTS

- 1.0 GENERAL INFORMATION
 - 1.1 PROGRAM ABSTRACT
 - 1.2 SYSTEM REQUIREMENTS
 - 1.3 RELATED DOCUMENTS AND STANDARDS
 - 1.4 DIAGNOSTIC HIERARCHY PREREQUISITES
 - 1.5 ASSUMPTIONS - RESTRICTIONS
- 2.0 OPERATING INSTRUCTIONS
 - 2.1 COMMANDS
 - 2.2 SWITCHES
 - 2.3 FLAGS
 - 2.4 HARDWARE QUESTIONS
 - 2.5 DATA COMM. LINK TEST COMMANDS
 - 2.5.1 MESSAGE COMMANDS
 - 2.5.2 TRIB COMMANDS
 - 2.5.3 STATISTICAL COMMANDS
 - 2.5.4 RUN COMMANDS
 - 2.5.5 PRINT COMMANDS
 - 2.5.6 DEFAULTS
 - 2.6 QUICK STARTUP PROCEDURE
- 3.0 ERROR INFORMATION
 - 3.1 TYPES OF ERROR MESSAGES
 - 3.2 SPECIFIC ERROR MESSAGES
 - 3.2.1 COMMAND LINE INTERPRETER ERRORS
 - 3.2.2 DCLT ERROR MESSAGES
 - 3.2.3 DEVICE ERROR MESSAGES
- 4.0 PERFORMANCE AND PROGRESS REPORTS
 - 4.1 PRINTING EVENT LOG
 - 4.2 OPERATOR STATUS MESSAGES
- 5.0 DEVICE INFORMATION TABLES
- 6.0 MODE AND MESSAGE DESCRIPTIONS
 - 6.1 MODE DESCRIPTIONS
 - 6.1.1 TRANSMIT MODE
 - 6.1.2 RECEIVE MODE
 - 6.1.3 PASSIVE MODE
 - 6.1.4 ACTIVE MODE
 - 6.1.5 DOWN LINE LOAD
 - 6.1.6 TALK AND LISTEN
 - 6.1.6.1 TALK MODE
 - 6.1.6.2 LISTEN MODE
 - 6.1.7 MAINTENANCE LOOP SUMMARY
 - 6.1.8 MODE SUMMARY TABLE
 - 6.2 MESSAGE DESCRIPTIONS
- 7.0 OTHER INFORMATION
 - 7.1 INTERFACING TO AN "ITEP" NODE
 - 7.2 TROUBLESHOOTING HINTS
 - 7.3 EXAMPLES OF COMMANDS
 - 7.4 THINGS TO WATCH OUT FOR

1.0 GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

THIS DCLT (DATA COMMUNICATION LINK TEST) PROGRAM IS MEANT TO PROVIDE FIELD SERVICE WITH A TOOL TO MAINTAIN DMP, DMV-11 TO DDCMP MULTIPOINT COMMUNICATION LINKS. THIS DCLT PROGRAM WILL PROVIDE THE COVERAGE NECESSARY TO DETECT FAILURES IN THE COMPUTER EQUIPMENT, THE COMMUNICATION LINK, OR THE MODEM.

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 (CHQUS?.SEQ WHERE ? IS REV. LEVEL OF THE 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 CZCLM DCLT PROGRAM, THE FOLLOWING MINIMUM HARDWARE IS REQUIRED:

- A PDP-11 CPU IF DMP OR A LSI-11 CPU IF DMV
- MINIMUM OF 24K WORDS OF MEMORY
- A WORKING, LINE OR REAL-TIME CLOCK
- A CONSOLE TERMINAL
- ANY XXDP+ SUPPORTED LOAD MEDIA
- ONE OF THESE DMP, DMV-11 CONFIGURATIONS:
 - DMV-11-AA EIA RS232 AND RS423
 - DMV-11-AB CCITT AND V.35
 - DMV-11-AC INTEGRAL MODEM
 - DMP-11-AA EIA RS232 AND RS423 WITH H3251 TURNAROUND
 - DMP-11-AB CCITT AND V.35
 - DMP-11-AC INTEGRAL MODEM
 - DMP-11-AE RS422
 - DMP-11-AD DMP WITH TURNAROUND CONN (H3254, H3255)

NOTE: OPTIONS AE, AC, AB, AND AA ALSO CONTAIN AD.

1.3 RELATED DOCUMENTS AND STANDARDS

- DMP USERS MANUAL EK-DMP11-UG-001
- DMP TECH MANUAL EK-DMP11-TM-001
- DMV USERS MANUAL EK-DMV11-UG-001
- DMV TECH MANUAL EK-DMV11-TM-001
- 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 DATA COMM. LINK TEST PROGRAM IS TO TEST THE COMMUNICATION LINK AND THEREFORE ASSUMES THAT THE CPU'S, CLOCKS, AND DMP,DMV-11'S AT EACH END OF THE LINK HAVE ALREADY BEEN TESTED.

IF NO LINE OR REAL-TIME CLOCK IS FOUND, THE PROGRAM WILL CONTINUE BUT ANY OF THE PROGRAM THAT TIMES THE DEVICE WILL HANG IF THE DEVICE TIMES OUT. ALSO, THE EVENT LOG WILL CONTAIN A ZERO EVENT TIME FOR ALL EVENTS LOGGED.

IT IS NOT THE INTENTION OF A DATA COMM. LINK TEST PROGRAM TO TEST THE DMP,DMV-11, BUT TO TEST THE COMMUNICATION LINK TO WHICH THEY ARE CONNECTED.

SOME OF THE DIAGNOSTICS THAT COULD BE RUN IF THE DMP,DMV-11 LOOKS BAD:

CZDMT - FUNCTIONAL DIAGNOSTIC FOR DMP,DMV-11

FOR DMP:

CZDMP - 8207 STATIC #1 (PROCESSOR)
CZDMQ - 8207 STATIC #2 (PROCESSOR)
CZDMR - 8203 STATIC #1 (LINE UNIT)
CZDMS - 8203 STATIC #2 (LINE UNIT)

FOR DMV:

CVDMA - MICRO PROCESSOR #1
CVDMB - MICRO PROCESSOR #2
CVDMC - LINE UNIT #1
CVDMD - LINE UNIT #2
CVDME - LINE UNIT #3

1.5 ASSUMPTIONS - RESTRICTIONS

IT IS ASSUMED THAT THE COMMUNICATIONS DEVICE (DMP,DMV-11) 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 DCLT.

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 'DDDD'.

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:DDDD	EXECUTE DDDD PASSES (DDDD = 1 TO 64000)
/FLAGS:FLGS	SET SPECIFIED FLAGS. FLAGS ARE DESCRIBED IN SECTION 2.3.
/EOP:DDDD	REPORT END OF PASS MESSAGE AFTER EVERY DDDD PASSES ONLY. (DDDD = 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)
ISF	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).

THE DMP,DMV-11 DATA COMM. LINK TEST PROGRAM WILL NOT USE MORE THAN ONE UNIT. FOR THE DMP,DMV-11 THE HARDWARE INFORMATION REQUESTED WILL BE:

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

UNIT 0
FULL DUPLEX OPERATION : (L) Y ?
DEVICE CSR ADDRESS : (0) 160170 ?
INTERRUPT VECTOR ADDRESS: (0) 300 ?
INTERRUPT PRIORITY: (0) 5 ?
OPTION TYPE
0=DMP
1=DMV: (0) 0 ?
IS THIS A MULTIPOINT NETWORK: (L) N ?
IS THIS A CONTROL STATION: (L) N ?
```

NOTE: THE QUESTION ABOUT CONTROL STATION IS ONLY ASKED IF YOU ANSWER YES TO THE MULTIPOINT QUESTION. WHEN YOU COMPLETE THE ABOVE SEQUENCE YOU WILL BE AT THE DCLT> COMMAND LEVEL

```
THIS IS DCLT. TYPE 'H' OR '?' FOR DETAILS
MODE=ACTIVE/PASS=00001
DCLT> (A) ? /NOSTATUS/CHECK/NOECHO/NOMODEM
```

2.5 DATA COMM. LINK TEST COMMANDS

THE 'DCLT>' COMMAND LEVEL FOLLOWS THE ANSWERING OF THE HARDWARE P-TABLE QUESTIONS. THESE COMMANDS CAN BE TYPED WHEN THE 'DCLT> (A) ?' PROMPT IS PRINTED.

YOU ONLY HAVE TO TYPE ENOUGH CHARACTERS TO UNIQUELY SPECIFY A COMMAND.

THE COMMAND LINE IS INTERPRETED FROM LEFT TO RIGHT. THEREFORE, IF A QUALIFIER ON THE COMMAND LINE IS RELATED OR EFFECTS A QUALIFIER TO THE LEFT ON THE COMMAND LINE, THE QUALIFIER FARTHEREST TO THE RIGHT TAKES PRECEDENCE SINCE IT IS INTERPRETED LAST. (I.E. IF /CHECK..... .../NOCHECK APPEAR ON THE SAME LINE, NOCHECK WILL BE INDICATED IN THE PARAMETERS WORD.)

REFER TO SECTION 6.0 FOR A DESCRIPTION OF THE DIFFERENT MODES OF OPERATION AND THE TYPES OF MESSAGES AVAILABLE.

2.5.1 MESSAGE COMMANDS

COMMAND	DESCRIPTION
CLEAR EXPECTLIST	ZEROES THE EXPECTLIST (000'S) AND THEN INITIALIZES LIST TO ONE DEFAULT ITEP MESSAGE
CLEAR TRANSMITLIST	FILLS TRANSMITLIST (000'S) AND THEN INITIALIZES LIST TO ONE DEFAULT ITEP MESSAGE
SET EXPECTMSG=TYPE/QUAL	DEFINE A MESSAGE TO BE PUT ON THE EXPECTED LIST
WHERE: "TYPE" IS: =ONES =ZEROES =1ALT =OALT =ITEP =CCITT =ALPHA ='A-Z,0-9,SPACES OR TABS IN QUOTES'	
WHERE THE OPTIONAL 'QUAL' IS: /SIZE=NNN	MAKE THE MESSAGE 'NNN' BYTES LONG. (DEFAULT VALUE IS SIZE OF MESSAGE SPEC'D BY

/COPY=NN

OPERATOR OR DEFAULTS.)
COPY THIS MESSAGE INTO THE
BUFFER 'NN' TIMES (DEFAULT
IS 0 = PUT THE MESSAGE IN
ONLY ONCE)

NOTE: SET'S ADD MESSAGES TO THE LIST IN THE ORDER THEY'RE
DEFINED. 'NNN' IS A DECIMAL NUMBER. THE FIRST SET
OVERWRITES THE DEFAULT ITEP MESSAGE PLACED THERE BY
INITIALIZATION OR A 'CLEAR' COMMAND.

SEE SECTION 6.2 FOR A DESCRIPTION OF THE PRE-DEFINED
MESSAGES THAT ARE AVAILABLE. (ZEROS,ONES ...)

SET	EXPECT=TRANSMIT	MAKES A COPY OF THE TRANSMIT LIST IN THE EXPECT LIST.
SET	TRANSMITMSG=TYPE/QUAL	DEFINE A MESSAGE TO BE PUT ON THE TRANSMIT LIST (SEE DESCRIPT FOR SET EXP)
SHOW	EXPECTLIST	LISTS THE MESSAGE SIZE AND TYPE FOR THE MESSAGES IN THE EXPECT LIST
SHOW	TRANSMITLIST	LISTS THE MESSAGE SIZE AND TYPE FOR THE MESSAGES IN THE TRANSMIT LIST

2.5.2 TRIBUTARY COMMANDS

NOTE: THESE COMMANDS ARE VALID ONLY IF IN MULTIPOINT MODE.

TRIB ESTABLISH=N,N,N/W

ADDS THE DECIMAL TRIBUTARY ADDRESSES SPECIFIED IN N TO THE TRIB LIST.

IF /W IS USED THEN PROGRAM WILL ASK USER FOR POLL PARAMS FOR ALL TRIBS THAT HAVE THE /W SWITCH APPENDED. AFTER ALL TRIB PARAM QUESTIONS HAVE BEEN ANSWERED THEN THE PROGRAM ASKS THE USER FOR THE GLOBAL POLL PARAMS

TRIB KILL=N,N,N OR ALL

REMOVE TRIB ADDRESSES FOR THE TRIB LIST IF 'ALL' IS USED ALL TRIBS ARE REMOVED.

TRIB SHOW

LISTS ALL TRIBS IN THE TRIB ADDRESS LIST.

2.5.2 STATISTICAL COMMANDS

HELP

TYPES HELP INFO FOR OPERATOR

?

TYPES HELP INFO FOR OPERATOR

DUMP SSSSSS-EEEEEE/B

PRINTS THE CONTENTS OF THE
MEMORY LOCATIONS BETWEEN
OCTAL ADDRESSES 'SSSSSS' AND
'EEEEEE' WHERE 'SSSSSS' IS

THE START ADDRESS AND
'-EEEEEE' IS THE END ADDRESS.
IF '-EEEEEE' IS NOT SPECIFIED
THEN THE CONTENTS OF 'SSSSSS'
IS PRINTED IN WORD FORMAT.

THE '/B' IS OPTIONAL.
DEFAULT IS PRINT WORDS
'/B' CAUSES PRINT BYTES

NOTE: THE DUMP COMMAND IS USEFUL FOR EXAMINING
MESSAGE DATA. STARTING ADDRESSES CAN
BE FOUND BY LOOKING IN THE EVENT LOG.

2.5.3 RUN COMMANDS

COMMAND	DESCRIPTION
RUN MODE=MTYPE/QUAL	STARTS DCLT EXECUTING IN THE MODE SPECIFIED
NOTE: MODE=ACTIVE IS NOT DEFAULT, A MODE=MTYPE MUST BE TYPED ----- EACH TIME A RUN IS TYPED	
WHERE THE 'MTYPE' IS ANY ONE OF THE FOLLOWING:	
=ACTIVE	(FORCES /NOECHO ,NO LOOPING)
=PASSIVE	(FORCES NO LOOPING)
=RECEIVE	(FORCES /NOECHO ,NO LOOPING)
=LISTEN	(FORCES /NOECHO ,NO LOOPING, /NOCHECK)
=TRANSMIT	(FORCES /NOECHO ,NO LOOPING, /NOCHECK)
=TALK	(FORCES /NOECHO ,NO LOOPING, /NOCHECK)
=DOWNLINELOAD	(FORCES /NOECHO ,NO LOOPING, /NOCHECK)
(FORCING NO LOOPING MEANS IT MUST BE SPECIFIED AS A QUALIFIER ANY TIME ITS DESIRED, THERE IS NO DEFAULT)	
AND OPTIONAL 'QUAL' IS ANY COMBINATION OF THE FOLLOWING:	
/CHECK/NOCHECK	ENABLES/DISABLES CHECKING OF RECEIVED DATA AGAINST THE EXPECTED DATA
NOTE: IF BOTH MODES IN ACTIVE AND "/NOCHECK" IS USED, ----- END-OF-PASS IS DEFINED AS RECEIVING THE SAME # OF MESSAGES THAT IS CONTAINED IN THE TX LIST. WITH NO DATA CHECKING, THERE IS NO WAY FOR DCLT TO KNOW HOW MANY MESSAGES IT SHOULD EXPECT TO RECEIVE.	
/STATUS/NOSTATUS	ENABLES/DISABLES PRINTING OF PROGRAM STATUS MESSAGES TO THE OPERATOR
/ECHO/NOECHO	ENABLES/DISABLES THE RETRANSMISSION OF THE DATA RECEIVED IN PASSIVE MODE. NOTE: THIS IS VALID ONLY FOR PASSIVE MODE. IF THIS SWITCH IS USED THE TRANSMIT LIST WILL HAVE TO BE RE BUILT.
/MODEM/NOMODEM	ENABLES/DISABLES THE REPORTING OF MODEM STATUS INTERRUPT CHANGES. NOTE: THIS SWITCH CAUSES NO ACTION IN THIS DCLT PROGRAM BUT IT IS INCLUDED BECUASE IT IS USED IN OTHER DCLT PROGRAMS.
/LOOP=LTYPE	SPECIFIES WHICH, IF ANY, TYPE OF

MAINTENANCE LOOPBACK IS BEING USED.
(IGNORED IN MODES OTHER THAN ACTIVE)
MUST BE SPECIFIED EACH TIME ELSE NO
LOOP IS USED.

'LTYPE' IS:

- =INTERNALTTL SETS THE LULOOP BIT IN BSEL1 IF DMP
AND IF DMV ENTERS MAINT LOOP AND
SETS THE INTERNAL LOOP BIT.
- =CABLE DOES NOT CAUSE ANY BITS TO BE SET OR
REQUESTS TO BE QUEUED, BUT MAKES FOR
A NICE BOOKKEEPING FEATURE. '/L=CABLE'
WILL THEN BE SHOWN WHEN THE COMMAND
LINE IS TYPED AS A REMINDER OF WHAT
TYPE OF LOOPING IS BEING ATTEMPTED.
REMEMBER TO INSTALL ANY CONNECTORS OR
ENABLE ANY LOOP FEATURES THAT ARE
NECESSARY TO MAKE CABLE LOOPBACK
POSSIBLE.

THE FOLLOWING LOOP TYPES ARE NOT SUPPORTED BY THE DMV.
INCLUDING THESE LOOP TYPES FOR A DMV WILL HAVE NO EFFECT AT ALL.

- =LOCALMODEM
ALSO CALLED ANALOG-LOOPBACK.
SETS MM1 AND DSR IN THE MODEM REG
THIS IS ONLY FOR RS449 MODEMS
- =REMOTEMODEM
ALSO CALLED DIGITAL-LOOPBACK.
SETS MM2 AND DSR IN THE MODEM REG
THIS ONLY FOR RS449 MODEMS

/PASS=NN SPECIFIES NUMBER OF ITERATIONS TO MAKE BEFORE
END-OF-PASS. DEFAULT VALUE OF 1
WILL BE USED ON ANY RUN THAT A /PASS=N
IS NOT ADDED TO THE 'RUN ...' COMMAND.
IF A '-1' IS TYPED, THEN THE PROGRAM
RUN UNTIL A ^C IS TYPED.

NOTE: SEE SECTION 6.1 FOR A DESCRIPTION
----- OF THE 'RUN MODES' AND 'LOOP MODES'

EXIT

THE EXIT COMMAND RETURNS THE USER TO THE SUPERVISOR DR>
PROMPT AFTER PRINTING A SUPERVISOR END OF PASS.

2.5.4 PRINT

THE PRINT COMMAND TAKES YOU A LEVEL BELOW DCLT> CALLED
REPORT THE COMMANDS AVAILABLE IN RPT> ARE...

<u>COMMAND</u>	<u>DESCRIPTION</u>
HELP OR ?	PRINTS HELP INFORMATION FOR RPT>
TSS NNN/SW	SHOWS TRIBUTARY STATUS SLOT INFORMATION WHERE NNN IS THE DECIMAL TRIBUTARY ADDR AND SW IS ONE OF THE FOLLOWING SWITCHES
ERROR	INDICATES ONLY ERROR SLOTS ARE TO BE PRINTED
FULL	INDICATES ALL TRIB STATUS SLOTS ARE TO BE PRINTED
OFFSET=NN	INDICATES THE TRIB STATUS SLOT WHOSE OFFSET IS NN IS TO BE PRINTED.
GSS/SW	PRINT THE GLOBAL STATUS INFORMATION SWITCHES ARE THE SAME AS FOR TSS.
LOG	DUMPS THE EVENT LOG
EXIT	EXITS BACK TO THE COMMAND LEVEL THAT YOU ENTERED FROM. [DCLT> OR DR>]

2.5.6 DEFAULTS

IF NO "SET'S" THEN THE DEFAULT IS SAME AS IF TYPED:

SET TRANSMITMSG=ITEP/SIZE=58/COPY=0
SET EXPECTMSG=ITEP/SIZE=58/COPY=0

THE DEFAULT COPY AND SIZE FOR EACH OF THE MESSAGE TYPES:

ONES - /SIZE=64/COPY=0
ZEROES - /SIZE=64/COPY=0
OALT - /SIZE=64/COPY=0
1ALT - /SIZE=64/COPY=0
CCITT - /SIZE=64/COPY=0
ALPHA - /SIZE=65/COPY=0
ITEP - /SIZE=58/COPY=0
OPER. SPEC'D - /SIZE=LENGTH-OF-TEXT-TYPED-BETWEEN-QUOTES/COPY=0

FOR THE RUN COMMAND THE DEFAULTS ARE:

RUN MODE=ACTIVE/NOSTATUS/CHECK/NOECHO/NOMODEM/PASS=1

NOTE: MODE=ACTIVE IS NOT DEFAULT, A MODE=MTYPE MUST BE TYPED
----- EACH TIME A RUN IS TYPED

IF THE DCLT PROGRAM IS RUN IN UNATTENDED MODE (UAM FLAG=1 OR CHAINED),
THE DEFAULTS ARE AS IF THESE SETUP AND RUN COMMANDS WERE TYPED:

SET TRANS=ITEP
SET EXPECT=ITEP
RUN MODE=ACTIVE/LOOP=INTERNAL/NOSTAT/NOECHO/NOMODEM/CHECK/PASS=1

OTHER NOTES:

^C ALWAYS RETURNS YOU TO 'DR>' (THE SUPERVISOR)
<CR> IS SEEN AS A COMMAND TERMINATOR
'RUBOUT' - DELETE LAST CHAR. TYPED IN COMMAND STRING

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. THE NUMBER OF UNITS THAT CAN DCLT CAN USE IS ALWAYS '1'.

WHEN YOU FOLLOW THIS PROCEDURE YOU WILL BE USING ONLY THE DEFAULTS FOR FLAGS. THESE DEFAULTS ARE DESCRIBED IN SECTION 2.3.

7. AFTER THE 'DCLT> (A) ?' PROMPT, TYPE
'RUN MODE=ACTIVE<CR>'

WHEN YOU FOLLOW THIS PROCEDURE YOU WILL BE USING THE DEFAULT TRANSMIT AND EXPECTED MESSAGES. THE DEFAULT PASS COUNT AND 'RUN' QUALIFIERS ARE ALSO BEING USED. THESE DEFAULTS ARE DESCRIBED IN SECTION 2.5.3.

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 "IBE" 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", "IBE" OR "IXE" 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

3.2.1 COMMAND LINE INTERPRETER ERRORS:

<u>ERROR MESSAGE:</u>	<u>MEANING</u>
?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.
?INCMPLTE CMD?	A REQUIRED PART OF A COMMAND WAS LEFT OUT.
?NUM 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.
?'LOOP' VALID ONLY IN ACTIVE?	THE '/LOOP=..' SWITCH WAS TYPED IN A RUN COMMAND BUT THE MODE WAS NOT SET

TO ACTIVE. MAINTENANCE LOOP IS ONLY POSSIBLE IF THE MODE OF OPERATION IS ACTIVE.

? 'ECHO' VALID ONLY IN PASSIVE? THE '/ECHO' SWITCH WAS TYPED IN A RUN COMMAND BUT THE MODE WAS NOT SET TO PASSIVE. ECHOING OF RECEIVED DATA IS ONLY POSSIBLE IF THE MODE OF OPERATION IS PASSIVE.

? ILL CHR- 'A-Z,0-9,SP,TAB' ONLY? A CHARACTER TYPED WITHIN QUOTES WHEN TRYING TO DEFINE THE CONTENTS OF A TRANSMIT OR EXPECT MESSAGE WAS NOT A 'A-Z,0-9,SPACE OR TAB'. RETYPE THE COMMAND WITH ONLY THESE CHARACTERS BETWEEN QUOTES.

? 'SIZE=0' NOT VALID? A MESSAGE ZERO BYTES LONG CAN NOT BE BUILT. RETYPE THE COMMAND WITH A '/SIZE=NNN'. IF NO '/SIZE=' IS TYPED A DEFAULT SIZE WILL BE USED.

? TRIB CMDS ILLEGAL IN PT-PT MODE? A TRIB COMMAND WAS ISSUED AND THE MODE DEFINED BY THE HARDWARE P-TABLE WAS POINT TO POINT. IF TRIB COMMANDS ARE TO BE USED PROGRAM MUST BE STARTED AGAIN WITH THE MODE SET TO MULTIPOINT

? TRANSMIT AND EXPECT LIST MUST BE IDENTICAL FOR LOOP?
IF RUN COMMAND WITH '/LOOP/CH' IS TYPED TRANSMIT AND EXPECT LISTS MUST BE EQUAL. USE 'SE E=T' COMMAND.

? TRIB ADDRESS= XXX IS NOT UNIQUE? THE TRIB WHOSE ADDRESS IS XXX IS ALREADY IN THE TRIB LIST

? TRIB ADDRESS= XXX NOT FOUND? THE TRIB WHOSE DECIMAL ADDRESS IS XXX WAS NOT FOUND IN THE TRIB LIST WHEN THE TRIB KILL COMMAND WAS EXECUTED

? CABLE,LOC,REM LOOP NOT VALID IN 'MULTIPOINT MODE'? A RUN COMMAND WAS ISSUED WITH LOOP= TO CABLE,LOCAL OR REMOTE WHILE THE MODE SET BY THE P-TABLE WAS MULTIPOINT THESE LOOP MODES ARE ONLY VALID FOR POINT TO POINT OPERATION

? TRIBS MUST BE ESTABLISHED TO EXECUTE? A RUN COMMAND WAS ISSUED IN MULTIPOINT MODE AND THE TRIB LIST WAS EMPTY. TO USE MULTIPOINT MODE A LEAST ONE TRIB MUST BE ESTABLISHED.

? TRIB STATION CANNOT DO LOOP? A RUN COMMAND WAS ISSUED WITH THE LOOP

SWITCH AND THE MODE IN P-TABLE WAS
MULTIPOINT TRIBUTARY. TRIBUTARY
STATIONS CANNOT DO LOOPBACK.

?ONLY ONE TRIB (TRIB ADDR 1) ALLOWED
FOR LOOP IN MULTIPOINT?

A RUN COMMAND WITH LOOP=INTERNAL
WAS ISSUED AND THE TRIB LIST DID
NOT HAVE ONLY 1 TRIB IN IT. IF IT
DID HAVE ONLY 1 TRIB IN IT THE ADDRESS
WAS NOT 1.

?TRIB ADDRESS= XXX INVALID?

A TRIB COMMAND WAS ISSUED WITH A TRIB
ADDRESS NOT IN THE RANGE 1-255

3.2.2 DCLT ERROR MESSAGES:

CLOCK NOT FOUND

THIS MEANS THAT NO CLOCK WAS FOUND
ON THE SYSTEM THE DIAGNOSTIC WILL
STILL RUN BUT NONE OF THE TIME OUT
CONDITIONS WILL OCCUR.

BAD CLOCK - PROGRAM WILL HANG ON "TIMEOUT"!!

THIS MEANS THAT EITHER NO CLOCK WAS
ON THE SYSTEM OR THE ONE THAT WAS FOUND
DID NOT INTERRUPT WHEN ASKED TO DO A
"TICK".
THE PROGRAM WILL STILL RUN, BUT ANY
OF THE PROGRAM THAT TIMES THE DEVICE
WILL HANG IF THE DEVICE TIMES OUT.
ALSO, THE EVENT LOG WILL CONTAIN A
ZERO EVENT TIME FOR ALL EVENTS LOGGED.

MAX. CHAR. MSG COUNT EXCEEDED - MSG. NOT BUILT !!

THIS MEANS THAT THE TRANSMIT OR EXPECT
BUFFER IS FULL. NO MORE MESSAGES CAN BE
ADDED TO THAT BUFFER.

BUFFER FULL - MSG. NOT BUILT !!

THIS MEANS THAT THE LAST MESSAGE YOU
TRIED TO ADD TO EITHER THE TRANSMIT OR
EXPECT BUFFER CAUSED THE TOTAL NUMBER
OF MESSAGES TO BE EXCEEDED. NO MORE
MESSAGES CAN BE ADDED TO THAT BUFFER.
THE LIMIT IS DETERMINED BY THE SIZE OF
THE MESSAGE POINTER TABLE. THE LIMIT
IS CURRENTLY 15.

CHAR. COUNT EXCEEDS BUFF LIMIT - MSG TRUNCATED

THIS MEANS THAT THE LAST MESSAGE YOU
TRIED TO ADD TO THE TRANSMIT OR EXPECT
BUFFER CAUSED THE TOTAL CHAR. COUNT
FOR THAT BUFFER TO EXCEED THE LIMIT.

TRIB ADDRESS LIST IS EMPTY

THE LIMIT IS 512. BYTES.
THE MESSAGE WAS TRUNCATED TO COMPLETELY
FILL THE BUFFER. NO MORE MESSAGES CAN
BE ADDED TO THAT BUFFER.

THERE ARE NO TRIBS IN THE TRIB LIST
WHEN THE THE TRIB SHOW COMMAND WAS
EXECUTED.

TRIB ADDRESS LIST FULL - ADDRESS= XXX NOT ADDED

A TRIB ESTABLISH COMMAND CAUSED
THE NUMBER OF TRIBS IN THE LIST TO
EXCEED THE MAXIMUM (DMV=12,DMP=32).
THIS ERROR MESSAGE IS REPEATED FOR
ALL TRIBS IN EXCESS FOR THIS STRING.
XXX= THE DECIMAL ADDRESS OF THE TRIB

RX BUFFER NOT BIG ENOUGH
TOO MANY TRIBS OR MSGS

A RUN COMMAND WAS ISSUED WITH
DATA CHECKING REQUESTED AND THE
NUMBER OF TRIBS TIMES THE NUMBER
OF EXPECTED MESSAGES EXCEEDED THE
MAXIMUM REC BUFFER TOTAL (2048 BYTES)
TO CORRECT FOR THIS EITHER THE NUMBER
OF MESSAGES, THE SIZE OF THE MESSAGE OR
THE NUMBER OF TRIBS MUST BE DECREASED.

3.2.3 DEVICE ERROR MESSAGES:

DATA COMPARISON DATA ERROR
BYTE # IN MSG=XXX EXPTD=YYY

RECD=ZZZ
XXX= OFFSET OF THAT BYTE FROM THE START
OF THE COMPARE OR EXPECT MESSAGE.
YYY= THE CONTENTS OF THAT BYTE IN THE
EXPECTED MESSAGE
ZZZ= THE CONTENTS OF THAT BYTE IN THE
RECEIVED MESSAGE

UP TO FIVE OF THESE ERRORS WILL BE
PRINTED PER MESSAGE COMPARED. ONLY
THE FIRST FIVE MISMATCHES WILL BE
INDIVIDUALLY REPORTED, BUT TOTAL
NUMBER OF MISMATCHES IS REPORTED
BY ANOTHER ERROR.

PRINTING THE EVENT LOG AND USING THE
DCLT 'DUMP' COMMAND WILL ALLOW YOU TO
FIND THE ADDRESS OF THE MESSAGE AND
EXAMINE IT.

DATA COMPARISON DATA ERROR
TOTAL MISMATCHES IN MSG = NNN

THIS MEANS THAT WHEN THE MESSAGE
RECEIVED WAS COMPARED AGAINST THE
MESSAGE THAT WAS EXPECTED, SOME OF

THE CHARS. WERE NOT THE SAME.

DATA COMPARISON LENGTH ERROR
COMPARE COUNT= XXX RECEIVE COUNT= ZZZ

XXX= NUMBER OF BYTES IN THE COMPARE
MESSAGE
ZZZ= NUMBER OF BYTES IN THE RECEIVED
MESSAGE
THIS MEANS THAT THE MESSAGE RECEIVED
WAS A DIFFERENT LENGTH THEN THE MESSAGE
THAT WAS EXPECTED.

* NOTE * - IN THE FOLLOWING ERROR DESCRIPTIONS XXXXX
***** REFERS TO THE OCTAL CONTENTS OF THE DEVICE REGISTERS
SPECIFIED.

DEVICE DID NOT RETURN RUN BIT
SEL0 SEL2
XXXXXX XXXXXX

:THIS ERROR INDICATES
:THAT THE DEVICE DID
:NOT RETURN THE RUN BIT
:AFTER 1000 TICKS OF THE CLOCK
:COULD INDICATE MICRO-DIAG
:FOUND A FAILURE.

FAILURE IN MICRO DIAGNOSTICS
SEL0 SEL6
XXXXXX XXXXXX

:THIS ERROR INDICATES THAT
:BSEL6 DOES NOT CONTAIN 305
:THIS IS CHECKED AFTER A MASTER
:CLEAR AND THE RUN BIT HAS
:BEEN SET

TIME OUT WAITING FOR TX OR RX TO COMPLETE
SE_0 SEL2
XXXXXX XXXXXX

:THIS ERROR IS THE MOST POPULAR
:IT INDICATES THAT THE 60 SEC
:TIMER EXPIRED WHEN THE DEVICE
:WAS EXPECTING TO GET A RX OR
:TRANSMIT COMPLETE. AFTER THIS
:ERROR OCCURS THE PROGRAM WILL
:RESET THE TIMER AND LOOP AGAIN

TIME OUT WAITING FOR RDI
SEL0 SEL2
XXXXXX XXXXXX

:THIS ERROR INDICATES THAT THE
:DEVICE DID NOT RETURN RDI IN
:RESPONSE TO AN RQI BEFORE THE
:TIMER EXPIRED. THE TIMER IS
: 100 TICKS FOR DMP AND 400
: TICKS FOR THE DMV.

CONTROL OR INFORMATION OUT ERROR

SEL2 SEL6
 XXXXXX XXXXXX YYYYYY

:THIS ERROR INDICATES THAT
 :A CONTROL OUT ERROR OCCURRED
 :OR AN UNEXPECTED INFORMATION
 :OUT OCCURRED. THE TYPE OF
 :ERROR IS INDICATED
 :BY THE ASCII
 :STRING YYYYYY WHICH CAN BE ONE
 :FROM THE LIST BELOW.
 :SOME CONTROL OUTS ARE FATAL
 :IF A FATAL ERROR OCCURS THE
 :PROGRAM WILL BE FORCED TO THE
 :DCLT> PROMPT THE FATAL ERRORS
 :ARE INDICATED BELOW

MSG ---	FATAL -----	DESCRIPTION -----
SELECT THRESHOLD	NO	SELECTION TIMER TIMED OUT MORE THAN 7 TIMES
START RXD IN RUN	YES	DDCMP START RX'D WHILE DEVICE WAS IN RUN STATE
MAINT RXD IN RUN	YES	DDCMP MAINT MESSAGE WAS RX'D WHILE DEVICE WAS IN THE RUN STATE
MAINT RXD IN HALT	YES	DDCMP MAINTINANCE MSG RX'D WHEN DEVICE WAS IN HALT STATE.
START RXD IN MAINT	YES	DDCMP START MSG RX'D WHILE DEVICE WAS IN MAINTINANCE MODE
RING DETECTED	NO	RING SIGNAL WAS SET BY MODEM. THIS OUTPUT FOR DMP ONLY.
DEAD TRIB	NO	INDICATES THAT A TRIB NO LONGER RESPONDS WHEN IT IS POLLED
RUN STATE ERR	NO	RUN STATE OUTPUT IS POSTED WHEN DCLT IS NOT EXPECTING IT.
BABBLING TRIB	YES	A TRIBUTARY IS HOGGING THE LINE AND NOT RETURNING THE SELECT FLAG.
STREAMING TRIB	YES	A TRIBUTARY IS SENDING DATA CONSTANTLY.
BUFFER TOO SMALL	YES	MSG WAS RX'D AND THE

NON EXIST MEM	YES	DEVICE HAS NO BUFFER BIG ENOUGH FOR IT. THIS IS PROBABLY OPER- RATER ERROR.
DISCONNECT	YES	INDICATES THAT DEVICE TRIED TO NPR TO A MEM LOCATION THAT IS NON EXISTENT.
QUEUE OVER	YES	INDICATES DEVICE SAW MODEM READY GO AWAY AFTER BEING SET. LOOK FOR CABLE OR MODEM
CARRIER LOSS	YES	DEVICE HAS TOO MUCH OUTPUT OR PROGRAM GAVE DEVICE TOO MUCH.
		INDICATES CARRIER SIG WENT AWAY WHILE RX'ING

**NOTE THE FOLLOWING ARE PROCEDURE ERRORS IF THEY OCCUR
**THE DEVICE IS PROBABLY BAD ALL PROCEDURE ERRORS ARE FATAL

NO MODE DEF	YES	PROCEDURE ERROR
ILLEGAL TYPE CODE		
MODE CHANGE		
CONTROL IN TO UNES. TRIB		
COMMAND TO TRIB 0		
COMMAND TO UNHALTED TRIB		
MAX TRIBS EXCEEDED		
ESTB TO ALREADY ESTABLISHED		
ILLEGAL REQUEST KEY		
ASSIGN BUFF UNEST. TRIB		
ASSIGN BUFF HALTD TRIB		
ASSIGN BUFF BYTE CNT 0		
ASSIGN TX BUFF TRIB 0		
R OR W RESERVED TSS		
USE RESERVED BIT IN BSEL7		
COMMON POOL ERROR		
QUOTA OVERFLOW		

**** END OF PROCEDURE ERRORS*****

ILLEGAL TRANSMIT COMPLETE SEL4 SEL6 XXXXXX XXXXXX	:INDICATES DEVICE GOT A TX :COMPLETE WHEN IT WAS NOT :EXPECTING IT.
ILLEGAL RECEIVE COMPLETE SEL4 SEL6 XXXXXX XXXXXX	:INDICATES DEVICE GOT A RX :COMPLETE WHEN IT WAS NOT :EXPECTING IT.
QUE OVERFLOW BUFFER COMPLETE SEL4 SEL6 XXXXXX XXXXXX	:INDICATES A BUFFER COMPLETE :WAS TAKEN FROM THE QUE AFTER :A QUE OVERFLOW
RLD OR MODE ENABLE OF PASSWORD SW NOT SET SEL0 SEL2 XXXXXX XXXXXX	:INDICATES THAN WHEN IN DLL :MODE ON A TRIB THE SWITCHES :ON THE DEVICE WERE NOT SET :CORRECTLY.
DOWN LINE LOAD ABORTED RXBUF TXBUF ZZZZZ XXXXXX YYYYYY	:WHEN RUNNING DOWN LINE LOAD :THE HOST HAS SENT A 'ENTER :MOP' MSG AND IS WAITING FOR :RESPONSE. ZZZZZ IS FIRST WORD :OF REC BUFFER XXXXX FIRST WORD :OF TX BUFFER AND YYYYYY IS AN :ASCII STRING THAT INDICATES :ONE OF THE FOLLOWING
TX NOT COMPLETE	:THE FIRST COMPLETE WAS :NOT A TRANSMIT COMPLETE
RX NOT COMPLETE	:THE SECOND COMPLETE WAS :NOT A RX COMPLETE
SEC REQ ERR WORD 1	:THE TX AND RX COMPLETE :HAPPEN BUT THE FIRST WORD :OF THE "SECONDARY BOOT" :MSG IS IN ERROR
SEC REQ ERR WORD 2	:THE TX AND RX COMPLETE HAPPEN :AND THE FIRST WORD IS GOOD IN :THEN "SECONDARY BOOT" MMSG BUT :THERE IS AN ERROR IN :BYTES 3 OR 4.

4.0 PERFORMANCE AND PROGRESS REPORTS

DCLT USES IT'S OWN METHOD FOR DETERMINING AN 'END OF PASS' WHICH IS CALLED A 'DCLT END OF PASS'. THE NUMBER OF 'DCLT PASSES' TO BE RUN IS SPECIFIED BY THE '/PASS=XXX' SWITCH ON THE DCLT RUN COMMAND. THE TOTAL NUMBER OF 'DCLT ERRORS' ARE LOGGED IN IN THE EVENT LOG WHEN EACH 'DCLT PASS' IS COMPLETED.

4.1 PRINTING OF EVENT LOG

SIGNIFICANT EVENTS OR CHECK-POINTS WILL BE LOGGED IN A 'CIRCULAR QUEUE' STORAGE AREA CALLED THE EVENT LOG. THE LAST 45 EVENTS ARE KEPT LOGGED AND CAN BE LISTED ON THE OPERATORS CONSOLE BY GIVING A 'PRINT' COMMAND AT THE 'DR>' (DIAGNOSTIC SUPERVISOR) OR 'DCLT>' (DCLT) LEVEL. THE PRINT COMMAND MUST BE FOLLOWED BY A LOG COMMAND. THE EVENTS ARE PRINTED IN A 'LAST-IN FIRST-OUT' ORDER.

EVENT TIME IS TYPED OUT AS MMM:SS:TT (LIKE 254:36:07) WHERE MMM,SS,TT REPRESENT THE NUMBER OF MINUTES, SECONDS, CLOCK TICKS SINCE THE LAST START OR RESTART. IT SHOULD BE NOTED THAT THE TIMES ARE RELATIVE SINCE WHILE THE PROCESSOR IS RUNNING AT PRIORITY 7 THE CLOCK CAN'T INTERRUPT TO KEEP TIME. THIS IS THE CASE WHILE THE PROGRAM IS FETCHING DCLT COMMANDS FROM THE OPERATOR. IT SHOULD ALSO BE NOTED THAT THERE ARE ONLY 8 BITS AVAILABLE TO STORE RELATIVE MINUTES SO 'TIME' WILL WRAP TO 000:00:00 AFTER 256:59:59.

A START OR RESTART COMMAND AT THE 'DR>' LEVEL INITIALIZES THE EVENT LOG. THEREFORE IT IS WISE TO DO A 'PRINT' 'LOG' AT THE 'DR>' LEVEL BEFORE GIVING A 'START' OR 'RESTART'.

THE TYPES OF EVENTS KEPT IN THE EVENT LOG ARE:

TRANSMIT MESSAGE QUEUED:

EVENT TIME, ADDRESS OF TRIBUTARY TO/FROM
ADDRESS OF 1ST BYTE OF MESSAGE,
TOTAL NO. OF BYTES, MODEM STATUS AT THAT TIME.

TRANSMIT MESSAGE COMPLETED:

EVENT TIME, ADDRESS OF TRIBUTARY TO/FROM
ADDRESS OF 1ST BYTE OF MESSAGE,
TOTAL NO. OF BYTES, MODEM STATUS AT THAT TIME.

RECEIVE SPACE QUEUED:

EVENT TIME, ADDRESS OF TRIBUTARY TO/FROM
ADDRESS OF 1ST BYTE OF MESSAGE,
TOTAL NO. OF BYTES, MODEM STATUS AT THAT TIME.

RECEIVE MESSAGE COMPLETED:

EVENT TIME, ADDRESS OF TRIBUTARY TO/FROM
ADDRESS OF 1ST BYTE OF MESSAGE,
TOTAL NO. OF BYTES, MODEM STATUS AT THAT TIME.

DATA COMPARISON STARTED:

EVENT TIME, ADDRESS OF TRIBUTARY TO/FROM
ADDRESS OF 1ST BYTE OF RECEIVED MSG.,
TOTAL NO. OF BYTES IN RCV. MSG., TOTAL NO. OF BYTES
IN EXPECT MSG.

DATA COMPARISON DATA ERROR:
EVENT TIME, ADDRESS OF TRIBUTARY TO/FROM
ADDRESS OF 1ST BYTE OF RECEIVED MSG.,
TOTAL NO. OF BYTES IN RCV. MSG., TOTAL NO. OF
COMPARISON FAILURES
DATA COMPARISON LENGTH ERROR:
EVENT TIME, ADDRESS OF TRIBUTARY TO/FROM
ADDRESS OF 1ST BYTE OF RECEIVED MSG.,
TOTAL NO. OF BYTES IN RCV. MSG., TOTAL NO. OF BYTES
IN EXPECT MSG.
DEVICE INIT AND SETUP:
EVENT TIME, MODE OF OPERATION, TYPE OF MAINTENANCE
LOOP, 'DCLT' PASS COUNT, 'RUN' PARAMETERS
DEVICE ERROR:
EVENT TIME, DEVICE ERROR MESSAGE, CONTENTS OF TWO
REGISTERS RELATING TO THE ERROR.
END OF PASS:
EVENT TIME, 'DCLT' PASS COUNT, 'DCLT' ERROR COUNT,
#OF RX THRESHOLD ERRORS, # OF TX THRESHOLD ERRORS

NOTE - RX THRESHOLDS AND TX THRESHOLDS OCCUR IF
ONE STATION IS STARTED BEFORE THE OTHER
OR IF LINKS ARE RUN AT HIGH SPEED

4.2 OPERATOR STATUS MESSAGES

THE '/STATUS, /NOSTATUS' QUALIFIERS FOR THE DCLT 'RUN' COMMAND
ENABLES/DISABLES THE PRINTING OF PROGRAM STATUS MESSAGES TO THE
OPERATOR. THESE MESSAGES ARE INTENDED TO TELL THE OPERATOR WHAT
THE DCLT PROGRAM IS CURRENTLY DOING. BELOW ARE THE MESSAGES THAT
MIGHT BE PRINTED AND THEIR MEANING:

MESSAGE	MEANING
TXQ	DEVICE IS ABOUT START TRANSMITTING A MESSAGE
TXC	TRANSMISSION OF MESSAGE COMPLETED
RXQ	DEVICE HAS QUEUED SPACE TO RECEIVE/ COMPLETED RECEIVE
ERR	DEVICE ERROR HAS OCCURRED
INI	DEVICE ABOUT TO BE INITIALIZED
CMP	ABOUT TO DO DATA CHECKING OF RECVD VS. EXPTD DATA
CML	LENGTH ERROR OCCURRED DURING DATA COMPARISON
CMD	DATA ERROR OCCURRED DURING DATA COMPARISON
EOP	END OF PASS

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. [10]) 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	1	:	[0]	FULL OR HALF DUPLEX FLAG (BIT0=1 IF FULL)	
.WORD	160170	:	[2]	CSR ADDRESS	
.WORD	300	:	[4]	INTERRUPT VECTOR	
.WORD	240	:	[6]	INTERRUPT PRIORITY (5)	
.WORD	0	:	[10]	DEVICE PARAMS BIT1	BIT0
		:		IF A ZERO TRIB	POINT-POINT
		:		IF A ONE CONTROL	MULTIPOINT
.WORD	0	:	[12]	OPTION TYPE 0=DMP 1=DMV	
		:			

6.0 MODE AND MESSAGE DESCRIPTIONS

THE FOLLOWING ABBREVIATIONS WILL BE USED IN THE MODE DESCRIPTIONS
MTP/TB - MULTIPOINT TRIBUTARY
MTP/CS - MULTIPOINT CONTROL STATION
PTP - POINT TO POINT

6.1 MODE DESCRIPTIONS

6.1.1 TRANSMIT MODE

IF PTP OR MTP/TB:

THE TRANSMIT LIST OF MESSAGES IS TRANSMITTED WITHOUT EXPECTING ANY DATA TO BE RECEIVED.

IF MTP/CS: THE LIST IS SENT TO EACH TRIBUTARY

6.1.2 RECEIVE MODE

IF PTP OR MTP/TB:

SPACE IS QUEUED FOR THE DEVICE TO RECEIVE MESSAGES. AFTER RECEIVING AN 'EXPECTED' NUMBER OF MESSAGES, THE DATA RECEIVED CAN BE COMPARED AGAINST A LIST OF 'EXPECT TO RECEIVE' MESSAGES IF DATA-CHECKING IS ENABLED.

IF MTP/CS: SPACE IS QUED FOR ALL TRIBUTARIES

6.1.3 PASSIVE MODE

IF PTP OR MTP/TB:

EVERY TIME A MESSAGE IS RECEIVED, A MESSAGE IS TRANSMITTED. DATA CHECKING CAN BE DONE ON THE RECEIVED DATA. THE "/ECHO, /NOECHO" ENABLES/DISABLES THE RETRANSMISSION OF THE DATA RECEIVED.

IF MTP/CS: A MESSAGE IS RECEIVED FROM EACH TRIB AND THEN A MESSAGE IS TRANSMITTED TO EACH TRIB.

6.1.4 ACTIVE MODE

A LIST OF MESSAGES IS TRANSMITTED AND MESSAGES ARE RECEIVED. AFTER RECEIVING AN "EXPECTED" NUMBER OF MESSAGES, THE DATA RECEIVED CAN BE COMPARED AGAINST A LIST OF "EXPECT TO RECEIVE" MESSAGES IF DATA-CHECKING IS ENABLED.

IF MTP/TB: THE TRANSMIT MESSAGES OF ALL TRIBS MUST BE IDENTICAL IF DATA CHECKING IS ENABLED.

NOTE: IF BOTH ENDS OF THE LINK ARE IN ACTIVE MODE, THEN THE LINK MUST BE A FULL DUPLEX LINK!

6.1.5 DOWN-LINE-LOAD

* NOTE * - THE SATELLITE IN MTP MODE WILL ALWAYS BE THE FIRST
TRIB IN THE TRIB LIST.

IF IN PTP MODE, THE SATELLITE WILL ENTER MOP MODE ONLY IF THE PASSWORD SUPPLIED BY THE USER MATCHES THAT SET IN ITS PASSWORD SWITCH PACK.

IF PTP OR MTP/CS:

THE "HOST" REQUESTS THE "SATELLITE" TO ENTER MOP MODE. THE SATELLITE THEN SENDS A "SECONDARY BOOT REQUEST MESSAGE". THE "HOST" THEN CHECKS THE RECEIVED MESSAGE TO SEE THAT IT IS A "SECONDARY BOOT REQUEST". THEN THE HOST SENDS A "MEMORY LOAD WITH TRANSFER ADDRESS" THAT CONTAINS IMAGE DATA TO BE LOADED BY THE SATELLITE'S MICRO-CODE INTO MAIN MEMORY STARTING AT LOC. 0. THIS IMAGE DATA WILL CONTAIN CODE THAT PRINTS A MESSAGE STATING DOWN-LINE-LOAD WAS SUCCESSFUL. THE BOOTING PROCESS OVERWRITES PART OF THE "VECTOR" AREA SO THE DCLT PROGRAM MUST BE RELOADED IN THE "SATELLITE" SYSTEM.

IF MTP/TB:

RUNNING DOWN LINE LOAD MODE IN A MULTIPOINT TRIB JUSTS ENABLES PRIMARY MOP MODE.
TRIBS CANNOT BE "HOSTS"

* NOTE * - THE SATELLITE MUST HAVE CERTAIN SWITCHES SET ON
***** THE LINE UNIT CARD IN ORDER TO ALLOW THE BOOT TO
OCCUR. THE MODE ENABLE SWITCH [SW 8 OF E121] MUST
BE SET TO A 1[OFF]. THE MODE MUST BE DEFINED IN THE
SWITCHES[SW'S 5 6 AND 7 OF E-121]. THE PASSWORD OR
TRIB ADDRESS MUST BE SET IN THE SWITCHES[SW'S IN
E-134]. THIS MUST BE DONE FOR ALL TYPES
OF DOWN LINE LOAD, IN ADDTION THE FOLLOWING MUST
BE DONE FOR.

REMOTE LOAD DETECT:
SWITCH 9 OF E-121 TO A ONE [OFF]
FOR POWER ON BOOT AND ENTER P MOP
SWITCH 10 OF E-121 TO A ZERO [ON]

INCLUDED IN THE "SECONDARY BOOT MESSAGE" IS THE
DEVICE TYPE CODE THAT IS DECIPHERED AND INCLUDED
IN AN IDENTIFICATION MESSAGE.

EXAMPLE:

SECONDARY BOOT REQ FROM XXX DEVICE TYPE = YY

YY	XXX
--	---
0	DP
2	DU
4	DL
6	DQ
8	DA
10	DUP
12	DMC
14	DN
16	DLV
18	DMP
20	DTE
22	DV
24	DZ
28	KDP
30	KDZ
32	KL
34	DMV

6.1.6 TALK AND LISTEN MODE

* NOTE * - IN MTP MODE TALK AND LISTEN USE ONLY THE FIRST TRIB
***** IN THE TRIB LIST

6.1.6.1 TALK MODE

THE "TALK" END OF THE LINK TRANSMITS OPERATOR-TYPED MESSAGES
UNTIL A "EXIT" MESSAGE IS TYPED. AT THAT POINT, THE NODE GOES

INTO 'LISTEN' MODE. AN 'EXIT MESSAGE' IS A MESSAGE WHOSE FIRST FOUR CHARACTERS ARE 'EXIT'. SINCE ONLY THE FIRST FOUR CHARACTERS NEED TO BE 'EXIT', MORE CHARACTERS CAN BE ADDED SO THAT A MESSAGE MAY BE SENT AND THE MODE SWITCHED ALL AT ONCE. FOR EXAMPLE:

TLK> EXIT ALL OF THIS LINE IS SENT THEN MODE SWITCHED

6.1.6.2 LISTEN MODE

THE 'LISTEN' END OF THE LINK PRINTS ALL OF THE MESSAGES RECEIVED BY THE DEVICE ON THE OPERATOR'S CONSOLE. IF THE MESSAGE RECEIVED IS AN 'EXIT' MESSAGE, THEN THE NODE ENTERS 'TALK' MODE. AN 'EXIT MESSAGE' IS A MESSAGE WHOSE FIRST FOUR CHARACTERS ARE 'EXIT'.

6.1.7 MAINTENANCE 'LOOP' MODES

REMEMBER THAT THE WHENEVER A 'RUN' COMMAND IS TYPED, THE DEFAULT IS NO LOOPBACK AND THAT A LOOP MODE MUST BE SPECIFIED BY A '/LOOP=..' IF A LOOP MODE IS DESIRED.
LOOP MODES ARE ONLY VALID IF THE MODE TO RUN IS ACTIVE

INTERNAL TTL LOOPS DATA INTERNALLY THIS WILL NOT WORK FOR MTP/TB. IF MTP/CS THEN TRIB 1 MUST BE ESTABLISHED.

THE FOLLOWING ARE ONLY VALID IN PTP MODE.

CABLE DOES NOT CAUSE ANY BITS TO BE SET OR REQUESTS TO BE QUEUED, BUT MAKES FOR A NICE BOOKKEEPING FEATURE. '/L=CABLE' WILL THEN BE SHOWN WHEN THE COMMAND LINE IS TYPED AS A REMINDER OF WHAT TYPE OF LOOPING IS BEING ATTEMPTED. REMEMBER TO INSTALL ANY CONNECTORS OR ENABLE ANY LOOP FEATURES THAT ARE NECESSARY TO MAKE CABLE LOOPBACK POSSIBLE.

LOCALMODEM SETS MM1 ON INTERFACE ALSO CALLED ANALOG-LOOPBACK.

REMOTEMODEM SETS MM2 ON RS449 INTERFACE ALSO CALLED DIGITAL-LOOPBACK.

6.1.8 MODE SUMMARY TABLE

THE FOLLOWING TABLE SUMMARIZES THE MODES THAT CAN BE RUN TOGETHER WHEN THE DCLT PROGRAM IS RUNNING ON TWO PROCESSORS (ONE AT EACH END OF THE LINK):

STATION A 'HOST' NODE	STATION A '/LOOP' ALLOWED?	STATION B 'REMOTE' NODE	DUPLEX
TALK	NO	LISTEN*, RECEIVE	HALF OR FULL
LISTEN	NO	TALK*, TRANSMIT	HALF OR FULL
TRANSMIT	NO	RECEIVE*, LISTEN	HALF OR FULL
RECEIVE	NO	TRANSMIT*, TALK	HALF OR FULL
PASSIVE	NO	ACTIVE*	HALF OR FULL
ACTIVE	YES	ACTIVE*	FULL
ACTIVE	YES	PASSIVE*	HALF OR FULL
DOWNLINELOAD	NO	PASSIVE*	HALF FORCED

*= MOST LIKELY TO BE IN THAT MODE

6.2 MESSAGE DESCRIPTIONS

NAME	DESCRIPTION
ZEROES	MESSAGE OF ALL 0'S (00000000,00000000,00000000,...)
ONES	MESSAGE OF ALL 1'S (11111111,11111111,11111111,...)
1ALT	MESSAGE OF ALTERNATING 1'S (10101010,10101010,...)
0ALT	MESSAGE OF ALTERNATING 0'S (01010101,01010101,...)
CCITT	"CCITT" 512-BIT (VS. 511 BITS) TEST PATTERN
ITEP	"INTERPROCESSOR TEST PROGRAM'S (ITEP)" MESSAGE 1(DP1:) (<177><177>/SA THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.<15><12><001><177><177><177><177>)
ALPHA	ALPHA-NUMERICS (OR FUTURE COMM TURNAROUND MSG) (#\$!' (AMPERSAND)'()*+,-.0123456789:;<=>?@ABCDEFGHIJK LMNOPQRSTUVWXYZ/[\] ^ _ %)
OPERATOR-SPECIFIED	'A-Z,0-9,SPACES,TABS' THESE ARE THE CHARACTERS THAT CAN BE TYPED BETWEEN QUOTATION MARKS ('..') TO SPECIFY A UNIQUE MESSAGE.

7.0 OTHER INFORMATION

7.1 INTERFACING TO AN 'ITEP' NODE

THIS DCLT WILL INTERFACE ONLY TO THE ITEP FOR DMC.
IF THIS LINK IS NEEDED THEN THE DMP/V-11 MUST BE IN POINT
TO POINT MODE AND THE FOLLOWING TABLE APPLIES TO THE ITEP
NODE:

ITEP NODE	DCLT NODE
ONE-WAY-OUT	RECEIVE OR LISTEN
ONE-WAY-IN	TRANSMIT OR TALK
INTERNAL LOOP	ACTIVE
EXTERNAL LOOP	ACTIVE OR PASSIVE

NOTE: WHEN INTERFACING TO ITEP IF THE RX BUFFER ON THE
ITEP SIDE IS ONLY 10 BYTES LARGER THAN THE TX BUFFER YOU
HAVE SELECTED, SO BE SURE TO SET THE TX BUFFER ON THE DCLT
NODE ACCORDINGLY.

WHEN ITEP IS IN A MODE THAT IT IS EXPECTING TO BE TRANSMITTED
TO, A SOFT ERROR 'BASE TABLE ERR COUNTS NON-ZERO' WILL OCCUR.
THIS IS DUE TO THE SPEED DIFFERENCES IN THE SOFTWARE.

WHEN DCLT IS IN LISTEN MODE THE RX BUFFER IS ONLY
82 BYTES LONG THEREFORE DO NOT SEND THE DCLT NODE
ITEP MSG. 3 FROM THE ITEP NODE OR A 'LOST DATA' ERROR WILL
OCCUR

BE SURE ITEP NODE HAS INCORPORATED PATCH FROM DEPO# MD-11-DZDMO-A1

ITEP NODE SHOULD ALWAYS BE RUN WITH SW 4 = TO 0

7.2 TROUBLESHOOTING HINTS

LISTED BELOW ARE SOME SETUPS THAT COULD BE USED FOR ISOLATING FAULTS. THESE ARE BY NO MEANS THE ONLY WAYS DCLT CAN BE USED !!!!!!! DCLT IS MEANT TO BE A VERY FLEXIBLE TOOL! THIS SECTION IS MEANT TO GIVE SOMEONE NOT TOO FAMILIAR WITH DCLT A PLACE TO START.

REMEMBER THAT THE PRINTING OF STATUS MESSAGES AND PRINTING OF THE EVENT LOG CAN PROVIDE A LOT OF INFORMATION ABOUT THE SEQUENCE OF EVENTS AND HOW THE DEVICE AND LINK ARE BEHAVING.

NOTE: IF BOTH NODES IN ACTIVE AND "/NOCHECK" IS USED, ----- END-OF-PASS IS DEFINED AS RECEIVING 1 MESSAGE AND COMPLETING THE TRANSMIT LIST. WITH NO DATA CHECKING, THERE IS NO WAY FOR DCLT TO KNOW HOW MANY MESSAGES IT SHOULD EXPECT TO RECEIVE.

7.2.1 INTERNAL LOOP AT EACH NODE

RUN EACH END OF THE LINK IN ACTIVE MODE WITH LOOP=INTERNAL. TRANSMIT TWO OR THREE MESSAGES WITH NO DATA CHECKING. STATUS PRINTING COULD BE TURNED OFF IF ON, BUT SEEING THE SEQUENCE OF EVENTS MIGHT BE INFORMATIVE.

INTERNAL LOOP WORKS ONLY FOR POINT TO POINT OR MULTIPPOINT CONTROL STATIONS. THE SEQUENCE BELOW IS FOR POINT TO POINT IF YOU WISH TO DO MULTIPPOINT ADD THE COMMAND WITH THE *

```
C E
C T
SE T=ONES/S=20/C=2
* T E=1
R M=A/LO=1/NOCH/STAT
```

WHAT THE ABOVE COMMAND SEQUENCE MEANS:

THE "C E" AND THE "C T" INITIALIZES THE "EXPECT" LIST AND THE "TRANSMIT LIST". THE "SE T=ONES/S=20/C=2" SETS THE TRANSMIT LIST TO CONTAIN 3 MESSAGES. THE MESSAGES CONTAIN DATA OF ALL ONES AND EACH ONE IS 20 BYTES IN LENGTH. THE "T E=1"(ONLY FOR MTP) ESTABLISHES ONE TRIB ,TRIB ADDRESS 1. THE "R M=A/LO=1/NOCH/STAT" SETS THE MODE TO RUN IN TO BE ACTIVE AND LOOP TYPE TO BE INTERNAL TTL. THE PROGRAM WILL NOT BE CHECKING DATA SO THERE WAS NO NEED TO SET UP AN EXPECT LIST. THE PROGRAM WILL BE PRINTING STATUS MESSAGES.

WHAT YOU SHOULD SEE AFTER ENTERING THE RUN COMMAND IF THINGS ARE RUNNING CORRECTLY :

```
INI RXQ TXQ RXQ TXC TXQ RXQ TXQ
RXQ TXC EOP
```

```
MODE=ACTIVE/LOOP=INTERNAL/PASS=00000  
/STATUS/NOCHECK/NOECHO/NOMODEM
```

DCLT> (A) ?

THIS GIVES YOU A IDEA IF THE COMM. DEVICE CAN TRANSMIT AND RECEIVE. ANY ERRORS REPORTED WILL PROBABLY BE DUE TO INCORRECT DEVICE ADDRESSES BEING USED OR A FAULTY DEVICE. CHECK ADDRESSES WITH 'DISPLAY' AND RUN THE PREREQUISITE DIAGNOSTICS FOR THE COMM. DEVICE.

NOW TRY RUNNING EACH NODE THE SAME WAY WITH DATA CHECKING ENABLED. A POSSIBLE COMMAND SEQUENCE IS:

```
SE E=T  
R M=A/LO=1/CH/PAS=3
```

WHAT THIS SEQUENCE MEANS:

THIS SEQUENCE IS SIMILAR TO THE ONE ABOVE. THE 'SE E=T' MAKES A COPY OF THE TRANSMIT LIST IN THE EXPECT LIST. THE EXPECT LIST NOW CONTAINS 3 MESSAGES (SAME AS TRANSMIT). THE MESSAGES WILL HAVE ALL ONES FOR DATA AND BE 20 BYTES EACH IN LENGTH. THE RUN COMMAND IS THE SAME WITH THE ADDITION OF TWO SWITCHES '/CH/PAS=3'. THE 'CH' SWITCH TELLS THE PROGRAM TO CHECK THE RECEIVED DATA AGAINST THE 'EXPECTED LIST'. THE 'PAS=3' SWITCH TELLS THE PROGRAM TO RUN 3 PASSES BEFORE RETURNING TO THE DCLT> PROMPT.

WHAT YOU SHOULD SEE AFTER ENTERING THE RUN COMMAND IF THINGS ARE RUNNING CORRECTLY :

```
INI RXQ TXQ RXQ TXC TXQ RXQ TXC  
TXQ TXC CMP CMP CMP EOP RXQ TXQ  
RXQ TXC TXQ RXQ TXC TXQ TXC CMP  
CMP CMP EOP RXQ TXQ RXQ TXC TXQ  
RXQ TXC TXQ TXC CMP CMP CMP EOP  
MODE=ACTIVE/LOOP=INTERNAL/PASS=00000  
/STATUS/CHECK/NOECHO/NOMODEM  
DCLT> (A) ?
```

IF A CABLE TURNAROUND CONNECTOR IS AVAILABLE, PUT IT ON THE END OF THE CABLE JUST BEFORE THE MODEM AND RUN IN ACTIVE MODE WITH NO LOOP. THIS COMMAND IS VALID FOR POINT TO POINT STATIONS ONLY. POSSIBLE COMMAND SEQUENCE IS:

```
R M=A/CH/PAS=3
```

WHAT THIS SEQUENCE MEANS:

THIS SEQUENCE HAS THE '/LO=1' REMOVED. THIS INFORMS THE DEVICE TO ACT AS IF IT WAS RECEIVING FROM ANOTHER NODE.

WHAT YOU SHOULD SEE AFTER ENTERING THE RUN COMMAND IF THINGS ARE RUNNING CORRECTLY :

```
RXQ TXQ TXC RXQ TXQ TXC RXQ TXQ
TXC CMP CMP CMP EOP RXQ TXQ TXC
RXQ TXQ TXC RXQ TXQ TXC CMP CMP
CMP EOP RXQ TXQ TXC RXQ TXQ TXC
RXQ TXQ TXC CMP CMP CMP EOP
MODE=ACTIVE/PASS=00000
/STATUS/CHECK/NOECHO/NOMODEM
DCLT> (A) ?
```

7.2.2 TRANSMIT ON ONE NODE RECEIVE ON THE OTHER

NOW TRY TRANSMITTING FROM ONE END AND RECEIVING ON THE OTHER. MAYBE WITH NO DATA CHECKING AT FIRST TO ESTABLISH IF THE LINK IS WORKING. POSSIBLE COMMAND SEQUENCES ARE:

* NOTE * - THESE SEQUENCES ARE FOR POINT TO POINT MODE
***** IF YOU WISH TO RUN MULTIPPOINT ADD THE COMMAND
COMMAND LINES MARKED WITH AN *.

NODE A	NODE B
-----	-----
C E	C E
C T	C T
SE T=1ALT/S=250	
* T E=1	T E=1
R M=TR/PAS=3	R M=R/NOCH/PAS=3

WHAT THIS SEQUENCE MEANS:

THE 'C E' AND 'C T' INITIALIZE BOTH THE TRANSMIT AND EXPECT LISTS. THE 'SE T=1ALT/S=250' SETS THE TRANSMIT LIST ON NODE A TO BE 1 MESSAGE WITH A LENGTH OF 250 BYTES AND DATA OF ALTERNATING ONES AND ZEROS. THE 'T E=1' ESTABLISHES 1 TRIBUTRAY WITH AN ADDRESS OF 1. THIS IS ONLY FOR MULTIPOINT SITUATIONS. THE 'R M=TR/PAS=3' SETS THE RUN MODE OF NODE A TO BE TRANSMIT AND THE PASS COUNT IS SET TO 3. THE 'R M=R/NOCH/PAS=3' SETS THE RUN MODE OF NODE B TO BE RECEIVE, NO DATA CHECKING IS TO BE DONE, AND THE PASS COUNT IS SET TO THREE.

WHAT YOU SHOULD SEE AFTER ENTERING THE RUN COMMAND IF THINGS ARE RUNNING CORRECTLY :

FOR NODE A:

```
INI TXQ TXC EOP TXQ TXC EOP TXQ
TXC EOP
MODE=TRANSMIT/PASS=00000
/STATUS/NOCHECK/NOECHO/NOMODEM
DCLT> (A) ?
```

FOR NODE B:
INI RXQ EOP RXQ EOP RXQ EOP
MODE=RECEIVE/PASS=00000
/STATUS/NOCHECK/NOECHO/NOMODEM
DCLT> (A) ?

NOW TRY DOING DATA CHECKING ON THE MESSAGE(S) BEING TRANSMITTED. POSSIBLE COMMAND SEQUENCES ARE:

R M=TR/PAS=3 SE E=1ALT/S=250
R M=R/CH/PAS=3

WHAT THIS SEQUENCE MEANS:

THE "SE E=1ALT/S=250" LINE MUST BE ADDED HERE TO SET UP THE "EXPECT" LIST ON THE RECEIVE NODE SO IT WILL KNOW WHAT TO COMPARE AGAINST. THE CHANGE IN THE RUN COMMAND IS FROM "NOCH" TO "CH" THE "CH" ENABLES DATA CHECKING

WHAT YOU SHOULD SEE AFTER ENTERING THE RUN COMMAND IF THINGS ARE RUNNING CORRECTLY :

NODE A: IS THE SAME AS ABOVE.

NODE B:
INI RXQ CMP EOP RXQ CMP EOP RXQ
CMP EOP
MODE=RECEIVE/PASS=00000
/STATUS/NOCHECK/NOECHO/NOMODEM
DCLT> (A) ?

NOW RUN THRU THE SEQUENCE AGAIN WITH NODE A RECEIVING AND NODE B TRANSMITTING TO CHECK OUT THE OPPOSITE DIRECTION OF DATA FLOW.

7.2.3

ONE NODE ACTIVE THE OTHER NODE PASSIVE

NOW TRY RUNNING ONE NODE IN ACTIVE MODE WHILE THE OTHER END RUNS IN PASSIVE. DATA CHECKING SHOULD BE TURNED OFF IF THE MESSAGE LISTS ARE NOT THE SAME. POSSIBLE COMMAND SEQUENCES ARE:

* NOTE * - THESE SEQUENCES ARE FOR POINT TO POINT MODE
***** IF YOU WISH TO RUN MULTIPPOINT ADD THE COMMAND
COMMAND LINES MARKED WITH AN *.

NODE A	NODE B
-----	-----
C E	C E
C T	C T
SE T=CCITT/S=10/C=2	SE T=1ALT/S=20/C=2

* T E=1 T E=1
R M=ACT/NOCH/PAS=3 R M=P/NOCH/PAS=3

WHAT THIS SEQUENCE MEANS:

THE EXECUTION OF THIS SEQUENCE CAUSES THE FOLLOWING THINGS TO HAPPEN ON NODE A. THE TRANSMIT AND EXPECT LISTS ARE INITIALIZED THEN THE TRANSMIT LIST IS SET TO 3 MESSAGES OF 10 BYTES EACH. THE DATA USED IN THE TRANSMIT MESSAGES IS THE CCITT PATTERN. THEN A IF THIS IS A MULTIPOINT NETWORK A TRIB IS ESTABLISHED (TRIB ADDR. 1) THEN NODE A IS RUN IN ACTIVE MODE WITH DATA CHECKING DISABLED AND THE PASS COUNT SET TO THREE. NOTE STATUS WOULD STILL BE PRINTED IF THE PREVIOUS SEQUENCES HAD BEEN RUN, IF YOU ARE RUNNING FROM LOAD TIME YOU WOULD HAVE TO ADD A '/STA TO THE RUN COMMAND LINE. NODE B: THE TRANSMIT AND EXPECT LISTS ARE INTIALIZED THEN THE TRANSMIT LIST IS SET TO 3 MESSAGES OF 20 BYTES EACH. THE DATA FOR EACH MESSAGE IS ALTERNATING 1'S AND 0'S. IF MULITPOINT ESTABLISH 1 TRIB. THEN RUN IN PASSIVE MODE WITH DATA CHECKING DISABLED AND THE PASS COUNT SET TO 3.

WHAT YOU SHOULD SEE AFTER ENTERING THE RUN COMMAND IF THINGS ARE RUNNING CORRECTLY :

FOR NODE A:

```
INI RXQ TXQ TXC TXQ RXQ TXC TXQ
RXQ TXC EOP RXQ TXQ RXC TXC TXQ
RXQ TXC TXQ RXQ TXC EOP RXQ TXQ
RXQ TXC TXQ RXQ TXC TXQ RXQ TXC
EOP
MODE=ACTIVE/PASS=00000
/STATUS/NOCHECK/NOECHO/NOMODEM
DCLT> (A) ?
```

FOR NODE B:

```
INI RXQ TXQ TXC RXQ TXQ TXC RXQ
TXQ TXC EOP
MODE=PASSIVE/PASS=00000
/STATUS/NOCHECK/NOECHO/NOMODEM
DCLT> (A) ?
```

NOW USE DATA CHECKING WITH THE 'EXPECT MESSAGE LISTS' SET UP APPROPRIATELY. ANOTHER VARIATION IS TO HAVE LARGE SIZE MESSAGES ON ONE SIDE WITH SMALL MESSAGES ON THE OTHER.

THEN REVERSE THE SETUP SO THAT THE NODE RUNNING IN ACTIVE IS RUNNING IN PASSIVE AND VICE VERSA.

7.2.4 BOTH NODES ACTIVE

NOW BOTH NODES CAN BE RUN IN ACTIVE WITH DATA CHECKING ON.
STATUS PRINTING COULD BE TURNED OFF IF YOU'RE NOT INTERESTED
IN THEM.

NOTE - THIS IS FOR POINT TO POINT ONLY

NODE A	NODE B
-----	-----
C E	C E
C T	C T
SE T=OALT/S=10	SE E=OALT/S=10
SE T=CCITT/S=20	SE E=CCITT/S=20
SE T=ALPHA/S=30	SE E=ALPHA/S=30
SE E=ZERO/S=11	SE T=ZERO/S=11
SE E=ONES/S=21	SE T=ONES/S=21
SE E=ITEP/S=31	SE T=ITEP/S=31
R M=A/CH/NOST/PAS=3	R M=A/CH/NOST/PAS=3

WHAT THIS SEQUENCE MEANS:

NODE A SETS UP IS TRANSMIT LIST TO BE
3 MESSAGES. MESSAGE 1 IS 10 BYTES LONG AND
CONTAINS DATA OF ALTERNATING 0'S AND 1'S
MESSAGE 2 IS 20 BYTES LONG AND CONTAINS
DATA OF THE CCITT PATTERN. MESSAGE THREE
IS 30 BYTES LONG AND CONTAINS ALPHANUMERICS
FOR DATA. THE EXPECT LIST ALSO CONTAINS
3 MESSAGES. MESSAGE 1 IS 11 BYTES LONG AND
CONTAINS 0'S FOR DATA. MESSAGE TWO IS 21
BYTES LONG AND CONTAINS 1'S FOR DATA. MESSAGE
3 IS 31 BYTES LONG AND CONTAINS THE ITEP DATA.
NODE B HAS THE SAME MESSAGES EXCEPT THAT THE
TRANSMIT MESSAGE LIST IS THE EXPECT MESSAGE LIST
AND VICE VERSA.
BOTH NODES ARE RUN IN THE ACTIVE MODE WITH NO
DATA CHECKING AND PASS COUNT EQUAL TO THREE.

WHAT YOU SHOULD SEE AFTER ENTERING THE RUN COMMAND
IF THINGS ARE RUNNING CORRECTLY :

ON BOTH NODES A AND B:

MODE=ACTIVE/PASS=00000
/NOSTATUS/CHECK/NOECHO/NOMODEM

DCLT> (A) ?

A VARIATION THAT CAN BE USED IS FOR ONE END TO SEND A LOT OF
SMALL MESSAGES AND THE OTHER TO SEND A FEW LARGE MESSAGES.
THE 'END-OF-PASS' POINT WILL BE OUT OF SYNC BUT THIS IS NOT
A PROBLEM.

7.2.5 TALK AND LISTEN MODES FOR COMMUNICATING

TALK AND LISTEN MODES ARE USEFUL IF THE OPERATORS WISH TO COMMUNICATE
WITH EACH OTHER. JUST SETUP A TIME THAT EACH WILL GO TO THEIR MODE,
TALK OR LISTEN, AND SEND MESSAGES OVER THE LINK. POSSIBLE COMMAND
SEQUENCES ARE. WHEN USING TALK AND LISTEN MODES ON MULTIPOINT LINKS

REMEMBER THAT YOU CAN ONLY USE THESE MODES FROM THE CONTROL STATION
TO THE FIRST TRIBUTARY IN THE TRIB LIST.

R M=LIS/NOST
LIS>

R M=TA/NOST
TLK>

7.3 EXAMPLES OF COMMANDS

THIS SECTION WILL SHOW A SAMPLING OF COMMANDS AND EXACTLY WHAT TO EXPECT FROM THEM.

7.3.1 EXAMPLES OF MESSAGES COMMANDS

THE CLEAR COMMANDS .

C E
C T

THIS WILL INITIALIZE THE TRANSMIT AND EXPECT LIST TO 1 MESSAGE OF 58 BYTES. THE DATA OF THE MESSAGE WILL BE THE ITEP MESSAGE.

IF THESE COMMANDS ARE FOLLOWED BY A SHOW COMMAND

SH E

SUCH AS THE SHOW EXPECT LIST. WHAT YOU WOULD SEE IS

MSG: TYPE=ITEP/SIZE=58
MODE=ACTIVE/PASS=00001
/NOSTATUS/CHECK/NOECHO/NOMODEM

DCLT> (A) ?

NOW IF YOU DID A SET EXPECT LIST COMMAND SUCH AS:

SE E=A/S=35/C=3

AND FOLLOWED IT WITH A SHOW EXPECT LIST COMMAND

SH E

WHAT YOU WOULD SEE IS

MSG: TYPE=ALPHA/SIZE=35
MSG: TYPE=ALPHA/SIZE=35
MSG: TYPE=ALPHA/SIZE=35
MSG: TYPE=ALPHA/SIZE=35
MODE=ACTIVE/PASS=00001
/NOSTATUS/CHECK/NOECHO/NOMODEM

DCLT> (A) ?

7.3.1 EXAMPLES TRIBUTARY COMMANDS

WHEN YOU FIRST GET TO THE DCLT> COMMAND LEVEL IN MULTIPOINT MODE AND YOU EXECUTE A TRIB SHOW COMMAND:

T S

WHAT YOU WOULD SEE IS

TRIB ADDRESS LIST IS EMPTY

THEN YOU COULD TO A TRIB ESTABLISH COMMAND

T E=1,2,3,4

THIS WOULD ESTABLISH TRIB ADDRS 1 2 3 AND 4

IF YOU FOLLOWED THIS WITH A TRIB SHOW COMMAND YOU WOULD SEE

TRIB ADDRESS LIST:

1, 2, 3, 4

IF YOU THEN DID A TRIB KILL COMMAND

T K=3

FOLLOWED BY A TRIB SHOW.

T S

WHAT YOU WOULD SEE IS

TRIB ADDRESS LIST:

1, 2, 4
IF YOU FOLLOWED THIS WITH A TRIB KILL ALL COMMAND
T K=A
AND ANOTHER TRIB SHOW
T S
WHAT YOU WOULD SEE IS
TRIB ADDRESS LIST IS EMPTY
IS YOU DID A TRIB ESTABLISH WITH A /W SWITCH
T E=1/W,2/W
WHAT YOU WOULD SEE IS SHOWN BELOW WHEN YOU GET TO THE ?
TYPE EITHER THE NEW PARAMATER OR CARRIAGE RETURN FOR
DEFALUT.

PARAMETERS FOR TRIB 001
000000 PRESET VALUE FOR TX DELAY TIMER
NEW POLL PARAMETERS (WORD)= (0) 0 ?

377 Q VAL FOR ACT
000 R VAL FOR ACT
NEW POLL PARAMETERS (BYTE LOW) = (0) 377 ?

NEW POLL PARAMETERS (BYTE HI) = (0) 0 ?
000 Q VAL FOR INACT
100 R VAL FOR INACT
NEW POLL PARAMETERS (BYTE LOW) = (0) 0 ?

NEW POLL PARAMETERS (BYTE HI) = (0) 100 ?
000 Q VAL FOR UNRSP
020 R VAL FOR UNRSP
NEW POLL PARAMETERS (BYTE LOW) = (0) 0 ?

NEW POLL PARAMETERS (BYTE HI) = (0) 20 ?
010 NDM TO INACT
002 # T-0 TO UNRSP
NEW POLL PARAMETERS (BYTE LOW) = (0) 10 ?

NEW POLL PARAMETERS (BYTE HI) = (0) 2 ?
010 #T-0 TO DEAD
004 MAX MSG COUNT
NEW POLL PARAMETERS (BYTE LOW) = (0) 10 ?

NEW POLL PARAMETERS (BYTE HI) = (0) 4 ?

005670 SELECTION INTERVAL TIMING COUNT
NEW POLL PARAMETERS (WORD)= (0) 5670 ?

013650 BABBLING TRIB TIMING COUNT
NEW POLL PARAMETERS (WORD)= (0) 13650 ?
PARAMETERS FOR TRIB 002
000000 PRESET VALUE FOR TX DELAY TIMER

·
·
·
·
THE SAME AS FOR TRIB 1

:
:
:
013650 BABBLING TRIB TIMING COUNT
NEW POLL PARAMETERS (WORD)= (0) 13650 ?

GLOBAL POLL PARAMETERS
0000015 NUM SYNC
NEW POLL PARAMETERS (WORD)= (0) 15 ?

013650 CARRIER WAIT TIMER COUNTER
NEW POLL PARAMETERS (WORD)= (0) 13650 ?

000062 DELTA T
NEW POLL PARAMETERS (WORD)= (0) 13650 ?

000000 DEAD T
NEW POLL PARAMETERS (WORD)= (0) 13650 ?

000000 POLL DELAY
NEW POLL PARAMETERS (WORD)= (0) 13650 ?

DCLT> (A) ?

7.3.1 EXAMPLES STATISTICAL COMMANDS

IF YOU TYPE A HELP COMMAND
HELP

WHAT YOU WILL SEE IS

DCLT CMDS:

CLEAR OR SHOW EXPECTLIST OR TRANSMITLIST
PRINT OR EXIT
DUMP START-END/B
TRIB SHOW, TRIB ESTABLISH=N/W,N(D)..OR TRIB KILL=N,ALL
WHERE W=INDICATES WRITE POLL PARAMS
SET EXPECTMSG OR TRANSMITMSG=TYPE/SIZE=N OR /COPY=N
SET EXPECT=TRANSMIT
TYPE=ONES,ZEROES,1ALT,0ALT,ITEP,CCITT,ALPHA
OR 'OPR SPCD=A-Z,SP,TAB,0-9 IN QUOTES'
RUN MODE=MTYP/LOOP=LTP/CHECK,STATUS,ECHO,MODEM,PASS=N
MTYP=TRAN,REC,ACT,PAS,TAL,LIS,DOWN
LTP=INT,CAB,LOC,REM/

DCLT> (A) ?

THE SAME WILL HAPPEN IF YOU USE THE ?

THE DUMP COMMAND WORKS LIKE THIS

DUM 41260-41300

THIS WILL DUMP THE DATA FROM ADDRESSES 41260 TO
41300 IN THE FOLLOWING MANNER

41260 104423 000167 177772 021122 012112 006312 006312 006312
41300 006312

IF YOU HAD USED THE /B SWITCH
DUM 41260-41300/B
WHAT YOU WOULD SEE IS
41260 023 211 167 000 372 377 122 024
41270 112 024 312 014 312 014 312 014
41300 312

7.3.1 EXAMPLES RUN COMMANDS

YOU CAN FIND SEVERAL EXPAMLES OF THE RUN COMMAND IN THE TROUBLE SHOOTING HINTS SECTION BUT HERE ARE SOME OTHERS.

IF YOU WERE TO EXECUTE THE RUN COMMAND
R M=TR/NOST/CH/PAS=4
WHAT WOULD HAPPEN IS AFTER 4 PASSES THE PROGRAM WOULD RETURN TO THE DCLT PROMT. AND PRINT.
MODE=TRANSMIT/PASS=0000
/NOSTATUS/CHECK/NOECHO/NOMODEM

DCLT> (A) ?
IF YOU WERE TO EXECUTE THE RUN COMMAND
R M=A/LO=I/ST/CH/PAS=4
WHAT YOU WOULD SEE (IF USING DEFUALT TRANSMIT AND EXPECT MESSAGES) IS
INI RXQ TXQ TXC CMP EOP RXQ TXQ
TXC CMP EOP RXQ TXQ TXC CMP EOP
MODE=ACTIVE/LOOP=INTERNAL/PASS=0000
/STATUS/CHECK/NOECHO/NOMODEM

DCLT> (A) ?

IF YOU USE THE EXIT COMMAND
EXIT
WHAT YOU WOULD SEE IS
CZCLM EOP
0 CUMLATIVE ERRORS

DR>

7.3.1 EXAMPLES PRINT COMMANDS

THE PRINT COMMAND CAN BE USED FROM THE SUPERVISOR (DR>) LEVEL OR THE DCLT (DCLT>) LEVEL. ONCE YOU ARE AT THAT LEVEL YOU WILL KNOW IT BY THE PROMPT 'RPT>'. AFTER TYPING PRI FOR EITHER THE THE DLCT> OR DR> PROMPTS

TYPE 'H' OR '?' FOR HELP!
RPT> (A) ?

HERE ARE SOME EXAMPLES OF RPT> LEVEL COMMANDS

THE HELP OR ? COMMAND
HELP

OR

?
PRODUCES THE FOLLOWING:

THIS WILL PRODUCE:

000000 TRIB STATUS FLAGS
000 NAK REASON
000 TRIB ADDR
000000 POLL STATUS FLAGS
000 POLL RATE
000 POLL PRIORITY
000 NA
000 MAX MSG COUNTER
000 COMM POLL QUOTA
000 RX THRESH ERRS
000 TX THRESH. ERRS
000 SELECT THRESH. ERRS
000000 DATA MSGS. TX'MITTD
000000 DATA MSGS. RX'CVD
000000 SELECTION INTERVALS
000 DATA ERRORS OUT
HBCC 0 BCC 0 REP 0
000 DATA ERRORS IN
HBCC 0 BCC 0 REP 0
000 LOCAL BUFFER ERRS
TU 0 TS 0
000 REMOTE BUFFER ERRS
TU 0 TS 0
000 SELECTION T-0
NRTS 0 IRTS 0
000 LOCAL REPLY T-0
000 REMOTE REPLY T-0
000 HIGHEST MSG # TX'D
000 HIGHEST MSG # ACK'D
000 NEXT MSG # TO TX
000 TPTR ADDR OF LKNBK
000 LAST MSG # TX'D
000 XPTR ADDR OF LNKBK
000 CTL X REPLY T-0
000 STRT OF TX BUFF Q
000 END OF TX BUFF Q
000 HIGHEST MSG # RX'D
000 STRT OF RX BUFF Q
000 END OF RX BUFF Q
000000 TX DELAY TIMER
000 NO DATA MSG COUNTER
000 T-0 COUNTER
000000 PRESET VALUE FOR TX DELAY TIMER
000 Q VAL FOR ACT
000 R VAL FOR ACT
000 Q VAL FOR INACT
000 R VAL FOR INACT
000 Q VAL FOR UNRSP
000 R VAL FOR UNRSP
000 NDM TO INACT
000 # T-0 TO UNRSP
000 # T-0 TO DEAD
000 MAX MSG COUNT
000000 SELECTION INTERVAL TIMING COUNT

000000 BABBLING TRIB TIMING COUNT

TO GET A SPECIFIC OFFSET LOCATION FROM THE
TSS USE THE COMMAND

T 1/0=4

THIS IS FOR THE VALUES AT OFFSET 4 BUT YOU COULD
USE ANY VALUE FROM 0 TO 36 OCTAL
THIS WILL PRODUCE:

000 MAX MSG COUNTER
000 COMM POLL QUOTA

TO GET THE GLOBAL ERROR COUNTERS USE
THE COMMAND

G

TO PRODUCE

TO GET THE FULL GSS USE THE COMMAND

G/F

TO PRODUCE:

000 POLPTR
000 RCVPTR
000 XMTPTR
000 TSP
000 NASP
000 BUFPTR
000 S-OF
000 E-OF
000 S-OQ
000 E-OQ
000 S-OC
000 E-OC
000 TIMER STATUS
000 S-R TIMER [L]
000 S-R
000 B-CW TIME [H]
000 RPM CNTR
000000 AACTIM
000 MODEM
000 MODE
000 ALT SW
000 XMTQRT
000000 RTNADD
000 REMOTE STA ERRS
OVRN 0 MHFE 0 SEL 0 STR 0
000 LOCAL STA ERRS
OVRN 0 MHFE 0 UNDR0 OVR 0
000 GBL HDR BCC
000 MAINT DATA BCC ERR
000 TX HDR 1
000 TX HDR 2
000 TX HDR 3
000 TX HDR 4
000 TX HDR 5
000 TX HDR 6

```
000 RX HDR 1
000 RX HDR 2
000 RX HDR 3
000 RX HDR 4
000 RX HDR 5
000 RX HDR 6
000000 R TIMER
000000 D TIMER
000000 POLL DELAY TIMER
000 POLL UPDATE PTR
000 DEAD SCAN
000 CARRIER LOSS TIM
000 USART HANG CTR
000000 NUM SYNC
000000 CARRIER WAIT TIMER COUNTER
000000 DELTA T
000000 DEAD T
000000 POLL DELAY
```

* NOTE * - DATA DISPLAYED HERE IS ZEROES ACTUAL DATA WILL VARY

TO GET AN OFFSET VALUE USE THE COMMAND

G/O=4

TO PRODUCE

```
000 E-OF
000 S-OQ
```

THE EXIT COMMAND WORKS LIKE THIS. IF YOU
ENTERED THE REPORT LEVEL FROM THE SUPERVISOR
(DR>) THEN TYPING

EXIT

WILL RETURN YOU TO THE SUPERVISOR

DR>

IF YOU ENTERED REPORT FROM THE DCLT LEVEL
THEN TYPING

EXIT

WILL RETURN YOU TO THE DCLT LEVEL

DCLT>

7.4 THINGS TO WATCH OUT FOR

IF YOU ARE RUNNING DCLT ON SYSTEMS THAT HAVE CONSOLES
WITH DIFFERENT SPEEDS YOU WILL BE UNABLE TO USE THE
PRINT STATUS FEATURE IN CERTAIN MODES. THE RULE IS
IF IT DOESNT WORK WITH STATUS PRINTING RUN THE MODE
WITH NOSTATUS.

IF YOU ARE USING PASSVIE MODE WITH THE ECHO SWITCH
THEN YOU WILL PROBABLY HAVE TO RE ENTER THE TRANSMIT
LIST ON THE SIDE WITH THE ECHO SWITCH. THE REASON IS
THAT THE TRANSMIT LIST GETS OVER WRITTEN WITH THE
RECEIVE LIST WHEN USING THE ECHO SWITCH

IF YOU ARE IN MULTIPOINT MODE AND YOU ARE USING THE
DATA CHECK FEATURE ALL TRIBUTARYS MUST USE THE SAME
TRANSMIT LIST.

&

2365
 2366
 2367
 2368 002000
 2369
 2370
 2371
 2372
 2373
 2374
 2375
 2376
 2377
 2378
 2379 002000
 2380
 2381
 2382
 2383
 2384
 2385 002000
 2386 002000
 2387 002000 103
 2388 002001 132
 2389 002002 103
 2390 002003 114
 2391 002004 115
 2392 002005 000
 2393 002006 000
 2394 002007 000
 2395 002010
 2396 002010 102
 2397 002011
 2398 002011 060
 2399 002012
 2400 002012 000000
 2401 002014
 2402 002014 003410
 2403 002016
 2404 002016 066562
 2405 002020
 2406 002020 000000
 2407 002022
 2408 002022 002130
 2409 002024
 2410 002024 000000
 2411 002026
 2412 002026 067222
 2413 002030
 2414 002030 000000
 2415 002032
 2416 002032 000000
 2417 002034
 2418 002034 000000
 2419 002036
 2420 002036 000000

.SBTTL PROGRAM HEADER
 BGNMOD

:++
 : THE PROGRAM HEADER IS THE INTERFACE BETWEEN
 : THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
 :--

POINTER BGNRPT,BGNAU,BGNDU

HEADER CZCLM,B,0,1800.,0,#PRI07

LSNAME::
 .ASCII /C/
 .ASCII /Z/
 .ASCII /C/
 .ASCII /L/
 .ASCII /M/
 .BYTE 0
 .BYTE 0
 .BYTE 0
 LSREV::
 .ASCII /B/
 LSDEPO::
 .ASCII /O/
 LSUNIT::
 .WORD 0
 LSTIML::
 .WORD 1800.
 LSHPCP::
 .WORD LSHARD
 LSSPCP::
 .WORD 0
 LSHPTP::
 .WORD LSHW
 LSSPTP::
 .WORD 0
 LSLADP::
 .WORD LSLAST
 L\$STA::
 .WORD 0
 L\$CO::
 .WORD 0
 L\$DTYP::
 .WORD 0
 L\$APT::
 .WORD 0

2421 002040
 2422 002040 002124
 2423 002042
 2424 002042 000340
 2425 002044
 2426 002044 000000
 2427 002046
 2428 002046 000000
 2429 002050
 2430 002050 003
 2431 002051 003
 2432 002052
 2433 002052 000000
 2434 002054 000000
 2435 002056
 2436 002056 000000
 2437 002060
 2438 002060 023264
 2439 002062
 2440 002062 050734
 2441 002064
 2442 002064 000000
 2443 002066
 2444 002066 000000
 2445 002070
 2446 002070 052030
 2447 002072
 2448 002072 052022
 2449 002074
 2450 002074 000000
 2451 002076
 2452 002076 023302
 2453 002100
 2454 002100 104035
 2455 002102
 2456 002102 000000
 2457 002104
 2458 002104 050750
 2459 002106
 2460 002106 051764
 2461 002110
 2462 002110 051762
 2463 002112
 2464 002112 050742
 2465 002114
 2466 002114 000000
 2467 002116
 2468 002116 000000
 2469 002120
 2470 002120 000000
 2471

LSDTP:: .WORD L\$DISPATCH
 L\$PRIO:: .WORD #PRI07
 L\$ENVI:: .WORD 0
 L\$EXP1:: .WORD 0
 L\$MREV:: .WORD 0
 .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 L\$AU
 L\$DUT:: .WORD L\$DU
 L\$LUN:: .WORD 0
 L\$DESP:: .WORD L\$DESC
 L\$LOAD:: EMT ESLOAD
 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

2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483

.SBTTL DISPATCH TABLE

+++
: THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
: IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.

DISPATCH 1

002122
002122 000001
002124
002124 052036

.WORD 1
LSDISPATCH::
.WORD T1

2484
 2485
 2486
 2487
 2488
 2489
 2490
 2491
 2492
 2493
 2494
 2495
 2496
 2497
 2498
 2499
 2500
 2501
 2502
 2503
 2504
 2505
 2506
 2507
 2508
 2509
 2510
 2511
 2512
 2513
 2514
 2515
 2516
 2517
 2518
 2519
 2520
 2521
 2522
 2523
 2524
 2525

002126
 002126 000010
 002130
 002130
 002130 000001
 002132 160170
 002134 000300
 002136 000240
 002140 000000
 002142 000000
 002144 000004
 002146 000000
 002150
 002150

```

.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.
:--

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

:INDEPENDENT SECTION
: THE NUMBERS IN BRACKETS ARE THE OFFSET VALUES USED IN THE PARAMETER
: CODING SECTION.

      .WORD  1                ;[0] FULL OR HALF DUPLEX FLAG (BIT0=1 IF FULL)

:DEVICE DEPENDENT SECTION
: ADDING OR REMOVING WORDS FROM THIS TABLE EFFECTS THE 'GET' CALLS IN
: THE HARDWARE PARAMTER CODING SECTION BY CHANGING 'OFFSETS'

      .WORD  160170          ;[2] CSR ADDRESS
      .WORD  300             ;[4] INTERRUPT VECTOR
      .WORD  240             ;[6] INTERRUPT PRIORITY (5)
      .WORD  0               ;[10] DEVICE PARAMETERS WORD
                                ; BIT0=(1=MULTI 0=POINT TO POINT)
                                ; BIT1=(1=CONTROL 0=TRIB)
      .WORD  0               ;[12] DEVICE OPTION TYPE
                                ; 0=DMP,1=DMV
      .WORD  4               ;[14]SPARE
      .WORD  0               ;[16]SPARE

      ENDPHW
                                L10000:
  
```

2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581

002150

.SBTTL GLOBAL EQUATES SECTION

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

EQUALS

:
: BIT DIFINITIONS

:
BIT15== 100000
BIT14== 40000
BIT13== 20000
BIT12== 10000
BIT11== 4000
BIT10== 2000
BIT09== 1000
BIT08== 400
BIT07== 200
BIT06== 100
BIT05== 40
BIT04== 20
BIT03== 10
BIT02== 4
BIT01== 2
BIT00== 1

:
BIT9== BIT09
BIT8== BIT08
BIT7== BIT07
BIT6== BIT06
BIT5== BIT05
BIT4== BIT04
BIT3== BIT03
BIT2== BIT02
BIT1== BIT01
BIT0== BIT00

:
: EVENT FLAG DEFINITIONS
: EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION

:
EF.START== 32. ; START COMMAND WAS ISSUED
EF.RESTART== 31. ; RESTART COMMAND WAS ISSUED
EF.CONTINUE== 30. ; CONTINUE COMMAND WAS ISSUED
EF.NEW== 29. ; A NEW PASS HAS BEEN STARTED
EF.PWR== 28. ; A POWER-FAIL/POWER-UP OCCURRED

:


```
2582  
2583  
2584  
2585 000340  
2586 000300  
2587 000240  
2588 000200  
2589 000140  
2590 000100  
2591 000040  
2592 000000  
2593  
2594  
2595  
2596 000004  
2597 000010  
2598 000020  
2599 000040  
2600 000100  
2601 000200  
2602 000400  
2603 001000  
2604 002000  
2605 004000  
2606 010000  
2607 020000  
2608 040000  
2609 100000  
2610
```

```
;  
; PRIORITY LEVEL DEFINITIONS  
;  
PRI07== 340  
PRI06== 300  
PRI05== 240  
PRI04== 200  
PRI03== 140  
PRI02== 100  
PRI01== 40  
PRI00== 0  
;  
; OPERATOR FLAG BITS  
;  
EVL== 4  
LOT== 10  
ADR== 20  
IDU== 40  
ISR== 100  
UAM== 200  
BOE== 400  
PNT== 1000  
PRI== 2000  
IXE== 4000  
IBE== 10000  
IER== 20000  
LOE== 40000  
HOE== 100000
```

```

2611          ; INDEPENDENT EQUATES
2612
2613          001000          BUFLIM=512.          ;MAX BUFFER SIZE IN BYTES
2614
2615          000017          MSGLIM=15.          ; APPLIES TO TX,RX AND CMP BUFFS
2616
2617
2618
2619
2620
2621          004000          RBFLIM=2048.          ;MAX NUMBER OF BYTES FROM ALL TRIBS
2622
2623
2624          ;MODE OF OPERATION EQUATES          ;ALLOWS FOR 32 TRIBS TIMES DEFAULT
2625          000000          REC=0          ;ITEP MESSAGE SIZE (58 BYTES).
2626          000001          TRA=1          ;RECEIVE MODE
2627          000002          PAS=2          ;TRANSMIT MODE
2628          000003          ACT=3          ;PASSIVE MODE
2629          000004          DOW=4          ;ACTIVE MODE
2630          000005          TAL=5          ;DOWN-LINE-LOAD MODE
2631          000006          LIS=6          ;TALK MODE
2632
2633          ;MAINT LOOP TYPE EQUATES          ;LISTEN MODE
2634
2635          000000          NONE= 0          ;NO LOOP
2636          000001          TTL= 1          ;INTERNAL TTL
2637          000002          CABLE= 2          ;CABLE LOOP
2638          000003          MODLOC= 3          ;MODMEM LOCAL
2639          000004          MODREM= 4          ;MODEM REMOTE
2640          000005          MOP= 5          ;MOP
2641
2642
2643          ;CLOCK ENABLE VALUES TO BE LOADED IN CLK'S CSR
2644
2645          000100          LCLKEN= 100          ;L-CLOCK CSR VALUE TO ENABLE THE CLOCK
2646          000111          PCLKEN= 111          ;P-CLOCK CSR VALUE TO ENABLE THE CLOCK
2647          001600          PCLKCT= 1600          ;P-CLOCK COUNT SET REGISTER FOR COUNTER
2648
2649          ;PARAM WORD EQUATES
2650
2651          000001          STATB= BIT0          ;OPERATOR AWAKE ASKED FOR
2652          000002          DATCKB= BIT1          ;DATA CHECK BIT
2653          000004          ECHOB= BIT2          ;ECHO BIT
2654          000010          MOCHK= BIT3          ;MODEM CHECK/NO CHECK
2655          000020          CRCB= BIT4          ;CRC CALCULATE ASKED FOR
2656          000040          PROTOB= BIT5          ;PROTOCOL PROCESSING ASKED FOR
2657
2658          ;OPTION TYPE EQUATES
2659
2660
2661          000000          DMP= 0          ;DMP OPTION
2662          000001          DMV= 1          ;DMV
2663          000004          DMP6= 4          ;DMP 8206
2664
2665          000001          MTP= BIT0          ;MULTIPOINT IF 1 IF PTPT =0
2666          000002          TRBB= BIT1          ;TRIB BIT IF 0=TRIB IF 1=CONTROL
    
```

2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722

000000
000002
000004
000006
000010
000012
000014
000016
000020
000022
000024

000001
000002
000004
000010
000100
000200

000400
001000

000000
000001
000002
000003
000004
000005
000006
000007
000010
000011
000012

000000
000001
000002
000003
000004
000005
000006
000007
000010
000011
000012
000013
000014

:EVENT LOG MESSAGE TYPES (USED TO LOCATE EVENT DESCRIPTION IN EVENT TABLE
 : AND DISPATCHING TO SEPARATE SECTIONS OF THE EVENT REPORTING SECTION)

TXQ=	0	:TRANSMIT MESSAGE QUEUED.
TXC=	2	:TRANSMIT COMPLETE
RXQ=	4	:RECEIVE BUFFER QUEUED
RXC=	6	:RECEIVE COMPLETE
DER=	10	:DEVICE INFORMATION
DVI=	12	:DEVICE ABOUT TO INIT
DCK=	14	:DATA COMPARISON RESULTS
MSC=	16	:MODEM STATUS CHANGE
DLE=	20	:DATA COMPARISON LENGTH ERROR
DDE=	22	:DATA COMPARISON DATA ERROR
EOP=	24	:END OF PASS

:EQUATES FOR FLAG WORD

ININT=	BIT0	:INPUT INT. REC.
OTINT=	BIT1	:OUTPUT INT REC
QRX=	BIT2	:RX QUED /COMPL
QTX=	BIT3	:TX QUED/COMPL
ERX=	BIT6	:EXPECT TO GET A RX COMPLED
ETX=	BIT7	:EXPECT TO GET A TX COMPLETED

RUNST=	BIT8	:INDICATES TRIB COULD GIVE RUN STATE INTERRUPT
DLLGA=	BIT9	:INDICATES GO AHEAD FOR DLL.

: SPECIAL CLI CODES FOR 'CHAR' ARGUMENT IN CLI CALLS
 : (COMMAND LINE INTERPRETER DEFINITIONS)

CLIERR=	0
CLIXI=	1
CLIBR=	2
CLIBIF=	3
CLISPA=	4
CLINUM=	5
CLIALP=	6
CLIALN=	7
CLIOCT=	8.
CLIDEC=	9.
CLISTR=	10.

: DEFS FOR COMMAND LINE INTERPRETATION ACTION VALUES

NULL=	0
CLEAR=	1
SHOW=	2
CHECK=	3
RUN=	4
HLP=	5
CSHEXP=	6
CSHTRN=	7
SETEXP=	10
SETTRN=	11
SIZE=	12
QCOPY=	13
NUM=	14

2723	000015	OPRMSG=15
2724	000016	STATUS=16
2725	000017	ENDQO=17
2726	000020	CMSG0=20
2727	000021	CMSG1=21
2728	000022	CMSG2=22
2729	000023	CMSG3=23
2730	000024	CMSG4=24
2731	000025	CMSG5=25
2732	000026	CMSG6=26
2733	000027	ATVMOD=27
2734	000030	PASMOD=30
2735	000031	RECMOD=31
2736	000032	LISMOD=32
2737	000033	DLLMOD=33
2738	000034	TRAMOD=34
2739	000035	TALMOD=35
2740	000036	NO=36
2741	000037	ECHO=37
2742	000040	CRC=40
2743	000041	PROTO=41
2744	000042	PASC=42
2745	000043	MOP=43
2746	000044	TTLLOP=44
2747	000045	CBLLOP=45
2748	000046	LMDLOP=46
2749	000047	RMDLOP=47
2750	000050	NOTNUF=50
2751	000051	BADCHR=51
2752	000052	DMPS=52
2753	000053	DMPE=53
2754	000054	DMPQ=54
2755	000055	PRNT=55
2756	000056	MOSC=56
2757	000057	SLST=57
2758	000060	ETRB=60
2759	000061	KTRB=61
2760	000062	KALL=62
2761	000063	EKTB=63
2762	000064	CTPP=64
2763	000065	ETWS=65
2764	000066	EXIT=66
2765	000067	SETET=67
2766		
2767	000001	RPHLP=1
2768	000002	RPEXT=2
2769	000003	RPLOG=3
2770	000004	RPGSS=4
2771	000005	RPTSS=5
2772	000006	RPTSN=6
2773	000007	RPSWE=7
2774	000010	RPSWF=10
2775	000011	RPSWO=11
2776	000012	RNOTNF=12
2777		
2778		

;REV B EC

2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801

000004
000010
000001
000040
000200
040000
002000

000200
000200
040000
000020
000001
000020

```
: DEVICE DEPENDENT EQUATES
: MODEM SIGNAL BIT DEFINITIONS
: IF SIGNAL AVAILABLE IN DEVICE, EQUATE NAME TO BIT POSITION,
: ELSE EQUATE IT TO = 0
      CTS=    BIT2      ;CLEAR TO SEND (CIRCUIT CB)
      DSR=    BIT3      ;DATA SET READY (CIRCUIT CC)
      DCD=    BIT0      ;DATA CARRIER DETECT (CIRCUIT CF)
      RTS=    BIT5      ;REQUEST TO SEND (CIRCUIT CA)
      RI=     BIT7      ;RING INDICATOR (CIRCUIT CE)
      SQD=    BIT14     ;SIGNAL QUALITY DETECT (CIRCUIT CG)
      TM=    BIT10     ;MODEM IN TEST MODE (RS 449 ONLY CIRCUIT TM)

: DEVICE BIT DEFINITIONS
      RQI=    BIT7
      RDO=    BIT7
      MCLR=   BIT14
      RDI=    BIT4
      IEI=    BIT0
      IEO=    BIT4
```

```

2802 .SBTTL GLOBAL DATA SECTION
2803 .SBTTL DEFAULT MESSAGE DEFINITIONS AND TABLES
2804
2805 :++
2806 : THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
2807 : IN MORE THAN ONE TEST.
2808 :--
2809
2810 ;MESSAGE BYTE COUNT TABLE
2811
2812 DMSGCT:
2813 MSG0C: .WORD EMSG0-MSG0 ;BYTE COUNT OF MESSAGE #0
2814 MSG1C: .WORD EMSG1-MSG1 ;BYTE COUNT OF MESSAGE #1
2815 MSG2C: .WORD EMSG2-MSG2 ;BYTE COUNT OF MESSAGE #2
2816 MSG3C: .WORD EMSG3-MSG3 ;BYTE COUNT OF MESSAGE #3
2817 MSG4C: .WORD EMSG4-MSG4 ;BYTE COUNT OF MESSAGE #4
2818 MSG5C: .WORD EMSG5-MSG5 ;BYTE COUNT OF MESSAGE #5
2819 MSG6C: .WORD EMSG6-MSG6 ;BYTE COUNT OF MESSAGE #6
2820 OPCNT: .WORD 0 ;BYTE COUNT FOR OPERATOR SPEC'D MSG.
2821 MSG8C: .WORD EMSG8-MSG8 ;BYTE COUNT OF RECEIVE BUFFER FILL PATTERN
2822 DLLM1C: .WORD DLLM1E-DLLM1 ;DLL MSG 1 COUNT
2823 DLLM2C: .WORD DLLM2E-DLLM2 ;DLL MSG 2 COUNT
2824
2825
2826 ;MESSAGE ADDRESS TABLE
2827
2828 DMSGAD:
2829 MSG0 ;ADDRESS OF MESSAGE #0
2830 MSG1 ;ADDRESS OF MESSAGE #1
2831 MSG2 ;ADDRESS OF MESSAGE #2
2832 MSG3 ;ADDRESS OF MESSAGE #3
2833 MSG4 ;ADDRESS OF MESSAGE #4
2834 MSG5 ;ADDRESS OF MESSAGE #5
2835 MSG6 ;ADDRESS OF MESSAGE #6
2836 OPBUF ;ADDRESS OF OPERATOR SPEC'D MSG.
2837 MSG8 ;ADDRESS OF RECEIVE BUFFER FILL PATTERN
2838
2839 MSG0: .BYTE 000 ;MESSAGE OF ALL 0'S
2840 EMSG0:
2841 MSG1: .BYTE 377 ;MESSAGE OF ALL 1'S
2842 EMSG1:
2843 MSG2: .BYTE 252 ;MESSAGE OF ALTERNATING 1'S
2844 EMSG2:
2845 MSG3: .BYTE 125 ;MESSAGE OF ALTERNATING 0'S
2846 EMSG3:
2847 MSG4: ;'CCITT' 512-BIT (VS. 511 BITS) TEST PATTERN
2848 .WORD 177603,157427,031011,047321,163715,105221,143325,142304
2849 047321,163715,105221
2850 143325,142304
2851 .WORD 040041,014116,052606,172334,105025,123754,111337,111523
2852 172334,105025,123754
2853 111337,111523
2854 .WORD 030030,145064,137642,143531,063617,135075,066730,026575
2855 143531,063617,135075
2856 066730,026575
2857 .WORD 052012,053627,070071,151172,165044,031605,166632,016741
  
```

2858 002312 151172 165044 031605
 2859 002320 166632 016741
 2860 002324
 2861 002324
 2862
 2863 002324 077577 040444 052040
 2864 002332 042510 050440 044525
 2865 002340 045503 041040 047522
 2866 002346 047127 043040 054117
 2867 002354 045040 046525 042520
 2868 002362 020104 053117 051105
 2869 002370 052040 042510 046040
 2870 002376 055101 020131 047504
 2871 002404 027107
 2872 002406 005015 077401 077577
 2873 002414 000177
 2874 002416
 2875 002416
 2876 002416 022043 021041 023040
 2877 002424 024047 025051 026053
 2878 002432 027055 030460 031462
 2879 002440 032464 033466 034470
 2880 002446 035472 036474 037476
 2881 002454 040500 041502 042504
 2882 002462 043506 044510 045512
 2883 002470 046514 047516 050520
 2884 002476 051522 052524 053526
 2885 002504 054530 132
 2886 002507 057 056133 057135
 2887 002514 022537 000
 2888 002517
 2889 002520
 2890
 2891
 2892
 2893
 2894 002520 047045 040445
 2895 002524 000122
 2896 002646
 2897
 2898
 2899
 2900
 2901 002646 033
 2902 002647
 2903
 2904
 2905
 2906 002647 006
 2907 002650 000
 2908 002651 000
 2909 002652 000
 2910 002653 000
 2911 002654
 2912 002654 000
 2913 002655 000

EMSG4:
 MSG5: ;'INTERPROCESSOR TEST PROGRAM'S (ITEP)' MESSAGE
 ; #1. (DP1:)
 .ASCII <177><177>/SA THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG./

.ASCIZ <15><12><001><177><177><177><177>

EMSG5:
 MSG6: ;ALPHA-NUMERICS (OR FUTURE COMM TURNAROUND MSG)
 .ASCII /#\$!' &'()*+,-.0123456789:;<=>?@ABCDEFGHIJKLMNPOQRSTUVWXYZ/

.ASCIZ ?/[\] ^ _ % ?

EMSG6:
 .EVEN

 ; THESE THREE STORAGE AREAS MUST NOT BE SEPARATED !!!!

OPBFPT: .ASCII /%N%A/
 OPBUF: .BLKB 82. ;BUFFER FOR OPERATOR SPEC'D MESSAGES
 OPEND:

; THE ABOVE THREE LINES MUST BE KEPT TOGETHER
 ; *****

MSG8: .BYTE 33 ;RECEIVE BUFFER FILL PATTERN
 EMSG8:

; DOWN-LINE-LOAD MESSAGE DEFINITIONS

DLLM1: .BYTE 6
 PASS1: .BYTE 0
 PASS2: .BYTE 0
 PASS3: .BYTE 0
 PASS4: .BYTE 0
 DLLM1E:
 DLLM2: .BYTE 0 ;CODE
 .BYTE 0 ;LOAD NUMBER

DEFAULT MESSAGE DEFINITIONS AND TABLES

```

2914 002656 006 .BYTE 6 ;LOAD ADDRESS LSB
2915 002657 000 .BYTE 0
2916 002660 000 .BYTE 0
2917 002661 000 .BYTE 0 ;LOAD ADDRESS
2918
2919 ;: IMAGE DATA
2920 ;:
2921 002662 000240 NOP ;BYTE COUNT=240 (USED ONLY IN CATS VTC LOADER)
2922 002664 005037 000006 CLR @#6
2923 002670 012706 001000 MOV #1000,SP
2924 002674 012701 177560 MOV #177560,R1 ;SET UP TTY
2925 002700 010700 MOV PC,R0 ;MAKE ADDR.PIC
2926 002702 062700 000034 ADD #<MSG-.>,R0 ;ADDRESS MSG.
2927 002706 105761 000004 1$: TSTB 4(R1) ;TTY READY?
2928 002712 100375 BPL 1$ ;WAIT TIL YES
2929 002714 112061 000006 MOVB (R0)+,6(R1) ;TYFE A CHAR
2930 002720 001372 BNE 1$ ;KEEP GOING
2931 002722 012737 000026 000024 MOV #26,@#24 ;SET UP POWER FAIL
2932 002730 005037 000026 CLR @#26 ;MAKE SURE T BIT CLAER
2933 002734 000777 BR ;JUMP ON YOURSELF
2934 002736 006412 047502 052117 MSG: .ASCII <12><15>/BOOT MESSAGE WAS RECEIVED SUCCESSFULLY -END OF TEST!!/
2935 002744 046440 051505 040523
2936 002752 042507 053440 051501
2937 002760 051040 041505 044505
2938 002766 042526 020104 052523
2939 002774 041503 051505 043123
2940 003002 046125 054514 026440
2941 003010 047105 020104 043117
2942 003016 052040 051505 020524
2943 003024 041
2944 003025 012 027015 027056 .ASCIZ <12><15>/.....RELOAD PROGRAM...../
2945 003032 051056 046105 040517
2946 003040 020104 051120 043517
2947 003046 040522 027115 027056
2948 003054 000056
2949
2950 ;: PADDING TO OBTAIN 240 BYTES OF DATA
2951 003056 177603 157427 031011 .WORD 177603,157427,031011
2952 003064 047321 163715 105221 .WORD 047321,163715,105221
2953 003072 143325 142304 040041 .WORD 143325,142304,040041
2954 003100 014116 052606 172334 .WORD 014116,052606,172334
2955 003106 105025 123754 111337 .WORD 105025,123754,111337
2956 003114 111523 030030 145064 .WORD 111523,030030,145064
2957
2958 ;: CRC VALUE FOR ABOVE 240 BYTES OF DATA
2959 003122 152645 .WORD 152645 ;CRC
2960
2961 003124 006 .BYTE 6
2962 003125 000 .BYTE 0
2963 003126 000 .BYTE 0
2964 003127 000 .BYTE 0
2965 003130 DLLM2E:
2966
2967 .EVEN
2968

```



```

2969 ;COMMAND LINE BUFFER, DATA LOCATIONS AND MESSAGES FOR ACTION ROUTINES
2970
2971 003130 000122 CMDBUF: .BLKB 82. ;BUFFER FOR OPERATOR COMMANDS
2972 003252 000000 KEYWD1: .WORD 0 ;THIS LOC WILL =1 IF CLEAR TYPED, 2 FOR SHOW,
2973 ; A 4 IF RUN WAS TYPED, 5 IF HELP WAS TYPED
2974 003254 000000 QUALFG: .WORD 0 ;THIS LOC HOLDS QUALIFIER VALUE (SIZE OR COPY)
2975 003256 000000 QUALVL: .WORD 0
2976 003260 024121 HLPTAB: .WORD HLP1
2977 003262 024134 .WORD HLP2
2978 003264 024252 .WORD HLP2B
2979 003266 024342 .WORD HLP2C
2980 003270 024415 .WORD HLP3
2981 003272 024502 .WORD HLP3A ;REV B EC
2982 003274 024527 .WORD HLP4
2983 003276 024606 .WORD HLP4A
2984 003300 024664 .WORD HLP5
2985 003302 024754 .WORD HLP6
2986 003304
2987 003304 025375 HLPEND:
2988 003306 025417 RHLPTB: .WORD RHLP1
2989 003310 025434 .WORD RHLP2
2990 003312 025466 .WORD RHLP3
2991 003314 .WORD RHLP4
2992
2993
2994
2995 003314 025645 025654 025661 SHTYTB: .WORD SHTYP0,SHTYP1,SHTYP2,SHTYP3,SHTYP4,SHTYP5,SHTYP6,SHTYP7
2996 003322 025666 025673 025701
2997 003330 025706 025714
2998
2999 ; THE LIST OF BYTES BELOW ARE THE FIRST BYTES OF THE PREDEFINED MESSAGES
3000 ; USED TO 'SHOW' THE TRANSMIT AND COMPARE BUFFER CONTENTS.
3001
3002 003334 000 377 252 SHTAB: .BYTE 0,377,252,125,203,177,043
3003 003337 125 203 177
3004 003342 043
3005 003343
3006 003344
3007 .EVEN
3008 003344 026672
3009 003346 026702
3010 003350 026713
3011 003352 026723
3012 003354 026732
3013 003356 026747
3014 003360 026754
3015
3016 003362 026763
3017 003364 026773
3018 003366 027004
3019 003370 027012
3020 003372 027025
3021
3022 ;COMMAND LINE TRAVERSE LOCATIONS (USED BY 'P$TRV')
3023
3024 003374 000000 P$BUFA: .WORD 0 ;LOC. TO HOLD ADDR. OF CMD LINE BUFFER

```

3025	003376	000000	P\$TREE:	.WORD	0	:	LOC. TO HOLD ADDR. OF PARSING TREE
3026	003400	000000	P\$ACT:	.WORD	0	:	LOC. TO HOLD ADDR. OF ACTION ROUTINE
3027	003402	000000	P\$CNT:	.WORD	0	:	LOC. TO BE A COUNTER LOCATION
3028	003404	000000	P\$NUM:	.WORD	0	:	LOC. TO HOLD NUMERIC VALUE FROM PARSE
3029	003406	000000	P\$RADX:	.WORD	0	:	LOC. TO HOLD RADIX USED(LO) AND +/- (HI BYTE)
3030	003410	000	P\$NNUF:	.BYTE	0	:	RETURN =0 IF ENOUGH OF COMMAND FOUND
3031	003411	000	P\$GDBD:	.BYTE	0	:	RETURN CODE 0 IF NO ERROR FOUND
3032	003412	000	WRFLG:	.BYTE	0	:	WRITE FLAG
3033		003414		.EVEN			
3034	003414	000000	VALTRB:	.WORD	0	:	VALID TRIB FLAG..IF SET -1 THEN VALID REV B EC
3035							

```

3036          .SBTTL          MESSAGE BUFFERS AND POINTER TABLES
3037
3038 003416 001000          TXBUF:  .BLKB  BUFLIM  ;TRANSMITTER BUFFERS
3039 004416 001000          CMPBUF: .BLKB  BUFLIM  ;COMPARISON BUFFERS
3040 005416 004000          RXBUF:  .BLKB  RBFLIM  ;RECEIVER BUFFERS
3041
3042 011416 000036          PTRTAB: .BLKW  MSGLIM*2      ;TABLE FOR MESSAGE ADDRS. & BYTE COUNTS
3043 011512 000036          PTR13:  .BLKW  MSGLIM*2
3044 011606 000036          PTR23:  .BLKW  MSGLIM*2
3045 011702 001642          .BLKW  MSGLIM*2*31.  ;TABLE FOR MULTIPOINT POINTERS
3046
3047 015406          PTREND:          ; END OF MSG. PTR. TABLE
3048
3049 015406 000002          .BLKW  2          ;FILLER FOR OVERFLOW OF RX POINTER TABLE
3050 015412 000040          CPTRLS: .BLKW  32.          ;TABLE FOR MULITPOINT RX POINTERS
3051 015512 000040          CPTTLS: .BLKW  32.          ;TABLE FOR MULITPOINT TX POINTERS
3052 015612 000040          DVRCLS: .BLKB  32.          ;TABLE (BYTES) FOR REC COUNTS
3053 015652 000040          DVTCLS: .BLKB  32.          ;TABLE (BYTES) FOR TX COUNTS
3054 015712 000040          TRIBLS: .BLKB  32.          ;TABLE (BYTES) OF TRIB ADDRESSES
3055 015752 177777          .WORD  177777
3056 015754 000000          TRBTOT: .WORD  0          ;TOTAL NUMBER OF TRIBS IN LIST
3057 015756 000000          TRIBN:  .WORD  0          ;CURRENT TRIB NUMBER
3058 015760 000000          INDW:   .WORD  0          ;WORD INDEX
3059 015762 000000          INDEX:  .WORD  0          ;BYTE INDEX FOR TRIBS
3060 015764 000000          CTX:    .WORD  0          ;COUNTER FOR TX BUFFER COMPLETE INTERRUPTS
3061 015766 000000          CRX:    .WORD  0          ;COUNTER FOR RX BUFFER COMPLETE INTERRUPTS
3062 015770 000000          RSPTRS: .WORD  0          ;STACK POINTER FOR RX INTERPUTS ON STACK
3063 015772 000000          RSPTRE: .WORD  0          ;STACK POINTER FOR RX INTERPUTS OFF STACK
3064 015774 000000          TSPTR:  .WORD  0          ;STACK POINTER FOR TX INTERRUPTS
3065 015776 000006          TXSTAK: .BLKW  6.
3066 016012 000066          RXSTAK: .BLKW  54.          ;TX AND RX INT STACKS
3067 016166          RXSKEN:

```

```

3068                                     ;POLL DEFAULTS FOR TRIBS
3069
3070 016166 000000          POLDEF: .WORD 0           ;TX DELAY TIMER
3071 016170           377      .BYTE 377          ;Q FOR ACTIVE
3072 016171           000      .BYTE 0           ;R FOR ACTIVE
3073 016172           000      .BYTE 0           ;Q FOR INACTIVE
3074 016173           100      .BYTE 100          ;R FOR INACTIVE
3075 016174           000      .BYTE 0           ;Q FOR PDEAD
3076 016175           020      .BYTE 20          ;R FOR PDEAD
3077 016176           010      .BYTE 10          ;#NDM INACTIVE
3078 016177           002      .BYTE 2           ;#T/O TO PDEAD
3079 016200           010      .BYTE 10          ;#T/O TO DEAD
3080 016201           004      .BYTE 4           ;MAX MESSAGE COUNTER
3081                                     DMP
3082 016202 005670          DMVDF1: .WORD 5670        ;SELCT TIMER [3 SECS ]          DMV 454
3083 016204 013560          DMVDF2: .WORD 13560       ;INTERVAL TIMER [6 SECS ]      DMV 1130
3084
3085                                     ;GLOBAL DEFAULTS
3086
3087 016206 000015          GLBDEF: .WORD 15          ;NUMSYNC
3088 016210 013560          DMVDF3: .WORD 13560       ;CARRIER WAIT TIMING [6 SECS] 1130
3089 016212 000062          DMVDF4: .WORD 62           ;DELTA T 24
3090 016214 023420          DMVDF5: .WORD 23420      ;DEAD T 1750
3091 016216 000000          .WORD 0           ;POLL DELAY
3092 016220
3093
3094                                     ;*****
3095                                     ;
3096                                     ;* NOTE * - THE VALUES FOR DMVDF1-DMVDF5 ARE ASSEMBLED FOR DMP IF
3097                                     ; THIS IS A DMV THE INIT CODE CHANGES THESE VALUES TO DEFAULTS
3098                                     ; FOR DMV. THIS IS POSSIBLE BECUASE THIS PROGRAM WILL BE LOADED
3099                                     ; ONE TIME FOR EVERY DEVICE.
3100                                     ;*****
3101
3102                                     ;TRIB LIST OF POLL PARAMETERS
3103 016220 000400          POLLIS: .BLKW 8.*32.
3104
3105                                     ;GLOBAL LIST OF POLL PARAMETERS
3106 017220 000005          GLBPLS: .BLKW 5.
3107

```

3108	017232	000000	MPLY:	.WORD	0	:MULTIPLIER
3109	017234	000000	RXPTR:	.WORD	0	:RECEIVER MESSAGE POINTER
3110	017236	000000	TXPTR:	.WORD	0	:TRANSMITTER BUFFER POINTER
3111	017240	000000	CMPPTR:	.WORD	0	:COMPARISON BUFFER POINTER
3112	017242	000000	CMPTOT:	.WORD	0	:CMP MSG TOTAL
3113	017244	000000	CTOTCC:	.WORD	0	:COMPARE BUFFER CHAR. COUNT
3114	017246	000000	CCURAD:	.WORD	0	:CURRENT ADDR OF CMP BUFF TO ADD AT
3115	017250	000000	DVTXA:	.WORD	0	:DEVICE TX ADDR
3116	017252	000000	DVTCC:	.WORD	0	:DEVICE TX CHAR COUNT
3117	017254	000000	DVTTB:	.WORD	0	:DEVICE TRIBN
3118	017256	000000	DVTCT:	.WORD	0	:DEVICE TX MESSAGE COUNT
3119	017260	000000	TXMTOT:	.WORD	0	:TX MSG TOTAL
3120	017262	000000	TTOTCC:	.WORD	0	:TX BUFFER CHAR. COUNT
3121	017264	000000	TCURAD:	.WORD	0	:CURRENT ADDR. OF TX BUFF TO ADD AT
3122	017266	000000	DVRTB:	.WORD	0	:RECEIVE TRIBN
3123	017270	000000	DVRXA:	.WORD	0	:DEVICE RX ADDR
3124	017272	000000	DVRCC:	.WORD	0	:DEVICE RX CHAR COUNT
3125	017274	000000	DVRCT:	.WORD	0	:DEVICE RX MESSAGE COUNT
3126	017276	000000	RXMTOT:	.WORD	0	:RX MSG TOTAL
3127						
3128	017300	000000	LCNNT:	.WORD	0	:NUMBER OF OPERATOR AWAKE MSGS
3129	017302	000000	OPVAR:	.WORD	0	:HOLDER FOR OPTIONAL VARIABLE (1)
3130	017304	000000	OPVAR1:	.WORD	0	:HOLDER FOR OPTION VARIABLE (2)
3131	017306	000000	PSCNT:	.WORD	0	:PASS COUNTER
3132	017310	000000	ERRCNT:	.WORD	0	:ERROR COUNTER
3133	017312	000000	STADD:	.WORD	0	:START ADDR.
3134	017314	000000	ENADD:	.WORD	0	:END ADDR. FOR DUMP
3135	017316	000000	BYTBIT:	.WORD	0	:BYTE BIT FOR DUMP ROUTINE
3136	017320	000000	CLNSET:	.WORD	0	:CLEANSET FLAG SET AND CLEARED IN CLEAN UP
3137						:INDICATES TO OUTPUT HANDELER THAN NO OUPUTS SHOULD
3138						:BE PRINTED
3139	017322	000000	RQIFLG:	.WORD	0	:RQI FLAG
3140	017324	000000	FTLFLG:	.WORD	0	:USED AS FATEL ERROR FLAG
3141	017326	000000	TSSFLG:	.WORD	0	:USED AS TSS FLAG
3142	017330	000000	OVRCNT:	.WORD	0	:USED FOR QUE OVERFLOW FLAG
3143			;OTHER MESSAGE RELATED STORAGE LOCATIONS			
3144						
3145	017332	000000	MSGTYP:	.WORD	0	:TYPE OF DATA 0=0'S,1=1'S,2=10'S,3=01'S
3146						:4=CCITT,5=QUICK FOX,6=ALPHA/NUM,7=OPER
3147	017334	000000	CURCC:	.WORD	0	:TX/RX/CMP CHAR COUNT
3148	017336	000000	CPTRR:	.WORD	0	:CURRENT RX POINTER
3149	017340	000000	CPTR:	.WORD	0	:CURRENT POINTER
3150	017342	000000	CURADD:	.WORD	0	:CURRENT TX/RX/CMP START ADDD
3151	017344	000000	TOTCC:	.WORD	0	:TOTAL CHAR COUNT NOT MORE THEN 'BUFLIM'
3152	017346	000000	OFSET:	.WORD	0	:OFFSET COUNT
3153	017350	000000	TEMP:	.WORD	0	:TEMPORARY LOCATIONS (USED A LOT)
3154	017352	000000	TEMP1:	.WORD	0	
3155	017354	000000	TEMP2:	.WORD	0	
3156	017356	000000	TEMP3:	.WORD	0	
3157	017360	000000	TEMP4:	.WORD	0	
3158	017362	000000	TEMP5:	.WORD	0	
3159	017364	000000	SAVSP:	.WORD	0	:STACK POINTER SAVE AREA
3160	017366	000000	CONOTM:	.WORD	0	:CONTROL OUT ERROR MSG. ADDRESS AND TSS AND GSS MSGS.
3161	017370	000	GOOD:	.BYTE	0	:BYTE TO HOLD EXPECTED MESSAGE DATA BYTE FOR ERR REPORT
3162	017371	000	BAD:	.BYTE	0	:BYTE TO HOLD RECEIVED MESSAGE DATA BYTE FOR ERR REPORT
3163						

MESSAGE BUFFERS AND POINTER TABLES

```

3164                                     ;MORE INDEPENDENT CODE STORAGE LOCATIONS
3165
3166 017372 000000 LOGUNT: .WORD 0 ;LOC. TO HOLD LOGICAL UNIT NUMBER
3167 017374 000000 PCADD: .WORD 0 ;LOC. HOLD PC OF CALLING ROUTINE
3168 017376 000000 DCLFLG: .WORD 0 ;LOC. TO HOLD DO CLEAN FLAG 1 IF DOCLEAN INIT 0 IF NOT.
3169 017400 000000 RESFLG: .WORD 0 ;LOC TO HOLD FLAG (-1) THAT A RESTART WAS GIVEN
3170 017402 000000 MODTYP: .WORD 0 ;DCLT MODE OF OPERATION TYPE
3171 ; (0=REC-ONLY, 1=TX-ONLY, 2=PASSIVE-LOOPBK,
3172 ; 3=ACTIVE-LOOPBK, 4=DOWN L.L., 5=TALK, 6=LISTEN)
3173 017404 000000 MLTYP: .WORD 0 ;MAINTENANCE LOOP TYPE (0=NONE, 1=INTERNAL TTL,
3174 ; 2=CABLE, 3=MODEM-ANALOG LOOPBK (LOCAL),
3175 ; 4=MODEM-DIGITAL LOOPBK (REMOTE), 5=MOP)
3176 017406 000000 FHDPLX: .WORD 0 ;FULL OR HALF DUPLEX FLAG (1=FULL FROM P-TABLE)
3177 017410 000002 PARAM: .WORD 2 ;PROGRAM PARAMETERS
3178 ; BIT0= STATUS MSGS TO OPR PRINTED (1=YES)
3179 ; BIT1= DATA CHECKING DONE ON RCVD MSGS (1=YES)
3180 ; BIT2= ECHO (TRANSMIT) RCV'D MSG.(PASSIVE)(1=YES)
3181 ; BIT3= MODEM STATUS CHECK (1=YES)
3182 ; BIT4= CRC CALC./CHECK DONE (1=YES)
3183 ; BIT5= PROTOCOL EMULATION (1=YES)
3184 ; BIT6= SPARE
3185 017412 000000 RPASS: .WORD 0 ;PASS NUMBER FROM RUN COMMAND
3186 017414 000000 FLAG: .WORD 0 ;DEVICE FLAG WORD
3187
3188 ;MODE DISPATCH TABLE
3189 017416 060010 MODE: .WORD RXONLY ;RX ONLY DISPATCH
3190 017420 060036 .WORD TXONLY ;TX ONLY DISPATCH
3191 017422 060074 .WORD PLCK ;PASSIVE LOOP BACK DISP
3192 017424 060122 .WORD ALCK ;ACTIVE LOOP BACK DISP
3193 017426 061366 .WORD DLL ;DOWN LINE LOAD DISP
3194 017430 062244 .WORD TALCK ;TALK MODE DISPATCH
3195 017432 062476 .WORD LISCK ;LISTEN MODE DISPATCH
3196
3197
3198 .SBTTL CLOCK TABLES, EVENT LOG AND POINTERS
3199 017434 000000 CLKCSR: .WORD 0 ;CLOCK CSR ADDRESS
3200 017436 000000 CLKBR: .WORD 0 ;CLOCK INTERRUPT LEVEL
3201 017440 000000 CLKVEC: .WORD 0 ;CLOCK INTERRUPT VECTOR
3202 017442 000074 CLKHZ: .WORD 60. ;CLOCK'S HERTZ RATE
3203 017444 000000 CLKEN: .WORD 0 ;CLOCK'S CSR VALUE TO INTRPT. ENABLE IT
3204
3205 017446 000000 TIMMIN: .WORD 0 ;PLACE TO KEEP TIME-SINCE-START
3206 017450 000000 TIMSEC: .WORD 0
3207 017452 000000 TIMTCK: .WORD 0 ;PLACE TO KEEP # OF TICKS/SEC
3208
3209 017454 000000 TIMER1: .WORD 0 ;EVENT TIMER #1 (TICKS)
3210 017456 000000 TIMER2: .WORD 0 ;EVENT TIMER #2 (TICKS)
3211 017460 000000 TIMERS: .WORD 0 ;EVENT TIMER #3 (SECONDS)
3212

```

3213			:EVENT LOG TABLE AND ITS NEXT ENTRY POINTER
3214	017462	017464	EVTPTL: .WORD EVTLOG ;POINTER TO NEXT FREE SPACE IN EVENT LOG
3215	017464	000416	EVTLOG: .BLKW 270. ;EVENT LOG BUFFER
3216	020520	000001	EVTEND: .BLKW 1. ;APPROXIMATE END OF EVENT TABLE (ALLOWS CIRCULAR QUE)
3217			
3218			.SBTTL MODEM DATA SECTION
3219			
3220	020522	000000	MODS: .WORD 0 ;MODEM STATUS
3221			

```

3222 ;TABLE OF MODEM SIGNAL BIT DEFINITIONS
3223
3224 020524 000004 MOBITS: .WORD CTS ;CLEAR TO SEND (CIRCUIT CB)
3225 020526 000010 .WORD DSR ;DATA SET READY (CIRCUIT CC)
3226 020530 000001 .WORD DCD ;DATA CARRIER DETECT (CIRCUIT CF)
3227 020532 000040 .WORD RTS ;REQUEST TO SEND (CIRCUIT CA)
3228 020534 000200 .WORD RI ;RING INDICATOR (CIRCUIT CE)
3229 020536 040000 .WORD SQD ;SIGNAL QUALITY DETECT (CIRCUIT CG)
3230 020540 002000 .WORD TM ;MODEM IN TEST MODE (RS 449 ONLY CIRCUIT TM)
3231 020542 MOBITE:
  
```

```

3232 ;TABLE OF ADDRESSES OF MODEM SIGNAL MESSAGE POSITIONS
3233
3234 MOMSGS: .WORD EVMCTS ;CLEAR TO SEND (CIRCUIT CB)
3235 020542 031526 .WORD EVMSDR ;DATA SET READY (CIRCUIT CC)
3236 020544 031532 .WORD EVMDCD ;DATA CARRIER DETECT (CIRCUIT CF)
3237 020546 031536 .WORD EVMRTS ;REQUEST TO SEND (CIRCUIT CA)
3238 020550 031542 .WORD EVMRI ;RING INDICATOR (CIRCUIT CE)
3239 020552 031546 .WORD EVMSQD ;SIGNAL QUALITY DETECT (CIRCUIT CG)
3240 020554 031552 .WORD EVMTM ;MODEM IN TEST MODE (RS 449 ONLY CIRCUIT TM)
3241 020556 031556
  
```

```

3242 ;TABLE OF ADDRESSES OF EVENT DESCRIPTION MESSAGES
3243 ; ORDER CORRESPONDS TO MESSAGE TYPE VALUES
3244
3245 EVTLST: .WORD EDTXQ ;TRANSMIT MESSAGE QUEUED
3246 020560 027777 .WORD EDTXC ;TRANSMIT OF MESSAGE COMPLETE
3247 020562 030023 .WORD EDRXQ ;RECEIVE MESSAGE SPACE QUEUED
3248 020564 030052 .WORD EDRXC ;MESSAGE RECEIVED - RECEIVE COMPLETE
3249 020566 030077 .WORD EDDER ;DEVICE INFORMATION
3250 020570 030125 .WORD EDDVI ;DEVICE INITIALIZE STARTED
3251 020572 030172 .WORD EDDCK ;DATA COMPARISON DONE
3252 020574 030142 .WORD LPO ;NULL STRING
3253 020576 026763 .WORD EDDLE ;DATA COMPARE LENGTH ERROR
3254 020600 030220 .WORD EDDDE ;DATA COMPARE DATA ERROR
3255 020602 030255 .WORD EDEOP ;END OF PASS
3256 020604 030310
  
```

```

3257 ;LOCATIONS USED DURING EVENT REPORTING
3258
3259 EVTSEC: .WORD 0 ;TEMPORARY LOCS TO KEEP EVENT TIME WHILE REPORTING
3260 020606 000000
3261 020610 000000 EVTMIN: .WORD 0
3262 020612 000000 EVTTCK: .WORD 0
3263 020614 000000 EVTADD: .WORD 0 ;TEMP. LOC. TO HOLD ADDRESS DURING EVENT REPORTING
3264 020616 000000 EVTBCT: .WORD 0 ; " " " " BYTE COUNT " " " "
3265 020620 000000 EVTTMP: .WORD 0 ; " " " " OTHER DATA " " " "
  
```

```

3266 ;REPORT CODING DISPATCH TABLE
3267
3268 RPTDSP: .WORD RPTTXQ ;TRANSMIT QUEUED ENTRY DECODING
3269 020622 043442 .WORD RPTTXQ ;TRANSMIT COMPLETE ENTRY DECODING
3270 020624 043442 .WORD RPTTXQ ;RECEIVER QUEUED ENTRY DECODING
3271 020626 043442 .WORD RPTTXQ ;RECEIVER COMPLETE ENTRY DECODING
3272 020630 043442 .WORD RPTDR ;DEVICE ERROR ENTRY DECODING
3273 020632 043520 .WORD RPTDVI ;DEVICE INIT ENTRY DECODING
3274 020634 043620 .WORD RPTDCK ;DATA COMPARISON ENTRY DECODING
3275 020636 044054 .WORD RPTMSC ;PLACE HOLDER
3276 020640 044134 .WORD RPTDLE ;DATA COMPARISON LENGTH ERROR
3277 020642 044054
  
```


3278	020644	043774	.WORD	RPTDDE	:DATA COMPARISON DATA ERROR
3279	020646	043670	.WORD	RPTTEOP	:END OF PASS
3280					
3281					
3282	020650	000000	DEV1:	.WORD	0
3283	020652	000000	DEV2:	.WORD	0
3284	020654	000000	DEV3:	.WORD	0
3285	020656	000000	DEV4:	.WORD	0

:TEMP LOCS TO HOLD DATA FOR EVENT REPORTING
: AND SHOW MODE,... SUBROUTINE

Address	SBTTL	TSSLST	TABLE FOR TSS ASCII AND ROUTINES
3286			
3287	020660	032110	.WORD TSS0A ; POINTER FOR OFFSET 0 ASCII
3288	020662	032144	.WORD TSS1A ; POINTER FOR OFFSET 1 ASCII
3289	020664	032214	.WORD TSS2A ; POINTER FOR OFFSET 2 ASCII
3290	020666	032250	.WORD TSS3A ; POINTER FOR OFFSET 3 ASCII
3291	020670	032323	.WORD TSS4A ; POINTER FOR OFFSET 4 ASCII
3292	020672	032371	.WORD TSS5A ; POINTER FOR OFFSET 5 ASCII
3293	020674	032453	.WORD TSS6A ; POINTER FOR OFFSET 6 ASCII
3294	020676	032543	.WORD TSS7A ; POINTER FOR OFFSET 7 ASCII
3295	020700	032601	.WORD TSS10A ; POINTER FOR OFFSET 10 ASCII
3296	020702	032635	.WORD TSS11A ; POINTER FOR OFFSET 11 ASCII
3297	020704	032673	.WORD TSS12A ; POINTER FOR OFFSET 12 ASCII
3298	020706	032770	.WORD TSS13A ; POINTER FOR OFFSET 13 ASCII
3299	020710	033064	.WORD TSS14A ; POINTER FOR OFFSET 14 ASCII
3300	020712	033147	.WORD TSS15A ; POINTER FOR OFFSET 15 ASCII
3301	020714	033233	.WORD TSS16A ; POINTER FOR OFFSET 16 ASCII
3302	020716	033316	.WORD TSS17A ; POINTER FOR OFFSET 17 ASCII
3303	020720	033402	.WORD TSS20A ; POINTER FOR OFFSET 20 ASCII
3304	020722	033474	.WORD TSS21A ; POINTER FOR OFFSET 21 ASCII
3305	020724	033563	.WORD TSS22A ; POINTER FOR OFFSET 22 ASCII
3306	020726	033651	.WORD TSS23A ; POINTER FOR OFFSET 23 ASCII
3307	020730	033736	.WORD TSS24A ; POINTER FOR OFFSET 24 ASCII
3308	020732	034025	.WORD TSS25A ; POINTER FOR OFFSET 25 ASCII
3309	020734	034112	.WORD TSS26A ; POINTER FOR OFFSET 26 ASCII
3310	020736	034143	.WORD TSS27A ; POINTER FOR OFFSET 27 ASCII
3311	020740	034226	.WORD TSS30A ; POINTER FOR OFFSET 30 ASCII
3312	020742	034300	.WORD TSS31A ; POINTER FOR OFFSET 31 ASCII
3313	020744	034357	.WORD TSS32A ; POINTER FOR OFFSET 32 ASCII
3314	020746	034442	.WORD TSS33A ; POINTER FOR OFFSET 33 ASCII
3315	020750	034525	.WORD TSS34A ; POINTER FOR OFFSET 34 ASCII
3316	020752	034604	.WORD TSS35A ; POINTER FOR OFFSET 35 ASCII
3317	020754	034663	.WORD TSS36A ; POINTER FOR OFFSET 36 ASCII
3318	020756	034735	.WORD TSS37A ; POINTER FOR OFFSET 37 ASCII
3319			

:TABLE FOR TSS ACTION ROUTINES
 :IF BYTE = 0 USE WORD ROUTINE
 :IF BYTE = 2 USE BYTE/BYTE ROUTINE
 :IF BYTE = 4 USE BYTE SPECIAL ROUTINE

Address	TSSIND	INDEX FOR TSS
3320		
3321		
3322		
3323		
3324		
3325	020760 000	.BYTE 0 ; INDEX FOR TSS 0
3326	020761 002	.BYTE 2 ; INDEX FOR TSS 1
3327	020762 000	.BYTE 0 ; INDEX FOR TSS 2
3328	020763 002	.BYTE 2 ; INDEX FOR TSS 3
3329	020764 002	.BYTE 2 ; INDEX FOR TSS 4
3330	020765 002	.BYTE 2 ; INDEX FOR TSS 5
3331	020766 002	.BYTE 2 ; INDEX FOR TSS 6
3332	020767 000	.BYTE 0 ; INDEX FOR TSS 7
3333	020770 000	.BYTE 0 ; INDEX FOR TSS 10
3334	020771 000	.BYTE 0 ; INDEX FOR TSS 11
3335	020772 004	.BYTE 4 ; INDEX FOR TSS 12
3336	020773 004	.BYTE 4 ; INDEX FOR TSS 13
3337	020774 004	.BYTE 4 ; INDEX FOR TSS 14
3338	020775 004	.BYTE 4 ; INDEX FOR TSS 15
3339	020776 004	.BYTE 4 ; INDEX FOR TSS 16
3340	020777 002	.BYTE 2 ; INDEX FOR TSS 17
3341	021000 002	.BYTE 2 ; INDEX FOR TSS 20

3342	021001	002	.BYTE	2	:INDEX FOR TSS	21
3343	021002	002	.BYTE	2	:INDEX FOR TSS	22
3344	021003	002	.BYTE	2	:INDEX FOR TSS	23
3345	021004	002	.BYTE	2	:INDEX FOR TSS	24
3346	021005	002	.BYTE	2	:INDEX FOR TSS	25
3347	021006	000	.BYTE	0	:INDEX FOR TSS	26
3348	021007	002	.BYTE	2	:INDEX FOR TSS	27
3349	021010	000	.BYTE	0	:INDEX FOR TSS	30
3350	021011	002	.BYTE	2	:INDEX FOR TSS	31
3351	021012	002	.BYTE	2	:INDEX FOR TSS	32
3352	021013	002	.BYTE	2	:INDEX FOR TSS	33
3353	021014	002	.BYTE	2	:INDEX FOR TSS	34
3354	021015	002	.BYTE	2	:INDEX FOR TSS	35
3355	021016	000	.BYTE	0	:INDEX FOR TSS	36
3356	021017	000	.BYTE	0	:INDEX FOR TSS	37
3357						
3358	021020	000000	TSSE:	.WORD	0	:WORD FOR LAST TSS TO BE PRINTED
3359	021022	000000	TSSA:	.WORD	0	:WORD FOR ADDRESS
3360	021024	000000	TSSKEY:	.WORD	0	:KEY WORD FOR READING TSS
3361						

			.SBTTL TABLE FOR GSS ASCII AND ACTION			
3362						
3363						
3364						
3365	021026	035002	GSSLST: .WORD	GSS0A	:POINTER FOR OFFSET	0 ASCII
3366	021030	035043	.WORD	GSS1A	:POINTER FOR OFFSET	1 ASCII
3367	021032	035101	.WORD	GSS2A	:POINTER FOR OFFSET	2 ASCII
3368	021034	035140	.WORD	GSS3A	:POINTER FOR OFFSET	3 ASCII
3369	021036	035175	.WORD	GSS4A	:POINTER FOR OFFSET	4 ASCII
3370	021040	035232	.WORD	GSS5A	:POINTER FOR OFFSET	5 ASCII
3371	021042	035267	.WORD	GSS6A	:POINTER FOR OFFSET	6 ASCII
3372	021044	035345	.WORD	GSS7A	:POINTER FOR OFFSET	7 ASCII
3373	021046	035423	.WORD	GSS10A	:POINTER FOR OFFSET	10 ASCII
3374	021050	035475	.WORD	GSS11A	:POINTER FOR OFFSET	11 ASCII
3375	021052	035515	.WORD	GSS12A	:POINTER FOR OFFSET	12 ASCII
3376	021054	035553	.WORD	GSS13A	:POINTER FOR OFFSET	13 ASCII
3377	021056	035614	.WORD	GSS14A	:POINTER FOR OFFSET	14 ASCII
3378	021060	035635	.WORD	GSS15A	:POINTER FOR OFFSET	15 ASCII
3379	021062	035745	.WORD	GSS16A	:POINTER FOR OFFSET	16 ASCII
3380	021064	036055	.WORD	GSS17A	:POINTER FOR OFFSET	17 ASCII
3381	021066	036137	.WORD	GSS20A	:POINTER FOR OFFSET	20 ASCII
3382	021070	036204	.WORD	GSS21A	:POINTER FOR OFFSET	21 ASCII
3383	021072	036251	.WORD	GSS22A	:POINTER FOR OFFSET	22 ASCII
3384	021074	036316	.WORD	GSS23A	:POINTER FOR OFFSET	23 ASCII
3385	021076	036363	.WORD	GSS24A	:POINTER FOR OFFSET	24 ASCII
3386	021100	036430	.WORD	GSS25A	:POINTER FOR OFFSET	25 ASCII
3387	021102	036475	.WORD	GSS26A	:POINTER FOR OFFSET	26 ASCII
3388	021104	036517	.WORD	GSS27A	:POINTER FOR OFFSET	27 ASCII
3389	021106	036541	.WORD	GSS30A	:POINTER FOR OFFSET	30 ASCII
3390	021110	036574	.WORD	GSS31A	:POINTER FOR OFFSET	31 ASCII
3391	021112	036651	.WORD	GSS32A	:POINTER FOR OFFSET	32 ASCII
3392	021114	036734	.WORD	GSS33A	:POINTER FOR OFFSET	33 ASCII
3393	021116	036757	.WORD	GSS34A	:POINTER FOR OFFSET	34 ASCII
3394	021120	037024	.WORD	GSS35A	:POINTER FOR OFFSET	35 ASCII
3395	021122	037046	.WORD	GSS36A	:POINTER FOR OFFSET	36 ASCII
3396	021124	037067	.WORD	GSS37A	:POINTER FOR OFFSET	37 ASCII

:TABLE FOR GSS ACTION ROUTINES
 :IF BYTE = 0 USE WORD ROUTINE
 :IF BYTE = 2 USE BYTE/BYTE ROUTINE
 :IF BYTE = 4 USE BYTE SPECIAL ROUTINE

			GSSIND: .BYTE			
3402						
3403	021126	002	.BYTE	2	:INDEX FOR	GSS 0
3404	021127	002	.BYTE	2	:INDEX FOR	GSS 1
3405	021130	002	.BYTE	2	:INDEX FOR	GSS 2
3406	021131	002	.BYTE	2	:INDEX FOR	GSS 3
3407	021132	002	.BYTE	2	:INDEX FOR	GSS 4
3408	021133	002	.BYTE	2	:INDEX FOR	GSS 5
3409	021134	002	.BYTE	2	:INDEX FOR	GSS 6
3410	021135	002	.BYTE	2	:INDEX FOR	GSS 7
3411	021136	002	.BYTE	2	:INDEX FOR	GSS 10
3412	021137	000	.BYTE	0	:INDEX FOR	GSS 11
3413	021140	002	.BYTE	2	:INDEX FOR	GSS 12
3414	021141	002	.BYTE	2	:INDEX FOR	GSS 13
3415	021142	000	.BYTE	0	:INDEX FOR	GSS 14
3416	021143	004	.BYTE	4	:INDEX FOR	GSS 15
3417	021144	004	.BYTE	4	:INDEX FOR	GSS 16

3418	021145	002	.BYTE	2	; INDEX FOR	GSS	17
3419	021146	002	.BYTE	2	; INDEX FOR	GSS	20
3420	021147	002	.BYTE	2	; INDEX FOR	GSS	21
3421	021150	002	.BYTE	2	; INDEX FOR	GSS	22
3422	021151	002	.BYTE	2	; INDEX FOR	GSS	23
3423	021152	002	.BYTE	2	; INDEX FOR	GSS	24
3424	021153	002	.BYTE	2	; INDEX FOR	GSS	25
3425	021154	000	.BYTE	0	; INDEX FOR	GSS	26
3426	021155	000	.BYTE	0	; INDEX FOR	GSS	27
3427	021156	000	.BYTE	0	; INDEX FOR	GSS	30
3428	021157	002	.BYTE	2	; INDEX FOR	GSS	31
3429	021160	002	.BYTE	2	; INDEX FOR	GSS	32
3430	021161	000	.BYTE	0	; INDEX FOR	GSS	33
3431	021162	000	.BYTE	0	; INDEX FOR	GSS	34
3432	021163	000	.BYTE	0	; INDEX FOR	GSS	35
3433	021164	000	.BYTE	0	; INDEX FOR	GSS	36
3434	021165	000	.BYTE	0	; INDEX FOR	GSS	37
3435							

3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456
3457
3458
3459
3460
3461
3462
3463
3464
3465
3466
3467
3468
3469
3470
3471
3472
3473
3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491

021166

021166
021172
021176
021200
021214
021216
021232
021234
021250
021252
021264
021270
021304
021310
021324
021330
021344
021350
021354
021366
021372
021404
021410

021412
021416
021432
021436
021454
021460
021476
021502
021520
021524
021542
021546

```
.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 NODE DISPLMNT !          ONLY IF 'ASCII' ARGUMENT DEFINED
-----
! ASCIZ MATCH STRING !          ONLY IF 'ASCII' ARGUMENT DEFINED
!          (.EVEN) !
-----

CLITRE:

;FIRST KEYWORD
N10$:  CLI      CLISPA,0,N10$          ;SKIP ANY LEADING SPACES
N42$:  CLI      <'?',HLP,N42$         ;IS THE FIRST NON-SP CHAR A '?'
N43$:  CLI      CLISTR,HLP,N43$,<'HELP'> ; IF YES DO 'HLP' AND EXIT
N44$:  CLI      CLISTR,PRNT,N44$,<'PRINT'> ;ELSE, IS FIRST WORD A 'HELP'
N45$:  CLI      CLISTR,EXIT,N45$,<'EXIT'> ; IF YES DO 'HLP' AND EXIT
N46$:  CLI      CLISTR,RUN,N46$,<'RUN'> ;ELSE, IS FIRST WORD A 'PRINT'
N40$:  CLI      CLIBR,0,N80$          ; IF YES DO 'PRINT' AND EXIT
N47$:  CLI      CLISTR,CLEAR,N47$,<'CLEAR'> ;ELSE, IS FIRST WORD A 'EXIT'
N20$:  CLI      CLISTR,NOTNUF,N40$,<'DUMP'> ; IF YES DO 'EXIT' AND EXIT
N25$:  CLI      CLISTR,SHOW,N25$,<'HOW'> ;ELSE, IS FIRST WORD A 'RUN'
N30$:  CLI      CLIBR,0,N110$         ; IF YES DO 'RUN' & GOTO N80$
N80$:  CLI      CLISPA,0,N30$         ;ELSE, IS FIRST WORD A 'DUMP'
N81$:  CLI      CLISTR,NOTNUF,N30$,<'MODE'> ; IF YES GOTO N80$
N82$:  CLI      CLISTR,ATVMOD,N82$,<'ACTIVE'> ;ELSE, IS FIRST WORD A 'CLEAR'
N83$:  CLI      CLISTR,PASMOD,N83$,<'PASSIVE'> ; IF YES DO 'CLR' & GOTO N100$
N84$:  CLI      CLISTR,RECMOD,N84$,<'RECEIVE'> ; ELSE IS FIRST WORD TRIB
N85$:  CLI      CLISTR,LISMOD,N85$,<'LISTEN'> ; ELSE, IS FIRST CHAR. A 'S'
N86$:  CLI      CLISTR,DLLMOD,N86$,<'DOWNLINELOAD'> ; IF YES IS REST OF WORD 'HOW'
N110$: CLI      CLIBR,0,N110$         ; IF YES, DO 'SHOW',BR N100$
N115$: CLI      CLIBR,0,N115$         ; ELSE, IS REST OF WORD 'ET'
N116$: CLI      CLIBR,0,N116$         ; IF YES, DO 'SET', BR N110$
N117$: CLI      CLIBR,0,N117$         ; OTHERWISE "ILL CMD" - EXIT

;SECOND KEYWORD (MODE=) FOR RUN COMMAND
N80$:  CLI      CLISPA,0,N30$          ;SKIP LEADING SPS, IF NONE-ERR
N81$:  CLI      CLISTR,NOTNUF,N30$,<'MODE'> ;IS NEXT WORD 'MODE='
N82$:  CLI      CLISTR,ATVMOD,N82$,<'ACTIVE'> ; IF NO, IT'S WRONG -ERR -EXIT
N83$:  CLI      CLIBR,0,N115$         ;IS NEXT WORD 'ACTIVE'
N84$:  CLI      CLISTR,PASMOD,N83$,<'PASSIVE'> ; IF YES, DO 'ACTIVE',BR N115$
N85$:  CLI      CLIBR,0,N115$         ;IS NEXT WORD 'PASSIVE'
N86$:  CLI      CLISTR,RECMOD,N84$,<'RECEIVE'> ; IF YES, DO 'PASSVE',BR N115$
N87$:  CLI      CLIBR,0,N115$         ;IS NEXT WORD 'RECEIVE'
N88$:  CLI      CLISTR,LISMOD,N85$,<'LISTEN'> ; IF YES, DO 'RECVE',BR N115$
N89$:  CLI      CLIBR,0,N115$         ;IS NEXT WORD 'LISTEN'
N90$:  CLI      CLISTR,DLLMOD,N86$,<'DOWNLINELOAD'> ; IF YES, DO 'LISTEN',BR N115$
N91$:  CLI      CLIBR,0,N115$         ; IS NEXT WORD 'DOW...'
```

```

3492 021572
3493 021576
3494 021602
3495 021620
3496 021624
3497 021636
3498
3499
3500
3501 021642
3502 021646
3503 021670
3504 021672
3505 021716
3506
3507
3508
3509
3510 021720
3511 021724
3512 021744
3513 021750
3514 021772
3515
3516
3517 021776
3518 022002
3519 022006
3520 022012
3521 022016
3522 022022
3523 022026
3524 022032
3525
3526
3527 022036
3528 022042
3529 022046
3530 022060
3531 022064
3532 022100
3533
3534
3535 022104
3536 022122
3537 022126
3538 022142
3539
3540 022146
3541 022162
3542
3543 022166
3544 022202
3545
3546 022206
3547 022222

N86$: CLI CLIBR,0,N115$ ; IF YES, DO 'DWNLL',BR N115$
      CLI <'T>,0,N30$ ; IS NEXT CHAR A 'T'
      CLI CLISTR,TRAMOD,N87$,<'RANSMIT'> ; IS REST OF WORD 'RANSMIT'
      CLI CLIBR,0,N115$ ; IF YES, DO 'TRANSM',BR N115$
N87$: CLI CLISTR,TALMOD,N30$,<'ALK'> ; IS REST OF WORD 'ALK'
      CLI CLIBR,0,N115$ ; IF YES, DO 'TALK',BR N115$
      ; IF NO, ERROR - EXIT

;SECOND KEYWORD (FOR CLEAR OR SHOW)
N100$: CLI CLISPA,0,N30$ ;SKIP LEADING SPACES, NONE=ERR
N102$: CLI CLISTR,CSHEXP,N104$,<'EXPECTBUFF'> ;IS NEXT WORD 'EXPE...'
      CLI CLIEXI,0 ; IF YES, DO CLR-EXP,EXIT
N104$: CLI CLISTR,CSHTRN,N30$,<'TRANSMITBUFF'> ;IS NEXT WORD 'TRANS...'
      CLI CLIEXI,0 ; IF YES, DO CLR-TRN,EXIT
      ; IF NO - ERROR - EXIT

;SECOND KEYWORD (FOR SET)
N110$: CLI CLISPA,0,N30$
N111$: CLI CLISTR,SETEXP,N112$,<'EXPECTMSG'>
      CLI CLIBR,0,N120$
N112$: CLI CLISTR,SETTRN,N30$,<'TRANSMITMSG'>
      CLI CLIBR,0,N120$

;GET ADDRESSES FOR DUMP COMMAND
N50$: CLI CLIALP,0,N51$
N51$: CLI CLISPA,0,N52$
N52$: CLI CLIOCT,DMP5,N30$
      CLI <'>,NOTNUF,N125$
      CLI CLIOCT,DMPE,N30$
      CLI <'>,NOTNUF,N125$
      CLI <'B>,DMPQ,N30$
      CLI CLIBR,0,N125$

;QUALIFIERS FOR THE RUN COMMAND
N115$: CLI CLIALP,0,N114$
N114$: CLI <'>,NOTNUF,N125$
      CLI CLISTR,NO,N116$,<'NO'>
N116$: CLI <'C>,0,N117$
      CLI CLISTR,CHECK,N117$,<'HECK'>
      CLI CLIBR,0,N115$

N117$: CLI CLISTR,STATUS,N118$,<'STATUS'>
      CLI CLIBR,0,N115$
N118$: CLI CLISTR,ECHO,N130$,<'ECHO'>
      CLI CLIBR,0,N115$

N130$: CLI CLISTR,0,N132$,<'PASS'>
      CLI CLIBR,0,N150$

N132$: CLI CLISTR,MOSC,N131$,<'MODEM'>
      CLI CLIBR,0,N115$

N131$: CLI CLISTR,0,N30$,<'LOOP'>
      CLI CLIBR,0,N140$

```

```
3548
3549 ;GET MESSAGE TYPE FOR SET MESSAGE COMMANDS
3550 022226 N120$: CLI <'=>,0,N30$
3551
3552 ; LOOK FOR DEFAULT MESSAGE NAME
3553 022232 N60$: CLI CLISTR,MSG1,N61$,<'ONES'>
3554 022246 CLI CLIBR,0,N121$
3555 022252 N61$: CLI CLISTR,MSG0,N62$,<'ZEROES'>
3556 022270 CLI CLIBR,0,N121$
3557 022274 N62$: CLI CLISTR,MSG2,N63$,<'1ALT'>
3558 022310 CLI CLIBR,0,N121$
3559 022314 N63$: CLI CLISTR,MSG3,N64$,<'0ALT'>
3560 022330 CLI CLIBR,0,N121$
3561 022334 N64$: CLI CLISTR,MSG5,N65$,<'ITEP'>
3562 022350 CLI CLIBR,0,N121$
3563 022354 N65$: CLI CLISTR,MSG4,N66$,<'CCITT'>
3564 022370 CLI CLIBR,0,N121$
3565 022374 N66$: CLI CLISTR,MSG6,N67$,<'ALPHA'>
3566 022410 CLI CLIBR,0,N121$
3567 022414 N67$: CLI CLISTR,SETET,N68$,<'TRANSMIT'>
3568 022434 CLI CLIBR,0,N125$
3569
3570 ; LOOK FOR QUOTED MESSAGE
3571 022440 N68$: CLI <'>,OPRMSG,N30$
3572 022444 N70$: CLI <'>,ENDQO,N71$
3573 022450 CLI CLIBR,0,N121$
3574 022454 N71$: CLI CLISPA,0,N72$
3575 022460 N72$: CLI CLIALN,0,N73$ ;ONLY A-Z,SP,TAB, OR 0-9 BETWEEN ''S
3576 022464 CLI CLIBR,0,N70$ ;PRINT ERROR IF NONE LEGAL CHAR FOR ''S
3577 022470 N73$: CLI CLIERR,BADCHR
3578
3579 ;GET QUALIFIERS (SIZE OR COPY) FOR SET MESSAGE COMMANDS
3580 022472 N121$: CLI CLIALP,0,N123$
3581 022476 N123$: CLI <'>,NOTNUF,N125$
3582 022502 CLI CLISTR,SIZE,N122$,<'SIZE'>
3583 022516 CLI CLIBR,0,N126$
3584 022522 N122$: CLI CLISTR,QCOPY,N30$,<'COPY'>
3585 022536 CLI CLIBR,0,N126$
3586
3587 ;NUMER FOR SIZE OR COPY
3588 022542 N126$: CLI <'=>,0,N30$
3589 022546 CLI CLIDEC,NUM,N30$
3590 022552 CLI CLIBR,0,N121$
3591
3592 ;GET MAINTENANCE LOOP TYPE FOR RUN 'LOOP' QUALIFIER
3593 022556 N140$: CLI <'=>,0,N30$
3594
3595
3596 022562 N141$: CLI CLISTR,TTLLOP,N142$,<'INTERNAL TTL'>
3597 022604 CLI CLIBR,0,N115$
3598 022610 N142$: CLI CLISTR,CBLLOP,N143$,<'CABLE'>
3599 022624 CLI CLIBR,0,N115$
3600 022630 N143$: CLI CLISTR,LMDLOP,N144$,<'LOCALMODEM'>
3601 022652 CLI CLIBR,0,N115$
3602 022656 N144$: CLI CLISTR,RMDLOP,N30$,<'REMOTEMODEM'>
3603 022700 CLI CLIBR,0,N115$
```



```
3604  
3605 ;GET LINE NUMBER FOR 'PASS' RUN QUALIFIER  
3606 022704 N150$: CLI <'=>,0,N30$  
3607 022710 CLI CLIDEC,PASC,N30$  
3608 022714 CLI CLIBR,0,N115$  
3609 ;GET TRIB SHOW OR ADDR FOR KILL OR ESTABLISH  
3610 022720 N105$: CLI CLISPA,NOTNUF,N106$  
3611 022724 N106$: CLI CLISTR,SLST,N107$,<'SHOW'>  
3612 022740 CLI CLIEXI,0  
3613 022742 N107$: CLI CLISTR,ETRB,N108$,<'ESTABLISH'>  
3614 022762 CLI CLIBR,0,N160$  
3615 022766 N108$: CLI CLISTR,KTRB,N30$,<'KILL'>  
3616 023002 N160$: CLI <'=>,0,N30$  
3617 023006 N161$: CLI CLISTR,KALL,N162$,<'ALL'>  
3618 023020 N162$: CLI CLIDEC,EKTB,N30$  
3619 023024 CLI CLISTR,ETWS,N163$,<'/W'>  
3620 023036 N163$: CLI 54,NOTNUF,N125$ ;LOOKING FOR ''''  
3621 023042 CLI CLIBR,0,N161$  
3622  
3623 ;END-OF-LINE  
3624 023046 N125$: CLI CLIEXI,0  
3625
```

```

3626
3627
3628 ;DEVICE DEPENDENT STORAGE LOCATIONS FOR
3629 ; CURRENT DEVICE PARAMTERS
3630
3631 SELO:
3632 023050 BSELO: .WORD 0 ;ADDRESSES OF REGISTERS SELO THRU BSEL7
3633 023052 BSEL1: .WORD 0
3634 023054 SEL2:
3635 023054 BSEL2: .WORD 0
3636 023056 BSEL3: .WORD 0
3637 023060 SEL4:
3638 023060 BSEL4: .WORD 0
3639 023062 BSEL5: .WORD 0
3640 023064 SEL6:
3641 023064 BSEL6: .WORD 0
3642 023066 BSEL7: .WORD 0
3643
3644
3645 023070 INVEC: .WORD 0 ;INPUT INTERRUPT VECTOR ADDRESS
3646 023072 OUTVEC: .WORD 0 ;OUTPUT INTERRUPT VECTOR ADDRESS
3647 023074 INTPRI: .WORD 0 ;INTERRUPT PRIORITY
3648 023076 OPTYP: .WORD 0 ;OPTION TYPE
3649 023100 DEVPAR: .WORD 0 ;DEVICE PARAM. BIT 0 BIT1
3650 ; 1 MTP CONT
3651 ; 0 PTP TRIB
3652 023102 STATYP: .WORD 0 ;STATION TYPE
3653 ; DEVICE ERROR MESG TABLES
3654 023104 CONOLS: .WORD 0 ;TABLE HOLDER
3655 023106 .WORD RXTHEM ;RX THRESHOLD ERROR MESSAGE ADDR.
3656 023110 .WORD TXTHEM ;TX THRESHOLD ERROR MESSAGE ADDR
3657 023112 .WORD SLTHEM ;SELECT THRESHOLD MESSAGE
3658 023114 .WORD STRCM ;DDCMP START REC MESSAGE ADDR.
3659 023116 .WORD MARM ;DDCMP MAINT REC IN RUN
3660 023120 .WORD MARHM ;MAINT RECEIVED IN HALD
3661 023122 .WORD STRMM ;START REC. IN MAINT MESSAGE.
3662 023124 .WORD PE142M ;SPARE
3663 023126 .WORD DEADTM ;DEAD TRIB MESSAGE
3664 023130 .WORD RUSM ;RUN STATE SET IN ERROR
3665 023132 .WORD BABTM ;BABLING TRIB MESSAGE
3666 023134 .WORD STREAM ;STREAMING TRIB MESSAGE
3667 023136 .WORD RIM ;RING DETECTED
3668 ;PROCEDURE ERRORS
3669
3670 023140 CONO1S: .WORD PE100M ;NO MODE DEF
3671 023142 .WORD PE102M ;ILLEGAL TYPE
3672 023144 .WORD PE104M ;ILLEGAL MODE CHANGE
3673 023146 .WORD PE106M ;CONTROL IN TO UNESTABLISHED TRIB
3674 023150 .WORD PE110M ;NON-GLOBAL TO TRIB 0
3675 023152 .WORD PE112M ;ILLEGAL REQUEST
3676 023154 .WORD PE114M ;ATTEMPT TO ESTABLISH MORE THAN MAX TRIBS
3677 023156 .WORD PE116M ;ESTABLISH TO ALREADY ESTABLISHED
3678 023160 .WORD PE120M ;ILLEGAL CONTROL IN
3679 023162 .WORD PE122M ;ASSIGN BUFFER FOR UNESTABLISHED TRIB
3680 023164 .WORD PE124M ;ASSIGN BUFFER FOR HALTED TRIB
3681 023166 .WORD PE126M ;ASSIGN BUFFER WITH BYTE COUNT =0
  
```

3682	023170	040726	.WORD	PE130M	:ASSIGN TX BUFFER TO TRIB 0	
3683	023172	040754	.WORD	PE132M	:ATTEMPT TO R/W RESERVED TSS/GSS	
3684	023174	041000	.WORD	PE134M	:USING RESERVED BITS IN BSEL7	
3685	023176	041032	.WORD	PE136M	:COMMON POOL ERROR	
3686	023200	041054	.WORD	PE140M	:COMMON POOL QUOTA ERROR	
3687	023202	041073	.WORD	PE142M	:SPARE	
3688	023204	041101	.WORD	PE144M	:SPARE	
3689						
3690	023206	041107	CONO3S:	.WORD	BUFTSM	:BUFFER TOO SMALL
3691	023210	041130		.WORD	NOEXM	:NONESTANT MEM
3692	023212	041146		.WORD	DISCON	:DISCON MESSAGE
3693	023214	041160		.WORD	QUEOM	:QUEOVER M.
3694	023216	041173		.WORD	CARLOS	:CARRIER LOSS
3695						
3696						
3697						
3698						
3699						

::: FOLLOWING TABLE USED IN DOWNLINE LOAD ROUTINE.
 ::::: CONTAINS POINTERS TO ASCIZ DEVICE DESCRIPTIONS
 :REV B EC

3700	023220	032004	DLLIND:	.WORD	DPM
3701	023222	032007		.WORD	DUM
3702	023224	032012		.WORD	DLM
3703	023226	032015		.WORD	DQM
3704	023230	032020		.WORD	DAM
3705	023232	032023		.WORD	DUPM
3706	023234	032027		.WORD	DMCM
3707	023236	032033		.WORD	DNM
3708	023240	032036		.WORD	DLVM
3709	023242	032042		.WORD	DMPM
3710	023244	032046		.WORD	DTEM
3711	023246	032052		.WORD	DVM
3712	023250	032055		.WORD	DZM
3713	023252	032060		.WORD	UNKM
3714	023254	032070		.WORD	KDPM
3715	023256	032074		.WORD	KDZM
3716	023260	032100		.WORD	KLM
3717	023262	032103		.WORD	DMVM
3718					
3719					
3720					
3721					

```
3722 .SBTTL GLOBAL TEXT SECTION
3723
3724 :++
3725 : THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
3726 : MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
3727 : MORE THAN ONE TEST.
3728 :--
3729
3730 .SBTTL DEVICE SUPPORTED
3731 :
3732 : NAMES OF DEVICES SUPPORTED BY PROGRAM
3733 :
3734
3735
3736 023264 DEVTYP <DMP OR DMV 11>
3737 023264 L$DVTYP::
3738 023264 046504 020120 051117 .ASCIZ /DMP OR DMV 11/
3739 023272 042040 053115 030440
3740 023300 000061
3741 .EVEN
3742
3743
3744 .SBTTL PROGRAM IDENTIFICATION
3745 : TEST DESCRIPTION
3746 :
3747 023302 DESCRIPT <CZCLMBO DMP DMV-11 DATA COMM. LINK TEST>
3748 023302 L$DESC::
3749 023302 055103 046103 041115 .ASCIZ /CZCLMBO DMP DMV
3750 023310 020060 046504 020120
3751 023316 046504 026526 030461
3752 023324 042040 052101 020101
3753 023332 047503 046515 020056
3754 023340 044514 045516 052040
3755 023346 051505 000124
3756 .EVEN
3757
3758 .EVEN
3759
3760
3761
3762
```

3763
3764

.SBTTL GLOBAL FORMAT STATEMENTS, MESSAGES, AND ASCII INFO
.NLIST BEX

023352	041504	052114	000076	CLISPM:	.ASCIZ	/DCLT>/
023360	047045	040445	044477	CLIERM:	.ASCIZ	/%N%?ILL CMD-BAD SYNTAX?/
023410	047045	040445	044477	CLINUF:	.ASCIZ	/%N%?INCMPLTE CMD?/
023433	045	022516	037501	CLINBG:	.ASCIZ	/%N%?NUM TOO BIG?/
023455	045	022516	037501	CLIBRX:	.ASCIZ	/%N%?BAD RADIX?/
023475	045	022516	037501	CLIBDL:	.ASCIZ	/%N%?'LOOP' VALID ONLY IN ACTIVE?/
023537	045	022516	037501	CLINPS:	.ASCIZ	/%N%?'ECHO' VALID ONLY IN PASSIVE?/
023602	047045	040445	044477	CLIBCR:	.ASCIZ	/%N%?ILL CHR- 'A-Z,0-9,SP,TAB' ONLY?/
023647	045	022516	037501	CLISE0:	.ASCIZ	/%N%?'SIZE=0' NOT VALID?/
023700	047045	040445	052077	CLIPPE:	.ASCIZ	/%N%?TRIB CMDS ILLEGAL IN PT-PT MODE?/
023746	047045	040445	052077	CLIPW:	.ASCIZ	/%N%?TRANSMIT & EXPECT LIST MUST BE IDENTICAL FOR LOOP?/
024036	047045	040445	044124	HLP0:	.ASCIZ	/%N%THIS IS DCLT. TYPE 'H' OR '?' FOR DETAILS/
024114	047045	052045	000	HLPF:	.ASCIZ	/%N%T/
024121	104	046103	020124	HLP1:	.ASCIZ	/DCLT CMDS:/
024134	041440	042514	051101	HLP2:	.ASCII	/ CLEAR OR SHOW EXPECTLIST OR TRANSMITLIST/<15><12>
024210	050040	044522	052116		.ASCII	/ PRINT OR EXIT/<15><12>
024230	042040	046525	020120		.ASCIZ	? DUMP START-END/B?
024252	052040	044522	020102	HLP2B:	.ASCIZ	? TRIB SHOW, TRIB ESTABLISH=N/W,N(D)..OR TRIB KILL=N,ALL?
024342	005015	020040	020040	HLP2C:	.ASCIZ	<15><12>/ WHERE W=INDICATES WRITE POLL PARAMS/
024415	040	042523	020124	HLP3:	.ASCIZ	? SET EXPECTMSG OR TRANSMITMSG=TYPE/SIZE=N OR /COPY=N?
024502	051440	052105	042440	HLP3A:	.ASCIZ	/ SET EXPECT=TRANSMIT/
024527	040	020040	054524	HLP4:	.ASCIZ	? TYPE=ONES,ZEROES,1ALT,0ALT,ITEP,CCITT,ALPHA?
024606	020040	020040	020040	HLP4A:	.ASCIZ	/ OR 'OPR SPCD=A-Z,SP,TAB,0-9 IN QUOTES'/
024664	051040	047125	046440	HLP5:	.ASCIZ	? RUN MODE=MTYP/LOOP=LTYP/CHECK,STATUS,ECHO,MODEM,PASS=N?
024754	020040	046440	054524	HLP6:	.ASCII	/ MTYP=TRAN,REC,ACT,PAS,TAL,LIS,DOWN/<15><12>
025023	040	020040	052114		.ASCIZ	/ LTYP=INT,CAB,LOC,REM/
025053	116	053505	050040	EQUQ:	.ASCIZ	/NEW POLL PARAMETERS (WORD)=/
025107	116	053505	050040	EQUQ1:	.ASCIZ	/NEW POLL PARAMETERS (BYTE LOW)=/
025147	116	053505	050040	EQUQ2:	.ASCIZ	/NEW POLL PARAMETERS (BYTE HI)=/
025206	040523	042524	046114	DLLQ1:	.ASCIZ	/SATELLITE PASSWORD=/
025232	047045	051445	022465	POLPM:	.ASCIZ	/%N%\$5\$APOLL PARAMETERS FOR TRIB %D5/
025277	045	022516	043501	POLPM3:	.ASCIZ	/%N%AGLOBAL POLL PARAMETERS/
025332	050122	037124	000	CLISRP:	.ASCIZ	/RPT>/
025337	045	022516	052101	RHLP0:	.ASCIZ	/%N%ATYPE 'H' OR '?' FOR HELP!//
025375	104	046103	020124	RHLP1:	.ASCIZ	/DCLT REPORT CMDS:/
025417	040	054105	052111	RHLP2:	.ASCIZ	/ EXIT OR LOG/
025434	052040	051523	047040	RHLP3:	.ASCIZ	? TSS NNN(D)/SW OR GSS/SW?
025466	020040	020040	020040	RHLP4:	.ASCIZ	? WHERE /SW= /FULL,/ERROR,/OFFSET=NN(O)?
025542	047045	040445	052077	RPTIV:	.ASCIZ	/%N%?TRIB STATUS OFFSET=%O2%A%TOO BIG?/
025611	045	022516	046501	SHMSG:	.ASCIZ	?%N%AMSG: TYPE=%T%A/SIZE=%D3?
025645	132	051105	042517	SHTYP0:	.ASCIZ	/ZEROES/
025654	047117	051505	000	SHTYP1:	.ASCIZ	/ONES/
025661	061	046101	000124	SHTYP2:	.ASCIZ	/1ALT/
025666	040460	052114	000	SHTYP3:	.ASCIZ	/0ALT/
025673	103	044503	052124	SHTYP4:	.ASCIZ	/CCITT/
025701	111	042524	000120	SHTYP5:	.ASCIZ	/ITEP/
025706	046101	044120	000101	SHTYP6:	.ASCIZ	/ALPHA/
025714	050117	020122	050123	SHTYP7:	.ASCIZ	/OPR SPEC/
025725	045	022516	052101	SHTRE:	.ASCIZ	\%N%TRIB ADDRESS LIST IS EMPTY\
025764	047045	040445	051124	SHTRH:	.ASCIZ	\%N%TRIB ADDRESS LIST:%N\
026015	045	031504	040445	SHTAP:	.ASCIZ	\%D3%A, \
026025	045	022516	052101	SHTFL:	.ASCIZ	\%N%TRIB ADDRESS LIST FULL - ADDRESS= %Z3%A NOT ADDED\
026113	045	022516	037501	SHTUN:	.ASCIZ	\%N%?TRIB ADDRESS= %Z3%A IS NOT UNIQUE?\

```

026163 045 022516 037501 SHTNF: .ASCIZ \N%?TRIB ADDRESS= %Z3%A NOT FOUND?\
026227 045 022516 037501 SHTLP: .ASCIZ \N%?CABLE,LOC,REM LOOP NOT VALID IN 'MULTIPT MODE'?\
026314 047045 040445 052077 SHTLPA: .ASCIZ /N%?TRIBS MUST BE ESTABLISHED TO EXECUTE?/
026367 045 022516 037501 SHTLPB: .ASCIZ /N%?TRIB STATION CANNOT DO LOOP?/
026431 045 022516 037501 SHTLPC: .ASCIZ /N%?ONLY ONE TRIB (TRIB ADDR 1) ALLOWED/
026502 047045 051445 022465 SHTLPD: .ASCIZ /N%S5%AFOR LOOP IN MULTIPOINT?/
026541 045 022516 037501 SHTIV: .ASCIZ \N%?TRIB ADDRESS= %Z3%A INVALID?\
026603 045 022516 051101 SHTBR: .ASCIZ /N%ARX BUFFER NOT BIG ENOUGHN%ATOO MANY TRIBS OR MSGS/
026672 042522 042503 053111 MO0: .ASCIZ /RECEIVE/
026702 051124 047101 046523 MO1: .ASCIZ /TRANSMIT/
026713 120 051501 044523 MO2: .ASCIZ /PASSIVE/
026723 101 052103 053111 MO3: .ASCIZ /ACTIVE/
026732 047504 047127 044514 MO4: .ASCIZ /DOWNLINELOAD/
026747 124 046101 000113 MO5: .ASCIZ /TALK/
026754 044514 052123 047105 MO6: .ASCIZ /LISTEN/
026763 000 LP0: .ASCIZ //
026764 046057 047517 036520 LP00: .ASCIZ ?/LOOP=?
026773 111 052116 051105 LP1: .ASCIZ ?INTERNAL?
027004 040503 046102 000105 LP2: .ASCIZ ?CABLE?
027012 047514 040503 046514 LP3: .ASCIZ ?LOCALMODEM?
027025 122 046505 052117 LP4: .ASCIZ ?REMOTEMODEM?
027041 116 117 PNST: .ASCII /NO/
027043 123 040524 052524 PST: .ASCIZ /STATUS/
027052 047516 PNCK: .ASCII /NO/
027054 044103 041505 000113 PCK: .ASCIZ /CHECK/
027062 047516 PNEC: .ASCII /NO/
027064 041505 047510 000 PEC: .ASCIZ /ECHO/
027071 116 117 PNMS: .ASCII /NO/
027073 115 042117 046505 PMS: .ASCIZ /MODEM/

027101 045 022516 046101 LISP: .ASCIZ /N%ALIS>/
027112 046124 037113 000 OPRMM: .ASCIZ /TLK>/
027117 124 044510 020123 L5060: .ASCIZ /THIS A 50. OR 60. HZ. LSI-11:/
027156 .EVEN

```

:
: FORMAT STATEMENTS USED IN PRINT CALLS
:

```

027156 047045 040445 047504 DLLCM: .ASCIZ /N%ADOWN LINE LOAD COMPLETED SUCCESSFULLY/

027230 047045 040445 046103 BDCLK: .ASCIZ /N%ACLOCK NOT FOUND/
027254 047045 040445 040502 NOCLK: .ASCIZ /N%ABAD CLOCK - PROGRAM WILL HANG ON 'TIMEOUT'!!!/
027335 115 054101 020056 TABEX: .ASCIZ /MAX. CHAR. MSG COUNT EXCEEDED -/
027375 102 043125 042506 BUFEX: .ASCIZ /BUFFER FULL -/
027413 045 022516 022524 MSGTRN: .ASCIZ /N%T%A MSG. NOT BUILT !!!/
027444 047045 040445 044103 MSGTRU: .ASCIZ /N%ACHAR. COUNT EXCEEDS BUFF LIMIT - MSG TRUNCATED/
027527 045 022516 032523 SHFO: .ASCIZ ?N%S5%AMODE=%T%T%T%A/PASS=%Z5?

027565 045 022516 032523 SHF1: .ASCIZ ?N%S5%S5%S5%A/%T%A/%T%A/%T%A/%T?
027625 045 032523 040445 EFM2: .ASCIZ /S5%ATOTAL MISMATCHES IN MSG = %D5/
027670 047045 051445 022463 PCPM: .ASCIZ /N%S3%ACALLED FROM PC=%06/
027722 051445 022465 041501 EFM11: .ASCIZ /S5%ACOMPARE COUNT=%D5%S3%ARECEIVE COUNT=%D5/

```


031564

.EVEN

;EXECUTION STATUS MESSAGES TO BE PRINTED TO KEEP OPERATOR AWAKE

031564	047045	000		CR:	.ASCIZ	/%N/	;CR FOR LINES IN A ROW
031567	045	031523	040445	STXQ:	.ASCIZ	/%S3%ATXQ/	;ABOUT TO TRANSMIT
031600	051445	022463	052101	STXC:	.ASCIZ	/%S3%ATXC/	;TX COMPLETED
031611	045	031523	040445	SRXQ:	.ASCIZ	/%S3%ARXQ/	;ABOUT TO RECEIVE
031622	051445	022463	042501	SDVE:	.ASCIZ	/%S3%AERR/	;DEVICE ERROR
031633	045	031523	040445	SCM:	.ASCIZ	/%S3%ACMP/	;ABOUT TO DO DATA CHECKING OF RECVD VS. EXPTD
031644	051445	022463	044501	SDVI:	.ASCIZ	/%S3%AINI/	;DEVICE ABOUT TO BE INITIALIZED
031655	045	031523	040445	SCML:	.ASCIZ	/%S3%ACML/	;COMPARE LENGTH ERROR
031666	051445	022463	041501	SCMD:	.ASCIZ	/%S3%ACMD/	;COMPARE DATA ERROR
031677	045	031523	040445	SEOP:	.ASCIZ	/%S3%AEOP/	;END OF PASS
031710	051445	022463	046501	SMSC:	.ASCIZ	/%S3%AMSC/	;MODEM STATUS CHANGE
	031722						.EVEN

;REV B BY EC

;NEXT ASCII LINES ARE USED IN SATELLITE ID MESSAGES

031722	047045	040445	042523	SECRM:	.ASCIZ	/%N%ASECONDARY BOOT REQ FROM %T%A DEVICE-TYPE= %D3/
032004	050104	000		DPM:	.ASCIZ	/DP/
032007	104	000125		DUM:	.ASCIZ	/DU/
032012	046104	000		DLM:	.ASCIZ	/DL/
032015	104	000121		DQM:	.ASCIZ	/DQ/
032020	040504	000		DAM:	.ASCIZ	/DA/
032023	104	050125	000	DUPM:	.ASCIZ	/DUP/
032027	104	041515	000	DMCM:	.ASCIZ	/DMC/
032033	104	000116		DNM:	.ASCIZ	/DN/
032036	046104	000126		DLVM:	.ASCIZ	/DLV/
032042	046504	000120		DMPM:	.ASCIZ	/DMP/
032046	052104	000105		DTEM:	.ASCIZ	/DTE/
032052	053104	000		DVM:	.ASCIZ	/DV/
032055	104	000132		DZM:	.ASCIZ	/DZ/
032060	047125	047113	053517	UNKM:	.ASCIZ	/UNKNOWN/
032070	042113	000120		KDPM:	.ASCIZ	/KDP/
032074	042113	000132		KDZM:	.ASCIZ	/KDZ/
032100	046113	000		KLM:	.ASCIZ	/KL/
032103	104	053115	000	DMVM:	.ASCIZ	/DMV/
	032110					.EVEN

.SBTTL

ASCII FORMATS FOR TSS AND GSS SLOTS

Address	Offset	Value	Code	Format
032110	047045	047445	022466	TSS0A: .ASCIZ /%N%06%S2%ATTRIB STATUS FLAGS/
032144	047045	047445	022463	TSS1A: .ASCIZ /%N%03%S5%ANAK REASON%N%03%S5%ATTRIB ADDR/
032214	047045	047445	022466	TSS2A: .ASCIZ /%N%06%S2%APOLL STATUS FLAGS/
032250	047045	047445	022463	TSS3A: .ASCIZ /%N%03%S5%APOLL RATE%N%03%S5%APOLL PRIORITY/
032323	045	022516	031517	TSS4A: .ASCIZ /%N%03%S5%ANAZ%N%03%S5%AMAX MSG COUNTER/
032371	045	022516	031517	TSS5A: .ASCIZ /%N%03%S5%ACOMM POOL QUOTA%N%03%S5%ARX THRESH ERRS/
032453	045	022516	031517	TSS6A: .ASCIZ /%N%03%S5%ATX THRESH. ERRS %N%03%S5%ASELECT THRESH. ERRS/
032543	045	022516	033117	TSS7A: .ASCIZ /%N%06%S2%ADATA MSGS. TX'MITTD/
032601	045	022516	033117	TSS10A: .ASCIZ /%N%06%S2%ADATA MSGS. RX'CVD/
032635	045	022516	033117	TSS11A: .ASCIZ /%N%06%S2%ASELECTION INTERVALS/
032673	045	022516	031517	TSS12A: .ASCIZ /%N%03%S5%ADATA ERRORS OUT%N%03%S5%AHBCC %01%A BCC %01%A REP %01/
032770	047045	047445	022463	TSS13A: .ASCIZ /%N%03%S5%ADATA ERRORS IN%N%03%S5%AHBCC %01%A BCC %01%A REP %01/
033064	047045	047445	022463	TSS14A: .ASCIZ /%N%03%S5%ALOCAL BUFFER ERRS%N%03%S5%ATU %01%A TS %01/
033147	045	022516	031517	TSS15A: .ASCIZ /%N%03%S5%AREMOTE BUFFER ERRS%N%03%S5%ATU %01%A TS %01/
033233	045	022516	031517	TSS16A: .ASCIZ /%N%03%S5%ASELECTION T-0%N%03%S5%ANRTS %01%A IRTS %01/
033316	047045	047445	022463	TSS17A: .ASCIZ /%N%03%S5%ALOCAL REPLY T-0%N%03%S5%AREMOTE REPLY T-0/
033402	047045	047445	022463	TSS20A: .ASCIZ /%N%03%S5%AHIGHEST MSG # TX'D%N%03%S5%AHIGHEST MSG # ACK'D/
033474	047045	047445	022463	TSS21A: .ASCIZ /%N%03%S5%ANEXT MSG # TO TX%N%03%S5%ATPTR ADDR OF LKBNK/
033563	045	022516	031517	TSS22A: .ASCIZ /%N%03%S5%ALAST MSG # TX'D%N%03%S5%AXPTR ADDR OF LKBNK/
033651	045	022516	031517	TSS23A: .ASCIZ /%N%03%S5%ACTL X REPLY T-0%N%03%S5%ASTRT OF TX BUFF Q/
033736	047045	047445	022463	TSS24A: .ASCIZ /%N%03%S5%AEND OF TX BUFF Q%N%03%S5%AHIGHEST MSG # RX'D/
034025	045	022516	031517	TSS25A: .ASCIZ /%N%03%S5%ASTRT OF RX BUFF Q%N%03%S5%AEND OF RX BUFF Q/
034112	047045	047445	022466	TSS26A: .ASCIZ /%N%06%S2%ATX DELAY TIMER/
034143	045	022516	031517	TSS27A: .ASCIZ /%N%03%S5%ANO DATA MSG COUNTER%N%03%S5%AT-O COUNTER/
034226	047045	047445	022466	TSS30A: .ASCIZ /%N%06%S2%A/
034240	051120	051505	052105	TS30AA: .ASCIZ /PRESET VALUE FOR TX DELAY TIMER/
034300	047045	047445	022463	TSS31A: .ASCIZ /%N%03%S5%AAQ VAL FOR ACT%N%03%S5%AAQ VAL FOR ACT/
034357	045	022516	031517	TSS32A: .ASCIZ /%N%03%S5%AAQ VAL FOR INACT%N%03%S5%AAQ VAL FOR INACT/
034442	047045	047445	022463	TSS33A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
034525	045	022516	031517	TSS34A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
034604	047045	047445	022463	TSS35A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
034663	045	022516	033117	TSS36A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
034735	045	022516	033117	TSS37A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035002	047045	047445	022463	GSS0A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035043	045	022516	031517	GSS1A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035101	045	022516	031517	GSS2A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035140	047045	047445	022463	GSS3A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035175	045	022516	031517	GSS4A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035232	047045	047445	022463	GSS5A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035267	045	022516	031517	GSS6A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035345	045	022516	031517	GSS7A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035423	045	022516	031517	GSS10A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035475	045	022516	033117	GSS11A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035515	045	022516	031517	GSS12A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035553	045	022516	031517	GSS13A: .ASCIZ /%N%03%S5%AAQ VAL FOR UNRSP%N%03%S5%AAQ VAL FOR UNRSP/
035614	047045	047445	022466	GSS14A: .ASCIZ /%N%06%S2%ARTNADD/
035635	045	022516	031517	GSS15A: .ASCIZ /%N%03%S5%AREMOTE STA ERRS%N%03%S5%AAOVRN %01%A MHFE %01%A SEL %01%A STR %01
035745	045	022516	031517	GSS16A: .ASCIZ /%N%03%S5%ALOCAL STA ERRS%N%03%S5%AAOVRN %01%A MHFE %01%A UNDR %01%A OVR %01
036055	045	022516	031517	GSS17A: .ASCIZ /%N%03%S5%AGBL HDR BCC%N%03%S5%AAOVRN %01%A MHFE %01%A SEL %01%A STR %01
036137	045	022516	031517	GSS20A: .ASCIZ /%N%03%S5%AGBL HDR BCC%N%03%S5%AAOVRN %01%A MHFE %01%A SEL %01%A STR %01
036204	047045	047445	022463	GSS21A: .ASCIZ /%N%03%S5%AGBL HDR BCC%N%03%S5%AAOVRN %01%A MHFE %01%A SEL %01%A STR %01
036251	045	022516	031517	GSS22A: .ASCIZ /%N%03%S5%AGBL HDR BCC%N%03%S5%AAOVRN %01%A MHFE %01%A SEL %01%A STR %01
036316	047045	047445	022463	GSS23A: .ASCIZ /%N%03%S5%AGBL HDR BCC%N%03%S5%AAOVRN %01%A MHFE %01%A SEL %01%A STR %01

036363	045	022516	031517	GSS24A:	.ASCIZ	/N%03%S5%ARX HDR 3%N%03%S5%ARX HDR 4/
036430	047045	047445	022463	GSS25A:	.ASCIZ	/N%03%S5%ARX HDR 5%N%03%S5%ARX HDR 6/
036475	045	022516	033117	GSS26A:	.ASCIZ	/N%06%S2%AR TIMER/
036517	045	022516	033117	GSS27A:	.ASCIZ	/N%06%S2%AD TIMER/
036541	045	022516	033117	GSS30A:	.ASCIZ	/N%06%S2%APOLL DELAY TIMER/
036574	047045	047445	022463	GSS31A:	.ASCIZ	/N%03%S5%APOLL UPDATE PTR%N%03%S5%ADEAD SCAN/
036651	045	022516	031517	GSS32A:	.ASCIZ	/N%03%S5%ACARRIER LOSG TIM%N%03%S5%AUSART HANG CTR/
036734	047045	047445	022466	GSS33A:	.ASCIZ	/N%06%S2%ANUM SYNC/
036757	045	022516	033117	GSS34A:	.ASCIZ	/N%06%S2%ACARRIER WAIT TIMER COUNTER/
037024	047045	047445	022466	GSS35A:	.ASCIZ	/N%06%S2%ADELTA T/
037046	047045	047445	022466	GSS36A:	.ASCIZ	/N%06%S2%ADEAD T/
037067	045	022516	033117	GSS37A:	.ASCIZ	/N%06%S2%APOLL DELAY/

:DEVICE ERROR MESSAGES

037114	042504	044526	042503	DVEM0:	.ASCII	/DEVICE DID NOT RETURN RUN BIT/
037151	015	020012	020040		.ASCIZ	<15><12>/ SEL0 SEL2/
037175	106	044501	052514	DVEM1:	.ASCII	/FAILURE IN MICRO DIAGNOSTICS/
037231	015	020012	020040		.ASCIZ	<15><12>/ SEL0 SEL6/
037255	124	046511	020105	DVEM2:	.ASCII	/TIME OUT WAITING FOR TX OR RX TO COMPLETE/
037326	005015	020040	020040		.ASCIZ	<15><12>/ SEL0 SEL2/
037352	044524	042515	047440	DVEM3:	.ASCII	/TIME OUT WAITING FOR RDI/
037402	005015	020040	020040		.ASCIZ	<15><12>/ SEL0 SEL2/
037426	047503	052116	047522	DVEM4:	.ASCII	/CONTROL OR INFORMATION OUT ERROR/
037466	005015	020040	020040		.ASCIZ	<15><12>/ SEL2 SEL6/
037512	046111	042514	040507	DVEM5:	.ASCII	/ILLEGAL TRANSMIT COMPLETE/
037543	015	020012	020040		.ASCIZ	<15><12>/ SEL4 SEL6/
037567	111	046114	043505	DVEM6:	.ASCII	/ILLEGAL RECEIVE COMPLETE/
037617	015	020012	020040		.ASCIZ	<15><12>/ SEL4 SEL6/
037643	121	042525	047440	DVEM7:	.ASCII	/QUE OVERFLOW BUFFER COMPLETE/
037677	015	020012	020040		.ASCIZ	<15><12>/ SEL4 SEL6/
037723	122	042114	047440	DVEM8:	.ASCII	/RLD OR MOD: ENABLE OF PASSWORD SW NOT SET/
037774	005015	020040	020040		.ASCIZ	<15><12>/ SEL0 SEL2/
040020	042040	053517	020116	DLLAB:	.ASCII	/ DOWN LINE LOAD ABORTED/
040047	015	020012	020040		.ASCIZ	<15><12>/ RXBUF TXBUF /
040074	042523	042514	052103	SLTHEM:	.ASCIZ	/SELECT THRESHOLD/
040115	123	040524	052122	STRCM:	.ASCIZ	/START RXD IN RUN/
040136	040515	047111	020124	MARM:	.ASCIZ	/MAINT RXD IN RUN/
040157	115	044501	052116	MARHM:	.ASCIZ	/MAINT RXD IN HALT/
040201	123	040524	052122	STRMM:	.ASCIZ	/START RXD IN MAINT/
040224	044522	043516	042040	RIM:	.ASCIZ	/RING DECTECTED/
040243	104	040505	020104	DEADTM:	.ASCIZ	/DEAD TRIB/
040255	122	047125	051440	RUSM:	.ASCIZ	/RUN STATE ERR/
040273	102	041101	044514	BABTM:	.ASCIZ	/BABLING TRIB/
040310	052123	042522	046501	STREAM:	.ASCIZ	/STREAMING TRIB/

040327	116	020117	047515	PE100M:	.ASCIZ	/NO MODE DEF/
040343	111	046114	043505	PE102M:	.ASCIZ	/ILLEGAL TYPE CODE/
040365	111	046114	043505	PE104M:	.ASCIZ	/ILLEGAL MODE CHANGE/
040411	103	047117	051124	PE106M:	.ASCIZ	/CONTROL IN TO UNES. TRIB/
040442	047503	046515	047101	PE110M:	.ASCIZ	/COMMAND TO TRIB 0/
040464	047503	046515	047101	PE112M:	.ASCIZ	/COMMAND TO UNHALTED TRIB/
040515	115	054101	052040	PE114M:	.ASCIZ	/MAX TRIBS EXCEEDED/
040540	051505	041124	052040	PE116M:	.ASCIZ	/ESTB TO ALREADY ESTABLISHED/
040574	046111	042514	043501	PE120M:	.ASCIZ	/ILLEGAL REQUEST KEY/
040620	051501	044523	047107	PE122M:	.ASCIZ	/ASSIGN BUFF UNEST. TRIB/
040650	051501	044523	047107	PE124M:	.ASCIZ	/ASSIGN BUFF HALTD TRIB/
040677	101	051523	043511	PE126M:	.ASCIZ	/ASSIGN BUFF BYTE CNT 0/
040726	051501	044523	047107	PE130M:	.ASCIZ	/ASSIGN TX BUFF TRIB 0/
040754	020122	051117	053440	PE132M:	.ASCIZ	/R OR W RESERVED TSS/
041000	051525	020105	042522	PE134M:	.ASCIZ	/USE RESERVED BIT IN BSEL7/
041032	047503	046515	047117	PE136M:	.ASCIZ	/COMMON POOL ERROR/
041054	052521	052117	020101	PE140M:	.ASCIZ	/QUOTA OVERFLOW/
041073				TXTHEM:		
041073				RXTHEM:		
041073	123	040520	042522	PE142M:	.ASCIZ	/SPARE/
041101	123	040520	042522	PE144M:	.ASCIZ	/SPARE/
041107	102	043125	042506	BUFTSM:	.ASCIZ	/BUFFER TOO SMALL/
041130	047516	020116	054105	NOEXM:	.ASCIZ	/NON EXIST MEM/
041146	044504	041523	047117	DISCON:	.ASCIZ	/DISCONNECT/
041160	052521	052505	020105	QUEOM:	.ASCIZ	/QUEUE OVER/
041173	103	051101	044522	CARLOS:	.ASCIZ	/CARRIER LOSS/
041210	047111	047506	046522	INFOM:	.ASCIZ	/INFORMATION OUT/
041230	054124	047040	052117	TXNC:	.ASCIZ	/TX NOT COMPLETE/
041250	054122	047040	052117	RXNC:	.ASCIZ	/RX NOT COMPLETE/
041270	042523	020103	042522	RXM1:	.ASCIZ	/SEC REQ ERR WORD 1/
041313	123	041505	051040	RXM2:	.ASCIZ	/SEC REQ ERR WORD 2/

.LIST .EVEN
BEX

3765
3766
3767
3768
3769
3770
3771
3772

3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828

.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

PRINTB #EVTF5A,OFSET,<B,GOOD>,<B,BAD>

ERR1::

;INDIVIDUAL DATA COMPARE ERROR

CLR -(SP)
BISB BAD,(SP)
CLR -(SP)
BISB GOOD,(SP)
MOV OFSET,-(SP)
MOV #EVTF5A,-(SP)
MOV #4,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #12,SP

ENDMSG

L10001:

TRAP C\$MSG

BGNMSG ERR2

PRINTB #EFM2,TEMP4

ERR2::

;TOTAL DATA COMPARE FAILS ERROR

MOV TEMP4,-(SP)
MOV #EFM2,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #6,SP

ENDMSG

L10002:

TRAP C\$MSG

BGNMSG ERR10

PRINTB #EFM11,R4,TEMP3

ERR10::

;LENGTH COMPARISON ERROR

MOV TEMP3,-(SP)
MOV R4,-(SP)
MOV #EFM11,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

ENDMSG

L10003:

TRAP C\$MSG

3829							
3830							
3831				:	PRINT THE 2 OCTAL #'S IN TEMP3/4		
3832				:			
3833				:			
3834	041456			BGNMSG	ERR13		
3835	041456					ERR13::	
3836	041456			PRINTB	#EVTF3C,TEMP3,TEMP4		
3837	041456	013746	017360			MOV	TEMP4,-(SP)
3838	041462	013746	017356			MOV	TEMP3,-(SP)
3839	041466	012746	030656			MOV	#EVTF3C,-(SP)
3840	041472	012746	000003			MOV	#3,-(SP)
3841	041476	010600				MOV	SP,R0
3842	041500	104414				TRAP	C\$PNTB
3843	041502	062706	000010			ADD	#10,SP
3844	041506			ENDMSG			
3845	041506					L10004:	
3846	041506	104423				TRAP	C\$MESSG
3847							
3848				:	PRINT THE 2 OCTAL #'S IN TEMP3/4		
3849				:	AND THE MMSG. WHOSE ADDR. IS IN CONOTM		
3850				:			
3851				:			
3852				:			
3853	041510			BGNMSG	ERR14		
3854	041510					ERR14::	
3855	041510			PRINTB	#EVTF3D,TEMP3,TEMP4,CONOTM		
3856	041510	013746	017366			MOV	CONOTM,-(SP)
3857	041514	013746	017360			MOV	TEMP4,-(SP)
3858	041520	013746	017356			MOV	TEMP3,-(SP)
3859	041524	012746	030673			MOV	#EVTF3D,-(SP)
3860	041530	012746	000004			MOV	#4,-(SP)
3861	041534	010600				MOV	SP,R0
3862	041536	104414				TRAP	C\$PNTB
3863	041540	062706	000012			ADD	#12,SP
3864	041544			ENDMSG			
3865	041544					L10005:	
3866	041544	104423				TRAP	C\$MESSG
3867							
3868							
3869	041546			BGNMSG	ERR15		
3870	041546					ERR15::	
3871	041546			PRINTB	#EVTF3F,RSEL4,RSEL6,<B,RSEL3>		
3872	041546	005046				CLR	-(SP)
3873	041550	153716	066526			BISB	RSEL3,(SP)
3874	041554	013746	066524			MOV	RSEL6,-(SP)
3875	041560	013746	066522			MOV	RSEL4,-(SP)
3876	041564	012746	030715			MOV	#EVTF3F,-(SP)
3877	041570	012746	000004			MOV	#4,-(SP)
3878	041574	010600				MOV	SP,R0
3879	041576	104414				TRAP	C\$PNTB
3880	041600	062706	000012			ADD	#12,SP
3881	041604			ENDMSG			
3882	041604					L10006:	
3883	041604	104423				TRAP	C\$MESSG
3884							

3885 041606
3886 041606
3887 041606
3888 041606 005046
3889 041610 153716 066520
3890 041614 013746 066514
3891 041620 013746 066516
3892 041624 012746 030715
3893 041630 012746 000004
3894 041634 010600
3895 041636 104414
3896 041640 062706 000012
3897 041644
3898 041644
3899 041644 104423
3900
3901 041646
3902 041646 000167
3903 041650 177772
3904

BGNMSG ERR16
PRINTB #EVTF3F,TSEL4,TSEL6,<B,TSEL3>

ENDMSG

EXIT MSG

ERR16::

CLR -(SP)
BISB TSEL3,(SP)
MOV TSEL6,-(SP)
MOV TSEL4,-(SP)
MOV #EVTF3F,-(SP)
MOV #4,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #12,SP

L10007:

TRAP C\$MSG

.WORD JS\$JMP
.WORD L10007-2-

3905
3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951

```
.SBTTL GLOBAL SUBROUTINES SECTION

:++
: THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
: THAT ARE USED IN MORE THAN ONE TEST.
:--

.SBTTL          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, THIS 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

: IMPLICIT INPUTS:
: THE SUPERVISOR SPACE WHERE CLOCK INFO WAS RETURNED BY THE "CLOCK" CALL

: OUTPUTS:
: "CLKCSR" GETS LOADED WITH THE CLOCK'S 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. (HERTZ RATE) WHICH DETERMINES
: THE NUMBER OF TICKS IN A SECOND

: CALLING SEQUENCE:
: JSR      PC,CLKSET          ;CALL CLOCK SETUP WITH R1 & R2 SETUP
:--

CLKSET:
MOV      (R1)+,(R2)+          ;LOAD CLOCK'S CSR ADDR. INTO "CLKCSR"
MOV      (R1)+,(R2)           ;LOAD CLOCK'S INT. LEVEL INTO "CLKBR"
ASL      (R2)                 ;ADJUST THE INT. LEVEL FOR LOADING INTO
; THE PSW WITH A "SETVEC" CALL
ASL      (R2)
ASL      (R2)
ASL      (R2)
ASL      (R2)+
MOV      (R1)+,(R2)+          ;LOAD CLOCK'S INT. VECTOR INTO "CLKVEC"
MOV      (R1)+,(R2)+          ;LOAD CLOCK'S HERTZ RATE INTO "CLKHZ"
RTS      PC
```

041652 012122
 041654 012112
 041656 006312
 041660 006312
 041662 006312
 041664 006312
 041666 006322
 041670 012122
 041672 012122
 041674 000207

3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000
4001
4002
4003
4004
4005
4006
4007

```

.SBTTL          CLOCK INTERRUPT SERVICE ROUTINE
:++
:FUNCTIONAL DESCRIPTION:
:THIS IS THE CLOCK INTERRUPT SERVICE ROUTINE WHICH TAKES CARE OF
:KEEPING THE 'TIME-SINCE-START' AND COUNTING DOWN ANY OF THE
:'EVENT' TIMERS. THE TIMERS ARE USED TO TIME COMPLETION OF DEVICE
:REQUESTS. THE 'TIME-SINCE-START' IS USED TO BE LOGGED WITH EACH ENTRY
:INTO THE EVENT LOG.
:
:IMPLICIT INPUTS:
:TIMTCK: THE CURRENT NO. OF TICKS LEFT TO BE COUNTED UNTIL A SECOND
:HAS BEEN COUNTED OFF
:CLKHZ: THE NO. OF TICKS IN A SECOND, DETERMINED BY THE SYS. LINE FREQ.
:TIMMIN & TIMSEC: CURRENT VALUE OF 'TIME-SINCE-START'
:                IN MINUTES & SECONDS
:TIMER 1,2, & S: CURRENT VALUES OF THE 'EVENT TIMERS'
:
:IMPLICIT OUTPUTS:
:NEW VALUE OF EVENT TIMER '1' DECREMENTED BY 1 TICK IF IT WAS NON-ZERO
:NEW VALUE OF EVENT TIMER '2' DECREMENTED BY 1 TICK IF IT WAS NON-ZERO
:NEW VALUE OF EVENT TIMER 'S' DECREMENTED BY 1 SECOND IF IT WAS NON-ZERO
:
:FUNCTIONAL SIDE EFFECTS:
:THE CLOCK IS DISABLED UPON ENTRY AND REENABLED WHEN LEAVING
:
:CALLING SEQUENCE:
:THIS ROUTINE IS CALLED WHEN THE CLOCK INTERRUPTS THRU 'CLKVEC'.
:THE ADDRESS OF THIS ROUTINE WAS LOADED INTO THE CLOCK'S INTERRUPT
:VECTOR WITH A SUPERVISOR 'SETVEC' CALL.
:--
    
```

```

041676
041676
041676 005077 155532
041702 005337 017452
041706 001015
041710 013737 017442 017452
041716 005237 017450
041722 022737 000074 017450
041730 001004
041732 005237 017446
041736 005037 017450
041742 005737 017454
041746 001402
041750 005337 017454
041754 005737 017456
041760 001402
041762 005337 017456
041766 005737 017460
041772 001406
041774 023737 017442 017452
042002 001002
042004 005337 017460
    
```

```

BGNSRV CLKINT          CLKINT::
CLR @CLKCSR           ;DISABLE THE CLOCK FORM INTERRUPTING
DEC TIMTCK            ;DECREMENT THE # OF TICKS/SEC.
BNE 1$                ;GO CHECK TIMERS (1&2-TICKS, 3-SECONDS)
MOV CLKHZ,TIMTCK      ;RESET THE # OF TICKS/SEC.
INC TIMSEC            ;INC # OF SECS-SINCE-START
CMP #60.,TIMSEC       ;SEE IF WE'VE COUNTED 60 SECS. YET
BNE 1$                ;IF NOT, GO CHECK TIMERS
INC TIMMIN            ; ELSE INC MINUTES-SINCE-START
CLR TIMSEC            ; AND RESTART SECOND COUNTER
:
1$: TST TIMER1        ;SEE IF TIMER #1, TIMING ANYTHING
BEQ 2$                ; IF=0, NOTHING BEING TIMED CHECK NEXT TIMER
DEC TIMER1            ; ELSE DECREMENT THE TIMER VALUE (BY 1 TICK)
:
2$: TST TIMER2        ;SEE IF TIMER #2, TIMING ANYTHING
BEQ 3$                ; IF=0, NOTHING BEING TIMED CHECK NEXT TIMER
DEC TIMER2            ; ELSE DECREMENT THE TIMER VALUE (BY 1 TICK)
:
3$: TST TIMERS        ;SEE IF TIMER #3, TIMING ANYTHING
BEQ 4$                ; IF=0, NOTHING BEING TIMED, LEAVE
CMP CLKHZ,TIMTCK      ;SEE IF A SECOND HAS BEEN COUNTED OFF
BNE 4$                ; BR IF NO
DEC TIMERS            ; ELSE DECREMENT THE TIMER VALUE (BY 1 SEC.)
    
```

```
4008 042010 013777 017444 155416 4$:  MOV   CLKEN,@CLKCSR  ;REENABLE THE CLOCK TO INTERRUPT
4009 042016                                     ENDSRV
4010 042016                                     L10010:
4011 042016 000002                                     RTI
```

```

4012 .SBTTL          EVENT LOG SUBROUTINES
4013
4014 :++
4015 : FUNCTIONAL DESCRIPTION:
4016 : THIS SUBROUTINE HAS A DIFFERENT ENTRY POINT
4017 : FOR EACH EVENT TO BE LOGGED AND ALWAYS PRINTS
4018 : THE SHORT 'OPERATOR AWAKE' MESSAGE TO CONSOLE THEN LOGS THE
4019 : EVENT TYPE, TIME, AND THE OTHER 3 WORDS OF INFO PASSED TO THE
4020 : SUBROUTINE AT CALLING TIME
4021
4022 : INPUTS:
4023 : TIMMIN & TIMSEC:          CURRENT VALUE OF 'TIME-SINCE-START'
4024 : TEMP2: WORD #1 OF EVENT LOG INFORMATION (FOR MOST EVENT TYPES)
4025 : TEMP3: WORD #2 OF EVENT LOG INFORMATION
4026 : TEMP4: WORD #3 OF EVENT LOG INFORMATION
4027 : MODS:  CURRENT VALUE OF THE MODEM SIGNALS AVAILABLE FROM THE DEVICE
4028
4029 : OUTPUTS:
4030 : 'OPERATOR AWAKE' MESSAGE SENT TO THE CONSOLE
4031 : NEW EVENT LOGGED IN 'EVTLOG' (EVENT LOG)
4032 : UPDATED 'EVTPTN' (EVENT LOG ENTRY POINTER)
4033
4034 : SUBORDINATE ROUTINES USED:
4035 : 'DVMODS' THE DEVICE SUBROUTINE THAT RETURNS MODEM STATUS IN 'MODS'
4036 : (FOR SOME EVENT TYPES)
4037
4038 : FUNCTIONAL SIDE EFFECTS:
4039 : TEMP:  USED TO STORE ADDRESS OF 'OPERATOR AWAKE' MESSAGE
4040 : TEMP1: USED TO SETUP THE VALUE OF THE 'EVENT TYPE' BYTE FOR LOGGING
4041
4042 : CALLING SEQUENCE:
4043 : JSR    PC,LOGTXQ          ;CALL THE LOG EVENT SUBROUTINE WITH TEMP,TEMP1,
4044 : ..    .. ..             ; TEMP2, TEMP3, AND TEMP4 SETUP
4045 : ..    .. ..
4046 : JSR    PC,LOGCMP
4047 : ---
4048
4049 042020 LOGTXQ:  MOV    #STXQ,TEMP1      ;SET UP MSG. TO PRINT
4050 042020 012737 031567 017352  MOV    #TXQ,TEMP      ;SET UP EVENT TYPE
4051 042026 012737 000000 017350  BR     LOGS1         ;GO LOG EVENT AND TIME
4052 042034 000522
4053
4054 042036 LOGTXC:  MOV    #STXC,TEMP1      ;SET UP MSG. TO PRINT
4055 042036 012737 031600 017352  MOV    #TXC,TEMP      ;SET UP EVENT TYPE
4056 042044 012737 000002 017350  BR     LOGS1         ;GO LOG EVENT AND TIME
4057 042052 000513
4058
4059 042054 LOGRXQ:  MOV    #SRXQ,TEMP1      ;SET UP MSG. TO PRINT
4060 042054 012737 031611 017352  MOV    #RXQ,TEMP      ;SET UP EVENT TYPE
4061 042062 012737 000004 017350  BR     LOGS1         ;GO LOG EVENT AND TIME
4062 042070 000504
4063
4064 042072 LOGRXC:  MOV    #RXC,TEMP      ;SET UP EVENT TYPE
4065 042072 012737 000006 017350  BR     LOGS1         ;GO LOG EVENT AND TIME
4066 042100 000500
4067 042102 LGDVE:

```

```

4068 042102 012737 031622 017352      MOV      #SDVE,TEMP1      ;SET UP MSG. TO PRINT
4069 042110 012737 000010 017350      MOV      #DER,TEMP        ;SET UP EVENT TYPE
4070 042116 000511                      BR        LOGS3           ;GO LOG EVENT AND TIME
4071
4072 042120                      LOGDVI:
4073 042120 012737 031644 017352      MOV      #SDVI,TEMP1      ;SET UP MSG. TO PRINT
4074 042126 012737 000012 017350      MOV      #DVI,TEMP        ;SET UP EVENT TYPE
4075 042134 113737 017402 017354      MOV      MODTYP,TEMP2
4076 042142 113737 017404 017355      MOV      MLTYP,TEMP2+1
4077 042150 013737 017412 017356      MOV      RPASS,TEMP3
4078 042156 013737 017410 017360      MOV      PARAM,TEMP4      ;SET UP EVNT ENTRIES
4079 042164 000466                      BR        LOGS3           ;GO LOG EVENT AND TIME
4080
4081 042166                      LOGCMP:
4082 042166 012737 031633 017352      MOV      #SCM,TEMP1      ;SET UP MSG. TO PRINT
4083 042174 012737 000014 017350      MOV      #DCK,TEMP        ;SET UP EVENT TYPE
4084 042202 000415                      BR        LOGS3A
4085 042204                      LOGCML:
4086 042204 012737 031655 017352      MOV      #SCML,TEMP1
4087 042212 012737 000020 017350      MOV      #DLE,TEMP        ;SET UP MSG. AND TYPE
4088 042220 000406                      BR        LOGS3A         ;GO LOG EVENT AND TIME
4089 042222                      LOGCMD:
4090 042222 012737 031666 017352      MOV      #SCMD,TEMP1
4091 042230 012737 000022 017350      MOV      #DDE,TEMP
4092 042236 013737 015756 017362      LOGS3A: MOV      TRIBN,TEMP5
4093 042244 000436                      BR        LOGS3           ;GO LOG MSG TYPE AND TIME
4094
4095 042246                      LOGEOP:
4096 042246 012737 031677 017352      MOV      #SEOP,TEMP1
4097 042254 012737 000024 017350      MOV      #EOP,TEMP
4098 042262 000427                      BR        LOGS3           ;GO LOG MSG TYPE AND TIME
4099
4100 042264                      LOGMSC:
4101 042264 012737 031710 017352      MOV      #SMSC,TEMP1
4102 042272 012737 000016 017350      MOV      #MSC,TEMP
4103 042300 000420                      BR        LOGS3
4104
4105 042302 013746 017310                      LOGS1: MOV      ERRCNT,-(SP)      ;SAVE CURRENT ERROR COUNT
4106 042306 013737 015756 017362      MOV      TRIBN,TEMP5      ;SAVE TRIBN
4107 042314 004737 063706                      JSR      PC,DVMODS        ;GO GET MODEM STATUS
4108 042320 012604                      MOV      (SP)+,R4         ;GET SAVED ERRCNT VALUE
4109 042322 020437 017310                      CMP      R4,ERRCNT        ;WERE ANY ERRORS FOUND
4110 042326 001402                      BEQ      1$               ;BR IF NONE
4111 042330 000137 042550                      JMP      LOGEX            ;ELSE, LEAVE WITHOUT LOGGING ANYTHING
4112
4113 042334 013737 020522 017360      1$:    MOV      MODS,TEMP4      ; BUT THE DEVICE ERROR FROM 'DVMODS'
4114
4115 042342                      LOGS3:
4116 042342 022737 000006 017350      CMP      #RXC,TEMP        ;IF RXC DON'T PRINT
4117 042350 001434                      BEQ      LOGS5
4118 042352 032737 000001 017410      BIT      #STATB,PARAM
4119 042360 001430                      BEQ      LOGS5           ;IF NO STATUS SELECTED
4120
4121
4122 042362 022737 000010 017300      CMP      #10,LNCNT        ;HAVE WE DONE 10?
4123 042370 001012                      BNE      LOGS4           ;IF NOT GO TO 4

```

```

4124 042372 005037 017300 CLR LNCNT ;ELSE CLEAR IT
4125
4126 042376 PRINTF #CR ;ELSE PRINT CR
4127 042376 012746 031564 MOV #CR,-(SP)
4128 042402 012746 000001 MOV #1,-(SP)
4129 042406 010600 MOV SP,R0
4130 042410 104417 TRAP C$PNTF
4131 042412 062706 000004 ADD #4,SP
4132 042416
LOGS4: INC LNCNT ;INC COUNTER OF # OF AWAKE MSGS
4133 042416 005237 017300 PRINTF TEMP1 ;PRINT OPERATOR AWAKE MSG.
4134 042422
4135 042422 013746 017352 MOV TEMP1,-(SP)
4136 042426 012746 000001 MOV #1,-(SP)
4137 042432 010600 MOV SP,R0
4138 042434 104417 TRAP C$PNTF
4139 042436 062706 000004 ADD #4,SP
4140 042442 010346 LOGS5: MOV R3,-(SP) ;SAVE R3 ON THE STACK
4141 042444 013703 MOV EVTPTR,R3
4142 042450 113723 MOVB TEMP,(R3)+ ;LOG EVENT
4143 042454 013737 017442 017350 MOV CLKHZ,TEMP
4144 042462 163737 017452 017350 SUB TIMTCK,TEMP
4145 042470 113723 017350 MOVB TEMP,(R3)+ ;LOG TIME SINCE START
4146 042474 113723 017450 MOVB TIMSEC,(R3)+
4147 042500 113723 017446 MOVB TIMMIN,(R3)+ ;TICKS,SECS AND MINS.
4148 042504 013723 017354 MOV TEMP2,(R3)+ ;LOG EVNT ENTRY 3
4149 042510 013723 017356 MOV TEMP3,(R3)+ ;LOG EVNT ENTRY 4
4150 042514 013723 017360 MOV TEMP4,(R3)+ ;LOG EVNT ENTRY 5
4151 042520 013723 017362 MOV TEMP5,(R3)+ ;LOG EVNT ENTRY 6
4152 042524 020327 020520 CMP R3,#EVTEND
4153 042530 103404 BLO LOGS2 ;IF EVENT LOG FULL GO
4154 ;CONTINUE;ELSE GO TO 2
4155 042532 012713 177777 MOV #-1,(R3) ;LOG A TABLE END
4156 042536 012703 017464 MOV #EVTLOG,R3 ;PUT R3 TO START OF TABLE
4157 042542 010337 017462 LOGS2: MOV R3,EVTPTR ;RESTORE POINTER
4158 042546 012603 MOV (SP)+,R3 ;RESTORE R3
4159 042550 000207 LOGEX: RTS PC
4160
4161

```

```

4162          .SBTTL          DUMP EVENT LOG AND BASE TABLE
4163
4164
4165 042552 010246          REPORT: MOV      R2,-(SP)          ;SAVE R2,R3,R4 ON THE STACK
4166 042554 010346          MOV      R3,-(SP)
4167 042556 010446          MOV      R4,-(SP)
4168
4169          ;PRINT REPORT HELP MESSAGE
4170
4171 042560          PRINTF  #RHLPO
4172 042560 012746 025337          MOV      #RHLPO,-(SP)
4173 042564 012746 000001          MOV      #1,-(SP)
4174 042570 010600          MOV      SP,R0
4175 042572 104417          TRAP    C$PNTF
4176 042574 062706 000004          ADD     #4,SP
4177 042600 105037 003411          GETRCL: CLRB   P$GDBD          ;CLEAR GOOD BAD FLAG
4178 042604 105037 003410          CLRB   P$NNUF
4179
4180          ;PRINT PROMPT RPT>
4181
4182 042610          GMANID  CLISRP,CMDBUF,A,0,1,72.,NO
4183 042610 104443          TRAP    C$GMAN
4184 042612 000406          BR     10000$
4185 042614 003130          .WORD  CMDBUF
4186 042616 000142          .WORD  T$CODE
4187 042620 025332          .WORD  CLISRP
4188 042622 000000          .WORD  0
4189 042624 000001          .WORD  T$LOLIM
4190 042626 000110          .WORD  T$HILIM
4191 042630          10000$:
4192 042630 012737 003130 003374          MOV     #CMDBUF,P$BUFA
4193 042636 012737 044772 003376          MOV     #CLIRT,P$TREE
4194 042644 012737 044356 003400          MOV     #CLIRAC,P$ACT
4195 042652 005037 003254          CLR     QUALFG          ;CLEAR QUALIFIER FLAG LOCATION
4196 042656 004737 047644          JSR     PC,P$TRV        ;GO PARSE COMMAND LINE
4197 042662 105737 003411          TSTB   P$GDBD          ;SEE IF PARSED OK OR AN ERROR
4198 042666 001412          BEQ    1$
4199 042670          PRINTF #CLIERM
4200 042670 012746 023360          MOV     #CLIERM,-(SP)
4201 042674 012746 000001          MOV     #1,-(SP)
4202 042700 010600          MOV     SP,R0
4203 042702 104417          TRAP    C$PNTF
4204 042704 062706 000004          ADD     #4,SP
4205 042710 000137 042600          JMP     GETRCL
4206 042714 105737 003410 1$:          TSTB   P$NNUF          ;SEE IF INCOMPLETE COMMAND TYPED
4207 042720 001412          BEQ    10$
4208 042722          PRINTF #CLINUF
4209 042722 012746 023410          MOV     #CLINUF,-(SP)
4210 042726 012746 000001          MOV     #1,-(SP)
4211 042732 010600          MOV     SP,R0
4212 042734 104417          TRAP    C$PNTF
4213 042736 062706 000004          ADD     #4,SP
4214 042742 000137 042600          JMP     GETRCL
4215
4216 042746 023727 003252 000005 10$:          CMP     KEYWD1,#RPTSS
4217 042754 001003          BNE    20$
    
```

```

4218 042756 004737 043004 JSR PC,RPTTSS ;JUMP TO REPORT TSS
4219 042762 000706 BR GETRCL ;IF EQUAL JUMP BACK
4220 042764 023727 003252 000002 20$: CMP KEYWD1,#RPEXT ;SEE IF EXIT REPORT SECTION
4221 042772 001302 BNE GETRCL
4222 042774 012604 ENDALL: MOV (SP)+,R4 ;RESTORE R4,R3,R2
4223 042776 012603 MOV (SP)+,R3
4224 043000 012602 MOV (SP)+,R2
4225 043002 000207 RTS PC ;RETURN TO CALLING ROUTINE
4226
4227
4228 043004 012737 000046 021022 RPTTSS: MOV #46,TSSA ;SET KEY UP TO FIRST ERROR
4229 043012 005737 015756 TST TRIBN
4230 043016 001003 BNE RDTSS2 ;BRANCH IF TSS
4231 043020 012737 000054 021022 MOV #54,TSSA ;IF GSS USE 55
4232 043026 012737 000057 021020 RDTSS2: MOV #57,TSSE ;SET UP 57 AS END
4233 043034 122737 000105 021024 CMPB #105,TSSKEY ;IS THIS AN E
4234 043042 001422 BEQ RDTSS ;AND GO READ THEM
4235 043044 012737 000037 021022 MOV #37,TSSA
4236 043052 012737 000077 021020 MOV #77,TSSE ;SET UP LIMITS
4237 043060 122737 000106 021024 CMPB #106,TSSKEY ;IS THIS FULL
4238 043066 001410 BEQ RDTSS ;IF SO READ FULL
4239 043070 013737 021024 021022 MOV TSSKEY,TSSA
4240 043076 005337 021022 DEC TSSA
4241 043102 013737 021024 021020 MOV TSSKEY,TSSE
4242
4243 043110 005237 021022 RDTSS: INC TSSA
4244 043114 152777 000200 157726 BISB #RQI,@BSEL0 ;MAKE RQEST
4245 043122 004737 065044 JSR PC,TOORIO
4246 043126 012737 177777 017326 MOV #-1,TSSFLG ;SET FLAG
4247 043134 113777 015756 157714 MOVB TRIBN,@BSEL3
4248 043142 013777 021022 157714 MOV TSSA,@SEL6
4249 043150 112777 000001 157676 MOVB #01,@BSEL2 ;DO CONTROL IN READ TSS
4250 043156 023737 021022 021020 CMP TSSA,TSSE ;ARE WE DONE
4251 043164 001351 BNE RDTSS ;IF NOT GO BACK FOR MORE
4252 043166 152777 000200 157654 BISB #RQI,@BSEL0 ;MAKE RQEST
4253 043174 004737 065044 JSR PC,TOORIO
4254 043200 113777 015756 157650 MOVB TRIBN,@BSEL3
4255 043206 105077 157652 CLRB @BSEL6
4256 043212 112777 000001 157634 MOVB #01,@BSEL2 ;DO CONTROL IN [NO REQUEST]
4257 ;THIS GETS LAST OUTPUT
4258 043220 000207 RTS PC ;RETURN WHEN DONE
4259
4260
4261
4262
4263 043222 010246 REPLOG: MOV R2,-(SP) ;SAVE R2,R3,R4 ON THE STACK
4264 043224 010346 MOV R3,-(SP)
4265 043226 010446 MOV R4,-(SP)
4266
4267 043230 013702 017462 MOV EVTPTTR,R2 ;MAKE R2 A POINTER TO EVENT TABLE
4268 043234 023727 017464 177777 CMP EVTLOG,#-1 ;SEE IF EVENT TABLE IS EMPTY
4269 043242 001034 BNE RPTO ;BR IF NO
4270 043244 PRINTS #NULEVT ;IF EMPTY TELL OPERATOR.
4271 043244 012746 030405 MOV #NULEVT,-(SP)
4272 043250 012746 000001 MOV #1,-(SP)
4273 043254 010600 MOV SP,R0

```

```

4274 043256 104416                                TRAP  C$PNTS
4275 043260 062706 000004                          ADD  #4,SP
4276 043264 000137 044202                        JMP  ENDEVT      ;AND END
4277
4278 043270 162702 000014      RPT:  SUB  #14,R2  ;NOW POINT BACK TO TOP OF ENTRY U
4279                                ;JUST PRINTED
4280
4281 043274 020227 017464                        CMP  R2,#EVTLOG  ;POINTING TO TOP OF EVNT LOG QUEUE?
4282 043300 001010                                BNE  RPT1        ; BR IF NO
4283 043302 012702 020520                        MOV  #EVTEND,R2  ;SET R2 TO POINT TO BOTTOM OF LOG
4284 043306 026227 177776 177777                CMP  -2(R2),#-1
4285 043314 001007                                BNE  RPT0        ;IF END OF LOG IS NOT EMPTY
4286 043316 000137 044202                        JMP  ENDEVT      ;CONTINUE...ELSE EXIT
4287
4288 043322 020237 017462      RPT1:  CMP  R2,EVTPTR ;ARE WE BACK TO POINTER?
4289 043326 001002                                BNE  RPT0        ;IF NOT CONTINUE
4290 043330 000137 044202                        JMP  ENDEVT      ;IF SO EXIT....
4291
4292 043334 162702 000014      RPT0:  SUB  #14,R2  ;POINT R2 TO START OF ENTRY
4293 043340      RPTAA: PRINTS #EVTFO ;PRINT EVENT ENTRY HEADER
4294 043340 012746 030445                                MOV  #EVTFO,-(SP)
4295 043344 012746 000001                                MOV  #1,-(SP)
4296 043350 010600                                MOV  SP,R0
4297 043352 104416                                TRAP C$PNTS
4298 043354 062706 000004                          ADD  #4,SP
4299 043360 112203                                MOVB (R2)+,R3    ;PUT EVENT TYPE INTO R3
4300 043362 112237 020612                                MOVB (R2)+,EVTTC ;PUT EVENT TIME (TICKS,SECS,MIN IN TEMP LOC.S)
4301 043366 112237 020606                                MOVB (R2)+,EVTSEC
4302 043372 112237 020610                                MOVB (R2)+,EVTMIN
4303 043376                                PRINTS #EVTFO1,EVTMIN,EVTSEC,EVTTC,EVTLST(R3) ;PRINT EVENT TIME AND DESCRIPT.
4304 043376 016346 020560                                MOV  EVTLST(R3),-(SP)
4305 043402 013746 020612                                MOV  EVTTC,-(SP)
4306 043406 013746 020606                                MOV  EVTSEC,-(SP)
4307 043412 013746 020610                                MOV  EVTMIN,-(SP)
4308 043416 012746 030543                                MOV  #EVTFO1,-(SP)
4309 043422 012746 000005                                MOV  #5,-(SP)
4310 043426 010600                                MOV  SP,R0
4311 043430 104416                                TRAP C$PNTS
4312 043432 062706 000014                          ADD  #14,SP
4313 043436 000173 020622                        JMP  @RPTDSP(R3) ;DISPATCH TO DECODING SECTION FOR SPECIFIC TYPE
4314
4315 043442 012237 020614      RPTTXQ: MOV (R2)+,EVTADD ;STORE MESSAGE ADDRESS FOR PRINTING
4316 043446 012237 020616                        MOV (R2)+,EVTBCT ;STORE BYTE COUNT FOR PRINTING
4317 043452 012203                                MOV (R2)+,R3    ;STORE MODEM STATUS FOR PRINTING
4318 043454 004737 044324                        JSR  PC,PNTTRB  ;PRINT TRIB NO.
4319 043460                                PRINTS #EVTFO2,EVTADD,EVTBCT ;PRINT ADDR,BYTE CNT
4320 043460 013746 020616                                MOV  EVTBCT,-(SP)
4321 043464 013746 020614                                MOV  EVTADD,-(SP)
4322 043470 012746 030572                                MOV  #EVTFO2,-(SP)
4323 043474 012746 000003                                MOV  #3,-(SP)
4324 043500 010600                                MOV  SP,R0
4325 043502 104416                                TRAP C$PNTS
4326 043504 062706 000010                          ADD  #10,SP
4327 043510 004737 044212                        JSR  PC,RPTMSB  ;GO PRINT MODEM STATUS
4328 043514 000137 043270                        JMP  RPT        ;GO BACK FOR NEXT EVENT ENTRY
4329

```


DUMP EVENT LOG AND BASE TABLE

```

4330 043520 012237 020620      RPTDER: MOV      (R2)+,EVTTMP      ;GET ADDRESS OF DEVICE INFO MESSAGE
4331 043524 012237 020650      MOV      (R2)+,DEV1      ;STORE DEVICE REG CONTENTS FOR PRINTING
4332 043530 012237 020652      MOV      (R2)+,DEV2
4333 043534 012237 017362      MOV      (R2)+,TEMP5
4334 043540      PRINTS #EVT3,EVTMP      ;PRINT DEVICE REG CONTENTS.
4335 043540 013746 020620      MOV      EVTMP,-(SP)
4336 043544 012746 030644      MOV      #EVT3,-(SP)
4337 043550 012746 000002      MOV      #2,-(SP)
4338 043554 010600      MOV      SP,R0
4339 043556 104416      TRAP    C$PNTS
4340 043560 062706 000006      ADD     #6,SP
4341 043564      PRINTS #EVT3C,DEV1,DEV2
4342 043564 013746 020652      MOV      DEV2,-(SP)
4343 043570 013746 020650      MOV      DEV1,-(SP)
4344 043574 012746 030656      MOV      #EVT3C,-(SP)
4345 043600 012746 000003      MOV      #3,-(SP)
4346 043604 010600      MOV      SP,R0
4347 043606 104416      TRAP    C$PNTS
4348 043610 062706 000010      ADD     #10,SP
4349 043614 000137 043270      JMP     RPT      ;GO BACK FOR NEXT EVENT ENTRY
4350
4351 043620 005037 020650      RPTDVI: CLR     DEV1
4352 043624 005037 020652      CLR     DEV2      ;CLEAR UPPER BYTES OF DEV1 & DEV2 BEFORE USE
4353 043630 112237 020650      MOV     (R2)+,DEV1 ;STORE SETUP OPERATION PARAMETERS FOR PRINTING
4354 043634 112237 020652      MOV     (R2)+,DEV2
4355 043640 012237 020654      MOV     (R2)+,DEV3
4356 043644 012237 020656      MOV     (R2)+,DEV4
4357 043650 010246      MOV     R2,-(SP)  ;SAVE R2 ON THE STACK
4358 043652 004737 047342      JSR    PC,SHWOP  ;GO PRINT MODE, MAINT-LOOP TYPE, PARAMTERS.
4359 043656 012602      MOV     (SP)+,R2  ;RESTORE R2
4360 043660 012237 017362      MOV     (R2)+,TEMP5 ;DUMMY MOVE
4361 043664 000137 043270      JMP     RPT      ;GO BACK FOR NEXT EVENT ENTRY
4362 043670 012237 020614      RPTTEOP: MOV    (R2)+,EVTADD
4363 043674 012237 020616      MOV    (R2)+,EVTBCT
4364 043700 012237 020620      MOV    (R2)+,EVTTMP
4365 043704 012237 017362      MOV    (R2)+,TEMP5 ;DUMMY MOVE
4366
4367      ;PRINT PASCOUNT ERROR COUNT RX THRES AND TX TTHRES
4368
4369      PRINTS #EVT4B,EVTADD,EVTBCT
4370 043710 013746 020616      MOV     EVTBCT,-(SP)
4371 043714 013746 020614      MOV     EVTADD,-(SP)
4372 043720 012746 031014      MOV     #EVT4B,-(SP)
4373 043724 012746 000003      MOV     #3,-(SP)
4374 043730 010600      MOV     SP,R0
4375 043732 104416      TRAP    C$PNTS
4376 043734 062706 000010      ADD     #10,SP
4377 043740      PRINTS #EVT44,EVTTMP,TEMP5
4378 043740 013746 017362      MOV     TEMP5,-(SP)
4379 043744 013746 020620      MOV     EVTTMP,-(SP)
4380 043750 012746 031053      MOV     #EVT44,-(SP)
4381 043754 012746 000003      MOV     #3,-(SP)
4382 043760 010600      MOV     SP,R0
4383 043762 104416      TRAP    C$PNTS
4384 043764 062706 000010      ADD     #10,SP
4385 043770 000137 043270      JMP     RPT      ;THEN GO GET NEXT EVENT ENTRY

```

```

4386
4387
4388 043774 012237 020614      RPTDDE: MOV      (R2)+,EVTADD ;STORE MESSAGE ADDRESS FOR PRINTING
4389 044000 012237 020616      MOV      (R2)+,EVTBCT ;STORE BYTE COUNT FOR PRINTING
4390 044004 012237 020620      MOV      (R2)+,EVTTMP ;STORE TOTAL # OF CMP ERRORS
4391 044010 004737 044324      JSR      PC,PNTTRB ;PRINT TRIB NO.
4392 044014      PRINTS #EVT4,EVTADD,EVTBCT,EVTTMP ;PRINT ADDR, BYTE CNT, # CMP ERRS
4393 044014 013746 020620      MOV      EVTTMP,-(SP)
4394 044020 013746 020616      MOV      EVTBCT,-(SP)
4395 044024 013746 020614      MOV      EVTADD,-(SP)
4396 044030 012746 031122      MOV      #EVT4,-(SP)
4397 044034 012746 000004      MOV      #4,-(SP)
4398 044040 010600      MOV      SP,R0
4399 044042 104416      TRAP    C$PNTS
4400 044044 062706 000012      ADD     #12,SP
4401 044050 000137 043270      JMP     RPT ;THEN GO GET NEXT EVENT ENTRY
4402
4403 044054      RPTDLE:
4404 044054 012237 020614      RPTDCK: MOV      (R2)+,EVTADD ;STORE MSG ADDR FOR PRINT
4405 044060 012237 020616      MOV      (R2)+,EVTBCT ;STORE BYTE COUNT
4406 044064 012237 020620      MOV      (R2)+,EVTTMP ;STORE BYTE COUNT COMP
4407 044070 004737 044324      JSR      PC,PNTTRB ;PRINT TRIB NO.
4408 044074      PRINTS #EVT4A,EVTADD,EVTBCT,EVTTMP ;PRINT ADDR,RXBYTES,CMPBYTES.
4409 044074 013746 020620      MOV      EVTTMP,-(SP)
4410 044100 013746 020616      MOV      EVTBCT,-(SP)
4411 044104 013746 020614      MOV      EVTADD,-(SP)
4412 044110 012746 031224      MOV      #EVT4A,-(SP)
4413 044114 012746 000004      MOV      #4,-(SP)
4414 044120 010600      MOV      SP,R0
4415 044122 104416      TRAP    C$PNTS
4416 044124 062706 000012      ADD     #12,SP
4417
4418 044130 000137 043270      JMP     RPT ;THEN GO GET NEXT EVENT ENTRY
4419
4420
4421 044134      RPTMSC:
4422
4423 044134 012203      MOV      (R2)+,R3 ;PUT OLD MODEM STATUS IN R3 FOR PRINTING
4424 044136 004737 044212      JSR      PC,RPTMSB ;GO PRINT OLD MODEM STATUS
4425 044142      PRINTS #EVMOCG ;GO PRINT "CHANGED TO:"
4426 044142 012746 031406      MOV      #EVMOCG,-(SP)
4427 044146 012746 000001      MOV      #1,-(SP)
4428 044152 010600      MOV      SP,R0
4429 044154 104416      TRAP    C$PNTS
4430 044156 062706 000004      ADD     #4,SP
4431 044162 012203      MOV      (R2)+,R3 ;PUT NEW MODEM STATUS IN R3 FOR PRINTING
4432 044164 004737 044212      JSR      PC,RPTMSB ;GO PRINT NEW MODEM STATUS
4433 044170 012203      MOV      (R2)+,R3 ;POP NULL WORD FROM ENTRY OUT OF LOG
4434 044172 012237 017362      MOV      (R2)+,TEMP5 ;DUMMY MOVE
4435 044176 000137 043270      JMP     RPT ;THEN GO GET NEXT EVENT
4436
4437 044202      ENDEVT: ;RETURN TO CALLER AFTER REG RESTORE
4438 044202 012604      MOV      (SP)+,R4 ;RESTORE R4,R3,R2
4439 044204 012603      MOV      (SP)+,R3
4440 044206 012602      MOV      (SP)+,R2
4441 044210 000207      RTS     PC ;RETURN TO CALLING ROUTINE

```

```

4442
4443
4444 ;REPORT MODEM STATUS SUBROUTINE
4445 ; PART OF STATISTICAL REPORTING (DUMPING EVENT LOG)
4446
4447 044212 RPTMSB: PRINTS #EVMOHD ;PRINT MODEM STATUS HEADER
4448 044212 012746 031431 MOV #EVMOHD,-(SP)
4449 044216 012746 000001 MOV #1,-(SP)
4450 044222 010600 MOV SP,R0
4451 044224 104416 TRAP C$PNTS
4452 044226 062706 000004 ADD #4,SP
4453 044232 012704 020524 MOV #MOBITS,R4 ;MAKE R4 A POINTER TO MODEM SIG. BIT DEF. TABLE
4454 044236 012705 020542 MOV #MOMSGS,R5 ;MAKE R5 A POINTER TO MODEM MSG. POSITION TABLE
4455 044242 005714 6$: TST (R4) ;SEE IF BIT AVAILABLE FROM DEVICE
4456 044244 001004 BNE 7$ ;BR IF THAT MODEM SIG. AVAILABLE
4457 044246 112735 000130 MOVB #'X,@(R5)+ ;ELSE PUT 'X' IN REPORT IF SIGNAL NOT AVAILABLE
4458 044252 005724 TST (R4)+ ;BUMP R4 TO POINT TO NEXT BIT DEFINITION
4459 044254 000407 BR 9$ ;GO SEE IF CHECKED ALL MODEM SIGNALS
4460 044256 032403 7$: BIT (R4)+,R3 ;IF THERE, SEE IF THAT BIT IN DEVICE'S ENTRY=1
4461 044260 001403 BEQ 8$ ;BR IF BIT (SIGNAL) VALUE =0
4462 044262 112735 000061 MOVB #'1,@(R5)+ ;IF=1, PUT '1' IN REPORT MESSAGE
4463 044266 000402 BR 9$ ;GO SEE IF ALL MODEM SIGNALS CHECKED
4464 044270 112735 000060 8$: MOVB #'0,@(R5)+ ;IF BIT(SIGNAL)=0, PUT '0' IN REPORT MESSAGE
4465 044274 020427 020542 9$: CMP R4,#MOBITE ;SEE IF ALL BITS(SIGNALS) CHECKED
4466 044300 002760 BLT 6$ ;LOOP UNTIL ALL SIGNALS(BITS) CHECKED
4467 044302 PRINTS #EVMOST ;THEN PRINT MODEM SIGNAL VALUE MESSAGE
4468 044302 012746 031511 MOV #EVMOST,-(SP)
4469 044306 012746 000001 MOV #1,-(SP)
4470 044312 010600 MOV SP,R0
4471 044314 104416 TRAP C$PNTS
4472 044316 062706 000004 ADD #4,SP
4473 044322 000207 RTS PC ;RETURN TO EVENT DECODING
4474
4475 ;PRINT TRIBNO
4476
4477 044324 012237 017362 PNTTRB: MOV (R2)+,TEMP5
4478 044330 PRINTS #EVTF6,TEMP5 ;PRINT TRIB NUMBER.
4479 044330 013746 017362 MOV TEMP5,-(SP)
4480 044334 012746 030754 MOV #EVTF6,-(SP)
4481 044340 012746 000002 MOV #2,-(SP)
4482 044344 010600 MOV SP,R0
4483 044346 104416 TRAP C$PNTS
4484 044350 062706 000006 ADD #6,SP
4485 044354 000207 RTS PC ;RETURN TO EVENT

```

4486						
4487	044356			.SBTTL	CLI FOR REPORT CODING SECTION	
4488	044356	006302		CLIRAC:		
4489	044360	016202	044374	ASL	R2	
4490	044364	062702	044374	MOV	10\$(R2),R2	:FORM ADDRESS OF ACTION ROUTINE
4491	044370	004712		ADD	#10\$,R2	
4492	044372	000207		JSR	PC,(R2)	
4493				RTS	PC	
4494	044374	000034		10\$:	.WORD	ACTRNL-10\$
4495	044376	000036			.WORD	ACTRHL-10\$
4496	044400	000102			.WORD	ACTREX-10\$
4497	044402	000112			.WORD	ACTRLG-10\$
4498	044404	000126			.WORD	ACTRGS-10\$
4499	044406	000156			.WORD	ACTRTS-10\$
4500	044410	000202			.WORD	ACTRTN-10\$
4501	044412	000150			.WORD	ACTRSE-10\$
4502	044414	000274			.WORD	ACTRSF-10\$
4503	044416	000310			.WORD	ACTRSO-10\$
4504	044420	000026			.WORD	ACTRNF-10\$
						:RPHLP
						:RPEXT
						:RPLOG
						:RPGSS
						:RPTSS
						:RPTSN
						:RPSWE
						:RPSWF
						:RPSWO
						:RNOTNF

```

4505 .SBTTL REPORT COMMAND ACTION ROUTINES
4506 044422 112737 177777 003410 ACTRNF: MOV # -1, P$NUF ;SET FLAG TO SAY MORE NEEDED
4507 044430 000207 ACTRNL: RTS PC
4508 044432 012702 003304 ACTRHL: MOV #RHLPTB, R2 ;SETUP R2 AS A POINTER TO HELP MSG TABLE
4509 044436 1$ : PRINTF #HLPF, (R2)+ ;PRINT HELP INFORMATION MESSAGES
4510 044436 012246 MOV (R2)+, -(SP)
4511 044440 012746 024114 MOV #HLPF, -(SP)
4512 044444 012746 000002 MOV #2, -(SP)
4513 044450 010600 MOV SP, R0
4514 044452 104417 TRAP C$PNTF
4515 044454 062706 000006 ADD #6, SP
4516 044460 020227 003314 CMP R2, #RHLPEN ;SEE IF ALL INFO PRINTED YET
4517 044464 001364 BNE 1$ ;IF NO KEEP PRINTING
4518 044466 012737 000001 003252 MOV #RPHLP, KEYWD1
4519 044474 000207 RTS PC
4520 044476 012737 000002 003252 ACTREX: MOV #RPEXT, KEYWD1 ;SET UP EXIT WORD
4521 044504 000207 RTS PC
4522 044506 004737 043222 ACTRLG: JSR PC, REPLOG ;GO REPORT DCLT EVENT LOG
4523 044512 012737 000003 003252 MOV #RPLOG, KEYWD1
4524 044520 000207 RTS PC
4525 044522 105037 015756 ACTRGS: CLRB TRIBN ;FOR GLOBAL STATUS MAKE TRIN =0
4526 044526 012737 000105 021024 MOV #105, TSSKEY
4527 044534 012737 000005 003252 MOV #RPTSS, KEYWD1 ;SET UP KEY WORD
4528 044542 000207 RTS PC ;AND RETURN
4529 044544 105037 003410 ACTRSE: CLRB P$NUF ;CLEAR NOT NUF FLAG
4530 044550 000207 RTS PC
4531 044552 012737 000105 021024 ACTRTS: MOV #105, TSSKEY
4532 044560 012737 000005 003252 MOV #RPTSS, KEYWD1 ;SET UP KEY WORD
4533 044566 112737 177777 003410 MOV # -1, P$NUF
4534 044574 000207 RTS PC ;AND RETURN
4535 044576 105037 003410 ACTRTN: CLRB P$NUF ;CLEAR NOT NUF
4536 044602 012705 000040 MOV #32, R5
4537 044606 012702 015712 MOV #TRIBLS, R2
4538 044612 122237 003404 3$: CMPB (R2)+, P$NUM
4539 044616 001420 BEQ 4$
4540 044620 005305 DEC R5
4541 044622 001373 BNE 3$
4542 044624 PRINTF #SHTNF, P$NUM
4543 044624 013746 003404 MOV P$NUM, -(SP)
4544 044630 012746 026163 MOV #SHTNF, -(SP)
4545 044634 012746 000002 MOV #2, -(SP)
4546 044640 010600 MOV SP, R0
4547 044642 104417 TRAP C$PNTF
4548 044644 062706 000006 ADD #6, SP
4549 044650 112737 177777 003411 MOV # -1, P$GDBD
4550 044656 000403 BR 5$
4551 044660 113737 003404 015756 4$: MOV P$NUM, TRIBN
4552 044666 000207 5$: RTS PC
4553 044670 105037 003410 ACTRSF: CLRB P$NUF
4554 044674 012737 000106 021024 MOV #106, TSSKEY
4555 044702 000207 RTS PC
4556 044704 105037 003410 ACTRSO: CLRB P$NUF
4557 044710 023727 003404 000037 CMP P$NUM, #37
4558 044716 003416 BLE 2$
4559 044720 PRINTF #RPTIV, P$NUM
4560 044720 013746 003404 MOV P$NUM, -(SP)
  
```

4561	044724	012746	025542
4562	044730	012746	000002
4563	044734	010600	
4564	044736	104417	
4565	044740	062706	000006
4566	044744	112737	177777 003411
4567	044752	000406	
4568	044754	013737	003404 021024 2\$:
4569	044762	052737	000040 021024 3\$:
4570	044770	000207	

MOVB	#-1,PSGDBD
BR	3\$
MOV	P\$NUM,TSSKEY
BIS	#BIT5,TSSKEY
RTS	PC

MOV	#RPTIV,-(SP)
MOV	#2,-(SP)
MOV	SP,R0
TRAP	C\$PNTF
ADD	#6,SP

```
4571 .SBTTL REPORT CODE COMMAND LINE PARSING TREE
4572
4573 044772 CLIRT: CLI CLISPA,0,R10$ ;SKIP ANY SPACES
4574 044776 R10$: CLI <'?'>,RPHLP,R11$ ;IS FIRST NON-SP CHAR A '?'?
4575 045002 CLI CLIEXI,0 ;EXIT
4576 045004 R11$: CLI CLISTR,RPHLP,R12$,<'HELP'>
4577 045020 CLI CLIEXI,0
4578 045022 R12$: CLI CLISTR,RPEXT,R13$,<'EXIT'>
4579 045036 CLI CLIEXI,0
4580 045040 R13$: CLI CLISTR,RPGSS,R14$,<'GSS'>
4581 045052 CLI CLIBR,0,R20$
4582 045056 R14$: CLI CLISTR,RPLOG,R15$,<'LOG'>
4583 045070 CLI CLIEXI,0
4584 045072 R15$: CLI CLISTR,RPTSS,R30$,<'TSS'>
4585 045104 CLI CLISPA,RNCTNF,R30$
4586 045110 CLI CLIDEC,RPTSN,R30$
4587 045114 R20$: CLI <'/'>,RNOTNF,R125$
4588 045120 CLI CLISTR,RPSWE,R21$,<'ERROR'>
4589 045134 CLI CLIEXI,0
4590 045136 R21$: CLI CLISTR,RPSWF,R22$,<'FULL'>
4591 045152 CLI CLIEXI,0
4592 045154 R22$: CLI CLISTR,RNOTNF,R30$,<'OFFSET'>
4593 045172 CLI <'=>,0,R30$
4594 045176 CLI CLIOCT,RPSWO,R30$
4595 045202 CLI CLIEXI,0
4596 045204 R30$: CLI CLIERR,0
4597 045206 R125$: CLI CLIEXI,0
4598
4599
```

4600
4601
4602
4603
4604
4605
4606
4607
4608
4609
4610
4611
4612
4613
4614
4615
4616
4617
4618
4619
4620
4621
4622
4623
4624
4625
4626
4627
4628
4629
4630
4631
4632
4633
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655

.SBTTL DUMP BYTES OR WORDS

```

  :++
  : FUNCTIONAL DESCRIPTION:
  : DUMPSR - DUMP BYTES OR WORDS SUBROUTINE
  :
  : THIS SUBROUTINE PRINTS THE CONTENTS OF THE LOCATIONS BETWEEN
  : A STARTING AND END ADDRESS IN LOCS. 'STADD' AND 'ENADD'.
  : THE WORD OR BYTE CONTENTS ARE PRINTED 8 TO A LINE WITH THE
  : ADDRESS OF THE FIRST BYTE AS THE FIRST 6 OCTAL CHARS. FOLLOWED
  : BY A SEMICOLON.
  :
  : INPUTS:
  : STADD= STARTING ADDRESS (FIRST LOC. TO PRINT)
  : ENADD= END ADDRESS (LAST LOCATION TO DUMP)
  : BYTBIT= 1 IF SUPPOSED TO PRINT 'BYTES'
  :         0 IF SUPPOSED TO PRINT 'WORDS'
  :
  : OUTPUTS:
  : CONTENTS OF A RANGE OF LOC.S PRINTED ON THE OPERATORS CONSOLE.
  :
  : CALLING SEQUENCE:
  : JSR PC,DUMPSR ;CALL DUMP BYTES SUBROUTINE
  :--
  
```

```

DUMPSR: MOV STADD,R2 ;SET R2 UP TO STARTING ADDR.
DUM4: CLR R3 ;CLEAR R3
      PRINTF #BASM1,R2 ;PRINT ADDRESS
  
```

```

MOV R2,-(SP)
MOV #BASM1,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTF
ADD #6,SP
  
```

```

DUM3: TST BYTBIT ;IS THIS BYTE OR WORD
      BEQ DUM1 ;BR IF WORD
      MOVB (R2)+,TEMP ;MOV BYTE TO TEMP
      PRINTF #BASM3,<B,TEMP> ;PRINT BYTE
  
```

```

CLR -(SP)
BISB TEMP,(SP)
MOV #BASM3,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTF
ADD #6,SP
  
```

```

DUM1: BR DUM2
      PRINTF #BASM2,(R2)+ ;PRINT WORD
  
```

```

MOV (R2)+,-(SP)
MOV #BASM2,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C$PNTF
ADD #6,SP
  
```

```

DUM2: CMP R2,ENADD ;COMPARE FOR LAST ADD
  
```

```

045210 013702 017312
045214 005003
045216 010246
045220 012746 030377
045224 012746 000002
045230 010600
045232 104417
045234 062706 000006
045240 005737 017316
045244 001416
045246 112237 017350
045252 005046
045254 153716 017350
045260 012746 030361
045264 012746 000002
045270 010600
045272 104417
045274 062706 000006
045300 000411
045302 012246
045304 012746 030370
045310 012746 000002
045314 010600
045316 104417
045320 062706 000006
045324 020237 017314
  
```


4656 045330 003005
4657 045332 005203
4658 045334 022703 000010
4659 045340 001725
4660 045342 000736
4661
4662 045344 000207
4663

BGT DUMEX
INC R3
CMP #8.,R3
BEQ DUM4
BR DUM3

DUMEX: RTS PC

:IF DONE EXIT
:ELSE BUMP R3
:HAVE WE PRINTED 8 ACROSS
:IF SO GO BACK TO 4
:ELSE GO BACK AND PRINT ANOTHER
:BYTE OR WORD
:RETURN TO CALLER

```

4664 .SBTTL UPDATE TOTAL CHAR. COUNT SUBROUTINE
4665
4666
4667 :++
4668 : FUNCTIONAL DESCRIPTION:
4669 : UPDATES TOTAL CHAR. COUNT TOTCC BASED ON CURCC.
4670 : LAST MESSAGE IS TRUNCATED TO FIT INTO THE
4671 : BUFFER IF TOTAL CHAR. COUNT EXCEEDS 'BUFLIM' A MESSAGE
4672 : IS PRINTED TELLING THE OPERATOR THE TRUNCATION OCCURRED.
4673
4674 : INPUTS:
4675 : CURCC= CHAR. COUNT OF MESSAGE BEING ADDED
4676 : TOTCC= TOTAL CHAR COUNT OF BUFFER ITS BEING ADDED TO
4677
4678 : OUTPUTS:
4679 : MESSAGE TO OPERATOR IF MESSAGE TRUNCATED TO FIT
4680
4681 : FUNCTIONAL SIDE EFFECTS:
4682 : LOCATION 'TEMP' USED FOR CALCULATIONS
4683 : CALLING SEQUENCE:
4684 : JSR PC,ADDCC ;UPDATED TOTAL CHAR. COUNT
4685 :--
4686
4687 045346 063737 017334 017344 ADDCC: ADD CURCC,TOTCC ;ADD CURRENT TO TOTAL
4688 045354 022737 001000 017344 CMP #BUFLIM,TOTCC ; COMPARE TO 'BUFLIM'
4689 045362 103027 BHIS ADDC1 ;IF NOT MORE THEN 'BUFLIM' EXIT
4690
4691 ; PRINT MESSAGE AND TRUNCATE COUNT
4692
4693 PRINTF #MSGTRU
4694 045364 012746 027444 MOV #MSGTRU,-(SP)
4695 045370 012746 000001 MOV #1,-(SP)
4696 045374 010600 MOV SP,R0
4697 045376 104417 TRAP C$PNTF
4698 045400 062706 000004 ADD #4,SP
4699 045404 163737 017334 017344 SUB CURCC,TOTCC ;SUB CURRENT FROM TOTAL
4700 045412 012737 001000 017350 MOV #BUFLIM,TEMP ;MOV 'BUFLIM' TO TEMP
4701 045420 163737 017344 017350 SUB TOTCC,TEMP ;SUB TOTAL FROM 'BUFLIM'
4702 045426 013737 017350 017334 MOV TEMP,CURCC ;AND ESTABLISH NEW CURRENT
4703 045434 063737 017334 017344 ADD CURCC,TOTCC ;ADD 'ADJUSTED CURRENT' TO TOTAL CHAR. CNT.
4704 045442 000207 ADDC1: RTS PC ;RETURN TO CALLER
4705

```

```

4706 .SBTTL BUILD MESSAGE BUFFERS SUBROUTINE
4707
4708
4709 : **
4710 : FUNCTIONAL DESCRIPTION:
4711 : BLDBUF-- BUILD POINTER TABLE AND BUFFERS
4712
4713 : THIS SUBROUTINE ADDS A MESSAGE TO THE TRANSMIT OR EXPECT LIST
4714 : USING THE POINTER, BYTE COUNT, AND ADDRESS PASSED TO IT.
4715
4716 : INPUTS:
4717 : CURCC= CHAR. COUNT OF MESSAGE TO BE ADDED
4718 : CURADD= ADDRESS OF MESSAGE TO BE ADDED
4719 : CPTR= ADDRESS OF POINTER TABLE WORD WHERE MESSAGE POINTERS ARE
4720 : TO BE BUILT
4721 : MSGTYP= VALUE TO USE AS AN INDEX TO FIND SOURCE OF MESSAGE DATA
4722 : INDEX INTO DMSGCT() AND DMSGAD().
4723
4724 : OUTPUTS:
4725 : A MESSAGE ADDED TO EITHER TXBUF OR CMPBUF
4726 : APPROPRIATE POINTERS IN PTRTAB POINTER TABLE
4727
4728 : CALLING SEQUENCE:
4729 : JSR PC,BLDBUF ;BUILD MESSAGE IN BUFFER AND ADD PTRS.
4730
4731 BLDBUF:
4732 MOV R2,-(SP) ;SAVE R2 AND R3 ON THE STACK
4733 MOV R3,-(SP)
4734 MOV CPTR,R2
4735
4736 BLDB1: MOV CURADD,(R2)+ ;PUT CURRENT ADD ON POINTER TAB
4737 MOV CURCC,(R2)+ ;PUT CURRENT CC ON POINTER TAB
4738 MOV R2,CPTR ;PUT UPDATED R2 BACK TO CURRENT POINT
4739 MOV MSGTYP,R2 ;GET MESSAGE TYPE TO USE AS INDEX
4740 ASL R2 ;DOUBLE FOR WORD INDEX
4741 MOV CURADD,TEMP ;MOVE CURRENT ADD TO TEMP
4742 ADD CURCC,TEMP ;ADD CHAR COUNT TO IT TO GET END
4743 MOV CURADD,R3 ;SET R3 TO CURRENT START ADD
4744 BLDB2: MOV DMSGCT(R2),TEMP2 ;GET BYTE COUNT
4745 MOV DMSGAD(R2),R4 ;PUT STARTING FROM ADD IN R4
4746 ADD R4,TEMP2 ;ADD IT TO TEMP2 TO GET END OF FROM
4747 BLDB3: MOVB (R4)+,(R3)+ ;MOV BYTE FROM PATTERN TO BUFFER
4748 CMP R3,TEMP ;ALL DONE?
4749 BEQ BLDBEX ;IF SO EXIT
4750 CMP R4,TEMP2 ;IS PATTERN COUNT EXPIRED
4751 BEQ BLDB2 ;IF SO GO START AGAIN
4752 BR BLDB3 ;IF NOT GET ANOTHER BYTE
4753 BLDBEX: ADD CURCC,CURADD ;BUMP CURADD
4754 MOV (SP)+,R3 ;RESTORE R3 AND R2
4755 MOV (SP)+,R2
4756 RTS PC ;RETURN TO CALLER
4757

```

4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813

.SBTTL CREATE FACSIMILE OF TX BUFFER AND MESSAGE LIST

..**
..FUNCTIONAL DESCRIPTION:

FACSIMILE: THIS ROUTINE IS USED TO CREATE A FACSIMILE OF THE OF THE TRANSMIT LIST AND TRANSMIT BUFFER IN THE EXPECT LIST AND EXPECT BUFFER. THE ROUTINE IS NORMALLY CALLED WHEN USER COMMAND 'SET E [XPECT]= T [RANSMIT] IS ENTERED.

CALLING SEQUENCE: JSR PC,FACSIMILE

DEFINITIONS CMPBUF = EXPECTED DATA BUFFER HOLDS MAX 512 BYTES
TXBUF = TRANSMIT DATA BUFFER HOLDS MAX 512 BYTES
TTOTCC = NUMBER OF BYTES IN TXBUF
PTRTAB = TOP OF MESSAGE LIST POINTER TABLE
CTOTCC = NUMBER OF BYTES IN EXPECT MESSAGE
CMPTOT = NUMBER OF EXPECTED MESSAGES
CMPPTR = EXPECTED MESSAGE LIST POINTER
TXPTR = TRANSMIT MESSAGE LIST POINTER
TXMTOT = NUMBER OF TRANSMIT MESSAGES
CCURAD = STORAGE ADDRESS OF MESSAGE IN CMPBUF
MSGLIN = MAXIMUM NUMBER OF MESSAGES THAT CAN BE STORED
BUFLIM = NUMBER OF BYTES IN BUFFER

BEGIN FACSIMILE ROUTINE
(*COPY TXBUF ==> CMPBUF*)
..SAVE R1
..INIT R1
..REPEAT
....[CMPBUF]R1=[TXBUF]R1
....R1=R1+1
..UNTIL R1 = BUFLIM

(*NOW CALCULATE EXPECT LIST MESSAGE POINTER*)
..CMPPTR = PTRTAB + (2 * MSGLIM)

(*NOW PRIME THE WHILE - DO LOOP*)
..TXPTR = PTRTAB
..CCURAD = CMPBUF
..TXPTR = TXPTR + 2
..CTOTCC = [TXPTR]
..CMPTOT = 0
..WHILE TXMTOT <> CMPTOT DO
....[CMPPTR] = CCURAD
....CMPPTR = CMPPTR + 2
....[CMPPTR] = CTOTCC
....TXPTR = TXPTR + 4
....CCURAD = CCURAD + CTOTCC
....CTOTCC = [TXPTR]
....CMPPTR = CMPPTR + 2
....CMPTOT = CMPTOT + 1
..END WHILE DO
..CTOTCC = TTOTCC
END FACSIMILE ROUTINE

```

4814
4815 045570
4816 045570 010146
4817 045572 005001
4818 045574 116161 003416 004416 10$:
4819 045602 005201
4820 045604 020127 001000
4821 045610 001371
4822
4823 045612 012701 000017 20$:
4824 045616 006301
4825 045620 006301
4826 045622 012737 011416 017240
4827 045630 060137 017240
4828 045634 005001
4829
4830
4831 045636 012737 011416 017236
4832 045644 012737 004416 017246
4833 045652 062737 000002 017236
4834 045660 017737 151352 017244
4835 045666 005037 017242
4836
4837
4838 045672 023737 017260 017242 30$:
4839 045700 001430
4840 045702 013777 017246 151330
4841 045710 062737 000002 017240
4842 045716 013777 017244 151314
4843 045724 062737 000004 017236
4844 045732 063737 017244 017246
4845 045740 017737 151272 017244
4846 045746 062737 000002 017240
4847 045754 005237 017242
4848 045760 000744
4849
4850 045762 013737 017262 017244 40$:
4851
4852
4853 045770 012601
4854 045772 000207
4855
4856
4857
  
```

```

FACSIMILE:
MOV R1,-(SP) ;SAVE R1
CLR R1 ;INIT R1
MOVB TXBUF(R1),CMPBUF(R1) ;COPY TX BUFFER TO EXPECTED BUFFER
INC R1 ;BUMP INDEX
CMP R1,#BUFLIM ;ALL DATA COPIED ?
BNE 10$ ;NO,BRANCH

20$:
MOV #MSGLIM,R1 ;MESSAGE LIMIT
ASL R1 ;MULTIPLY BY 2
ASL R1 ;MULTIPLY BY 2
MOV #PTRTAB,CMPPTR ;TOP OF POINTER TABLE
ADD R1,CMPPTR ;START OF EXPECTED POINTER TABLE
CLR R1 ;INIT R1

;SET UP WHILE - DO LOOP
MOV #PTRTAB, TXPTR ;TX POINTER NOW AT TOP OF TABLE
MOV #CMPBUF,CCURAD ;TRANSFER ADDRESS OF 1ST MESSAGE
ADD #2, TXPTR ;BUMP POINTER
MOV @TXPTR,CTOTCC ;BYTE COUNTER 1ST MESSAGE
CLR CMPTOT ;INIT EXPECTED MESSAGE COUNT

;WHILE TX MESSAGE TOTAL <> EXPECTED MESSAGE TOTAL DO
CMP TXMTOT,CMPTOT ;ALL MESSAGES COPIED ?
BEQ 40$ ;YES,BRANCH
MOV CCURAD,@CMPPTR ;TRANSFER ADDRESS OF MESSAGE
ADD #2,CMPPTR ;BUMP POINTER
MOV CTOTCC,@CMPPTR ;BYTE COUNT OF MESSAGE
ADD #4, TXPTR ;BUMP TX MESSAGE POINTER
ADD CTOTCC,CCURAD ;CALC. TRANSFER ADDRESS
MOV @TXPTR,CTOTCC ;BYTE COUNT NEXT MESSAGE
ADD #2,CMPPTR ;BUMP POINTER
INC CMPTOT ;INCREMENT MESSAGE COUNT
BR 30$ ;DO IT AGAIN

;END WHILE - DO
MOV TTOTCC,CTOTCC ;COPY TOTAL CHARACTER COUNT

;END ROUTINE
MOV (SP)+,R1 ;RESTORE R1
RTS PC ;RETURN
  
```

```

4858      .SBTTL          DO ALL GLOBAL PARMAS
4859
4860
4861      :++
4862      : FUNCTIONAL DESCRIPTION:          DOGLOB - ASK QUESTIONS ABOUT ALL GLOBALS
4863      :
4864      : THIS ROUTINE ASKS QUESTIONS TO ALL GLOBAL POLL PARMS
4865      : IF NESCESSARY THEN CLEARS THE WRITE GLOBAL FLAG
4866      :
4867      : CALLING SEQUENCE:
4868      :             JSR          PC,DOGLOB
4869      :--
4870      DOGLOB: MOV          R2,-(SP)          ;SAVE R2,R3,R4 ON THE STACK
4871      MOV          R3,-(SP)
4872      MOV          R4,-(SP)
4873      CLRB         WRFLG          ;CLEAR WRITE GLOBAL FLAG
4874      PRINTF       #POLPM3        ;PRINT GLOBAL PARAMS ARE
4875      MOV          #POLPM3,-(SP)
4876      MOV          #1,-(SP)
4877      MOV          SP,R0
4878      TRAP        C$PNTF
4879      ADD          #4,SP
4880
4881      DOGL1: CLR          R3
4882      MOV          #32,R2
4883      INC          R2
4884      MOV          R2,R4
4885      ASL          R4
4886      MOV          GLBPLS(R3),TEMP ;GET DEFAULT
4887      MOV          GSSLST(R4),CONOTM
4888      PRINTF       CONOTM,TEMP
4889      MOV          TEMP,-(SP)
4890      MOV          CONOTM,-(SP)
4891      MOV          #2,-(SP)
4892      MOV          SP,R0
4893      TRAP        C$PNTF
4894      ADD          #6,SP
4895      GMANID EQUQ,TEMP,0,-1,0,-1,YES ;GET INPUT
4896      TRAP        C$GMAN
4897      BR          10001$
4898      .WORD       TEMP
4899      .WORD       T$CODE
4900      .WORD       EQUQ
4901      .WORD       -1
4902      .WORD       T$LOLIM
4903      .WORD       T$HILIM
4904      MOV          TEMP,GLBPLS(R3) ;PUT ANSWER BACK
4905      ADD          #2,R3 ;BUMP R3
4906      BIT          #TRBB,DEVPAR ;IS THIS TRIB
4907      BEQ         DOGL4 ;BRANCH IF TRIB
4908
4909      DOGL2: CMP          #37,R2          ;ALL DONE
4910      BNE         DOGL1
4911
4912      DOGL4: MOV          (SP)+,R4        ;RESTORE R4,R3,R2
4913      MOV          (SP)+,R3
  
```

CZCLMBO DMP/V-11 DCLT MACY11 30A(1052) 28-JUL-81 13:35 PAGE 118 N 9
CZCLMB.P11 28-JUL-81 13:34 DO ALL GLOBAL PARMAS

SEQ 0117

4914 046156 012602
4915 046160 000207

MOV (SP)+,R2
RTS PC

;RETURN TO CALLING ROUTINE

4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931
4932
4933
4934
4935
4936
4937
4938
4939
4940
4941
4942
4943
4944
4945
4946
4947
4948
4949
4950
4951

.SBTTL QUEUE UP ALL REC BUFFERS FOR MULTIPOINT

```

: **
: FUNCTIONAL DESCRIPTION: QURXAL - QUEUE ALL REC BUFFERS
:
: THIS ROUTINE QUEUES ALL REC BUFFERS FOR VALID TRIBS
: IF MODE IS POINT TO POINT TRIB LIST WILL STILL BE ONE.

```

```

: SUBORDINATE ROUTINES USED:
:         GTVIND - LOADS INDEX WITH OFFSET TO NEXT
:                   VALID TRIB AND LOADS TRIBN WITH
:                   ADDRESS OF NEXT VALID TRIB
:         ULRPLS - MOVES RXPTR FOR THIS TRIB TO
:                   CPTRR FROM CPTRLS.
:         LOGAQR - QUES REC BUFFER POINTED TO BY
:                   CPTRR AND LOGS THIS IN EVENT LOG
:         LDRPLS - MOVES VALUE OF CPTRR TO SLOT IN
:                   CPTRLS FOR THIS TRIB

```

```

: CALLING SEQUENCE:
:         JSR     PC,RXQUAL
: --

```

```

4941 046162 012737 177777 015762 RXQUAL: MOV     #-1,INDEX      ;SET INDEX TO -1
4942 046170 004737 046460          RXQU1: JSR     PC,GTVIND   ;GET NEXT VALID INDEX
4943 046174 022737 000040 015762          CMP     #32,INDEX     ;IS ALL DONE
4944 046202 001412          BEQ     RXQU1        ;IF SO EXIT
4945 046204 004737 046524          JSR     PC,ULRPLS    ;LOAD CPTRR FOR THIS TRIB
4946 046210 052737 000004 017414          BIS     #QRX,FLAG    ;SET THE QRX,FLAG
4947 046216 004737 047300          JSR     PC,LOGAQR
4948 046222 004737 046504          JSR     PC,LDRPLS
4949 046226 000760          BR      RXQU1        ;RELOAD RX PTR LIST
4950 046230 000207          RXQU1: BR      RXQU1   ;AND THEN GO BACK FOR MORE
4951                                RXQU1: RTS     PC      ;RETURN TO CALLER

```


4952
4953
4954
4955
4956
4957
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982
4983
4984
4985
4986
4987

```
.SBTTL          LOAD CPTLS LIST INITIALLY

: **
: FUNCTIONAL DESCRIPTION:          LCPRLS -LOAD CPTR LIST INITIALLY
:
: THIS ROUTINE LOADS UP THE CPTLS LIST FOR ALL
: VALID TRIB ADDRESS IN THE TRIBLS IT ALSO LOADS
: THE DVRCLS LIST FOR MSG COUNTS.
:
: INPUTS:          RXMTOT - TOTAL NUMBER OF RX MSGS PER TRIB
:
: OUTPUTS:         CPTLS - LOADED WITH POINTERS TO THE RXPTR TABLE
:                   FOR EACH TRIB
:                   DVRCLS - LOADED WITH RXTOT COUNT FOR EACH TRIB
:
: SUBORDINATE ROUTINES USED:
:                   GTVIND - GETS NEXT VALID INDEX BY
:                           CHECKING TRIBLS FOR NON ZERO ENTRY
:                   LCPRL1 - LOADS POINTER TABLE FOR TRIB AT THIS
:                           INDEX VALUE AND RXMTOT TO DVRCLS FOR
:                           THIS TRIB.
:
: CALLING SEQUENCE:
:                   JSR          PC,LCPRLS
: --
```

```
4978 046232 012737 177777 015762 LCPRLS: MOV      #-1,INDEX      ;SET UP INDEX VALUE TO -1
4979 046240 004737 046460          LCPRL1: JSR      PC,GTVIND    ;GET VALID INDEX
4980 046244 022737 000040 015762          CMP      #32,INDEX    ;IS IT 32?
4981 046252 001403          BEQ      LCPREX      ;BRANCH IF 32.
4982 046254 004737 046264          JSR      PC,LCPRL1    ;IF NOT LOAD CPTLS FOR THIS TRIB.
4983 046260 000767          BR       LCPRL1      ;GO BACK FOR NEXT
4984 046262 000207          LCPREX: RTS     PC      ;RETURN TO CALLER WHEN DONE WITH ALL.
```

4988
4989
4990
4991
4992
4993
4994
4995
4996
4997
4998
4999
5000
5001
5002
5003
5004
5005
5006
5007
5008
5009
5010
5011
5012
5013
5014
5015
5016
5017
5018
5019
5020
5021

```
.SBTTL          LOAD CPTCLS AND DVRCLS FROM INDEX

: **
: FUNCTIONAL DESCRIPTION:          LCPRL1 - LOAD CPTCLS AND DVRCLS FROM INDEX
:
: THIS ROUTINE LOADS UP THE CPTCLS LIST FOR THE
: INDEX VALUE AND THE DVRCLS IS LOADED WITH RXMTOT.
:
: INPUTS:          RXMTOT - TOTAL NUMBER OF RX MSGS PER TRIB
:                 PTR23 - START OF RX POINTER TABLE
:
: OUTPUTS:         CPTCLS - LOADED WITH POINTERS TO THE RXPTR LIST
:                 DVRCLS - LOADED WITH RXMTOT COUNT
:
: SUBORDINATE ROUTINES USED:
:                 MTPLY - MULTIPLIES VALUE IN INDEX BY VALUE IN
:                       TEMP AND THEN ADDS THAT RESULT TO VALUE
:                       IN TEMP2 AND PUTS FINAL RESULT IN TEMP2
:
: CALLING SEQUENCE:
:                 JSR          PC,LCPRL1
: --
```

```
5011 046264 012737 011606 017354 LCPRL1: MOV      #PTR23,TEMP2      ;SET UP TEMP 2 AS BASE
5012 046272 013737 015762 017232      MOV      INDEX,MPLY      ;SET UP MULTIPLIER
5013 046300 012737 000074 017350      MOV      #60.,TEMP      ;SET UP MULTIPLICAN
5014 046306 004737 046434              JSR      PC,MTPLY        ;GO MULTIPY
5015 046312 013703 015762 LCPRL2: MOV      INDEX,R3
5016 046316 113763 017276 015612      MOV      RXMTOT,DVRCLS(R3) ;LOAD UP COUNT LIST
5017 046324 006303              ASL      R3              ;MAKE R3 WORD INDEX
5018 046326 013763 017354 015412      MOV      TEMP2,(CPTCLS(R3)) ;SET UP POINTER TABLE
5019 046334 000207              RTS      PC              ;RETURN TO CALLER
```

5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039
5040
5041
5042

.SBTTL CLEAR RECEIVE POINTER LIST
:++
: FUNCTIONAL DESCRIPTION: CLRPLS - CLEAR RX POINTER LIST
: THIS ROUTINE CLEARS ALL 32 SLOTS OF THE CTRLS
: OUTPUTS: CTRLS - IS ZEROED IN ALL SLOTS
: CALLING SEQUENCE:
: JSR PC,CLRPLS
:--

046336	012737	000040	017350	CLRPLS: MOV	#32,TEMP	
046344	012703	015412		MOV	#CTRLS,R3	;LOAD START OF LIST TO R3
046350	005023			CLRPL1: CLR	(R3)+	;CLEAR THIS SLOT
046352	005337	017350		DEC	TEMP	
046356	001374			BNE	CLRPL1	;IF NOT DONE GO BACK
046360	000207			CLRPEX: RTS	PC	;RETURN TO CALLER WHEN DONE

5043
5044
5045
5046
5047
5048
5049
5050
5051
5052
5053
5054
5055
5056
5057
5058
5059
5060
5061
5062
5063
5064
5065
5066
5067
5068
5069
5070
5071
5072
5073
5074
5075
5076
5077
5078
5079
5080
5081

```
.SBTTL          LOAD TX POINTER LIST INITIALLY

: **
: FUNCTIONAL DESCRIPTION:      LCPTLS - LOAD TRANSMIT POINTER LIST
:                             THIS ROUTINE LOADS CPTTLS WITH TX POINTERS
:                             FOR EACH VALID TRIB.
:
: INPUTS:
:     TXMTOT - TOTAL NUMBER OF TX MSGS
:     PTRTAB - POINTER TO TOP OF TX POINTER TABLE
:
: OUTPUTS:
:     CCTLS - LOADED WITH POINTERS TO TX POINTER TABLE
:             FOR ALL VALID TRIBS
:     DVTCLS - TX MSG COUNT LIST LOADED WITH MSG COUNTS
:             FOR ALL VALID TRIBS
:
: SUBORDINATE ROUTINES USED:
:     GTVIND - GETS NEXT VALID INDEX BY
:             CHECKING TRIBLS FOR NON ZERO ENTRY
:
:     LDTPLS - LOADS VALUE FROM CPTR TO CPTTLS INDEXED
:             BY TRIBN
:
:     LDTCLS - LOADS DVTCT TO DVTCLS INDEXED BY TRIBN
:
: CALLING SEQUENCE:
:     JSR     PC,LCPTLS
```

```
LCPTLS: MOV     TXMTOT,DVTCT      ;LOAD UP COUNT
        MOV     #PTRTAB,CPTR
        MOV     #-1,INDEX       ;LOAD INDEX WITH -1
LCPT1:  JSR     PC,GTVIND        ;GET VALID INDEX
        CMP     #32,INDEX       ;IS THIS THE END
        BEQ     LCPTX           ;EXIT IF SO
        JSR     PC,LDTPLS       ;LOAD TX POINTER LIST
        JSR     PC,LDTCLS       ;LOAD TX COUNT LIST
        BR      LCPT1           ;GO BACK
LCPTX:  RTS     PC              ;RETURN TO CALLER
```

```
046362 013737 017260 017256
046370 012737 011416 017340
046376 012737 177777 015762
046404 004737 046460
046410 022737 000040 015762
046416 001405
046420 004737 046642
046424 004737 046702
046430 000765
046432 000207
```

5082
 5083
 5084
 5085
 5086
 5087
 5088
 5089
 5090
 5091
 5092
 5093
 5094
 5095
 5096
 5097
 5098
 5099
 5100
 5101
 5102
 5103
 5104
 5105
 5106

.SBTTL MULTIPY

```

: **
: FUNCTIONAL DESCRIPTION: MTPLY- MULTIPY
: THIS ROUTINE MULTIPLIES THE VALUE IN MPLY BY
: THE VALUE IN TEMP AND THEN ADDS IN THE VALUE OF TEMP2
: WITH THE RELSUT GOING TO TEMP2
:
: INPUTS:      TEMP2 - INITIALLY VALUE
:              TEMP  - VALUE TO MULTIPLY BY
:              MPLY  - NUMBER OF TIMES TO MULITPLY
:
: OUTPUTS:     TEMP2 - RESULT OF [MPLY * TEMP]+TEMP2
:
: CALLING SEQUENCE:
:              JSR   PC,MTPLY
: --
  
```

```

046434 005737 017232
046440 001406
046442 063737 017350 017354
046450 005337 017232
046454 000767
046456 000207
  
```

```

MTPLY:  TST   MPLY
        BEQ   MTPLEX
        ADD   TEMP,TEMP2
        DEC   MPLY
        BR    MTPLY
MTPLEX: RTS   PC
        ;IF MULITPLIER IS ZERO QUIT
        ;ADD THE FACTOR TO BASE
        ;COUNT DOWN THE MULTIPLIER
        ;GO BACK FOR MORE
        ;RETURN TO CALLER
  
```

5107
5108
5109
5110
5111
5112
5113
5114
5115
5116
5117
5118
5119
5120
5121
5122
5123
5124
5125
5126
5127
5128
5129
5130

.SBTTL GET NEXT VALID INDEX

..**
: FUNCTIONAL DESCRIPTION: GTVIND - GET NEXT VALID INDEX
: THIS LOADS INDEX WITH INDEX VALUE OF NEXT VALID TRIB. THIS ALSO
: LOADS TRIBN WITH THE ADDRESS.
: TRIB BEING THE LOCATION IN THE TRIBLS THAT HAS A NON-ZERO
: ENTRY.

: INPUTS: INDEX - SET TO VALUE OF LAST INDEX
: OUTPUTS: INDEX - SET TO VALUE OF THIS TRIB
: TRIBN - ADDRESS OF THIS TRIB

: CALLING SEQUENCE:
: JSR PC,GTVIND
:--

GTVIND: MOV INDEX,R3
GTVI1: INC R3
MOVB TRIBLS(R3),TRIBN :LOAD TRIBN
BEQ GTVI1 :IF ZERO GO GET ANOTHER
MOV R3,INDEX :LOAD INDEX VALUE IF NOT ZERO
RTS PC :RETURN TO CALLER WHEN DONE

046460 013703 015762
046464 005203
046466 116337 015712 015756
046474 001773
046476 010337 015762
046502 000207

```
5131 .SBTTL          LOAD REC POINTER LIST
5132 :
5133 : **
5134 : FUNCTIONAL DESCRIPTION:      LDRPLS - LOAD RX POINTER LIST FROM CPTRR
5135 : THIS ROUTINE MOVES DATA FROM CPTRR TO THE SLOT IN THE
5136 : CPTRLS INDEXED BY INDW.
5137 : INPUTS:          INDW - WORD INDEX INTO LIST
5138 : OUTPUTS:         CPTRLS - CORRECT SLOT LOADED WITH DATA FROM CPTRR
5139 : SUBORDINATE ROUTINES USED:
5140 : GETIND -          GETS INDW FOR THIS TRIBN
5141 : CALLING SEQUENCE
5142 : JSR          PC,LDRPLS
5143 : --
```

```
5144 046504 004737 047152 LDRPLS: JSR          PC,GETIND          ;GET INDW FOR THIS TRIBN
5145 046510 013703 015760          MOV          INDW,R3          ;MOVE WORD INDEX TO R3
5146 046514 013763 017336 015412          MOV          CPTRR,CPTRLS(R3)      ;LOAD CPTRLS LIST
5147 046522 000207          RTS          PC          ;RETURN TO CALLER
```

```
5148 .SBTTL          UNLOAD CPTRR LIST
5149 :
5150 : **
5151 : FUNCTIONAL DESCRIPTION:      ULRPLS - UNLOAD RX POINTER LIST
5152 : THIS ROUTINE MOVES DATA FROM CPTRLS SLOT INDEXED
5153 : BY INDW TO CPTRR.
5154 : IMPLICIT INPUTS:
5155 : TRIBN -          ADDRESS OF CURRENT TRIB
5156 : OUTPUTS:         CPTRR -          VALUE FROM CPTRLS
5157 : SUBORDINATE ROUTINES USED:
5158 : GETIND -          GET INDW FOR THIS TRIBN
5159 : CALLING SEQUENCE:
5160 : JSR          PC,ULRPLS
5161 : --
```

```
5162 ULRPLS: JSR          PC,GETIND          ;GET INDEX
5163 046524 004737 047152          MOV          INDW,R3          ;MOVE WORD INDEX TO R3
5164 046530 013703 015760          MOV          CPTRLS(R3),CPTRR      ;LOAD CPTRR FROM LIST INDEX
5165 046534 016337 015412 017336          RTS          PC          ;RETURN TO CALLER
5166 046542 000207
```

5167
5168
5169
5170
5171
5172
5173
5174
5175
5176
5177
5178
5179
5180
5181
5182
5183
5184
5185
5186
5187
5188
5189

```
.SBTTL          GET REC POINTER TO CPTR

: **
: FUNCTIONAL DESCRIPTION:      GRPTCP - GET RX POINTER TO CPTR
: THIS ROUTINE GETS THE RX POINTER TO CPTR FOR USE IN BUILD
: BUFFER.
:
: INPUTS:          INDEX - INDEX VALUE FOR TRIB
:
: OUTPUTS:         CPTR - LOADED WITH ADDRESS OF RX BUFFER FOR THIS TRIB
: SUBORDINATE ROUTINES USED:
: MTPLY - MULTIPLIES INDEX BY TEMP AND ADDS TEMP2 TO RESULT
: CALLING SEQUENCE:
: JSR          PC,GRPTCP
: --

GRPTCP: MOV      INDEX,MPLY      ;SET UP MULPILIER
        MOV      #60.,TEMP
        MOV      RXPTR,TEMP2
        JSR      PC,MTPLY      ;[INDEX VALUE X 60.] + RXPTR = POINTER ADDRESS
        MOV      TEMP2,CPTR    ;SET UP POINTER ADDR.
        RTS      PC
```



```

5190 .SBTTL          LOAD DVRCT LIST
5191 :++
5192 : FUNCTIONAL DESCRIPTION:  LDRCLS - LOAD RX COUNT LIST
5193 : THIS ROUTINE LOADS THE VALUE FROM DVRCT TO
5194 : THE SLOT IN DVRCLS INDEXED BY TRIBN
5195 : INPUTS:  TRIBN - ADDRESS OF TRIB IN USE
5196 :          DRVCT - COUNT VALUE TO GO TO LIST
5197 : OUTPUTS: DVRCLS- VALUE OF DRVCT
5198 : SUBORDINATE ROUTINES USED:
5199 :          GETIND - GET INDEX FROM TRIBLS
5200 : CALLING SEQUENCE:
5201 :          JSR    PC,LDRCLS
5202 :--
5203
5204 046602 004737 047152 LDRCLS: JSR    PC,GETIND          ;GET INDEX
5205 046606 013703 015762      MOV    INDEX,R3          ;LOAD R3 WITH BYTE INDEX
5206 046612 113763 017274 015612      MOVB  DVRCT,DVRCLS(R3)      ;LOAD LIST WITH COUNT
5207 046620 000207          RTS    PC          ;RETURN TO CALLER
5208
5209 .SBTTL          UNLOAD DVRCT LIST
5210 :++
5211 : FUNCTIONAL DESCRIPTION:  ULRCLS - UNLOAD RX COUNT LIST
5212 : THIS ROUTINE UNLOADS THE VALUE TO DVRCT FROM
5213 : THE SLOT IN DVRCLS INDEXED BY TRIBN
5214 : INPUTS:  TRIBN - ADDRESS OF TRIB IN USE
5215 :          DVRCLS- VALUE OF DRVCT
5216 : OUTPUTS:
5217 :          DRVCT - COUNT VALUE FROM LIST
5218 : SUBORDINATE ROUTINES USED:
5219 :          GETIND - GET INDEX FROM TRIBLS
5220 : CALLING SEQUENCE:
5221 :          JSR    PC,ULRCLS
5222 :--
5223
5224 046622 004737 047152 ULRCLS: JSR    PC,GETIND          ;GET INDEX
5225 046626 013703 015762      MOV    INDEX,R3          ;MOVE INDEX TO R3
5226 046632 116337 015612 017274      MOVB  DVRCLS(R3),DVRCT      ;UNLOAD LIST
5227 046640 000207          RTS    PC          ;RETURN TO CALLER
  
```

```

5228      .SBTTL          LOAD CPTR LIST (TRANSMIT POINTER)
5229
5230      : **
5231      : FUNCTIONAL DESCRIPTION:      LDTPLS - LOAD TX POINTER LIST
5232      : THIS ROUTINE LOADS THE VALUE FROM CPTR TO
5233      : THE TX POINTER LIST INDEXED BY TRIBN INDEX.
5234      : INPUTS:          TRIBN - ADDRESS OF TRIB IN USE
5235      : OUTPUTS:         CPTTLS - SLOT LOADED WITH CPTR DATA
5236      : SUBORDINATE ROUTINES USED:
5237      : GETIND -          GET INDEX VALUE FROM TRIBLS
5238      : CALLING SEQUENCE:
5239      : JSR          PC,LDTPLS
5240      : --
5241
5242      046642 004737 047152      LDTPLS: JSR          PC,GETIND          :GET INDEX
5243      046646 013703 015760      MOV          INDW,R3          :MOVE INDEX TO R3
5244      046652 013763 017340 015512  MOV          CPTR,CPTTLS(R3) :LOAD LIST
5245      046660 000207              RTS          PC              :RETURN TO CALLER
5246
5247      .SBTTL          UNLOAD CPTR LIST (TRANSMIT POINTER)
5248
5249      : **
5250      : FUNCTIONAL DESCRIPTION:      ULTPLS - UNLOAD TX POINTER LIST
5251      : THIS ROUTINE MOVES DATA FROM TX POINTER LIST
5252      : TO CPTR.
5253      : INPUTS:          TRIBN - ADDRESS OF TRIB IN USE
5254      : OUTPUTS:         CPTR - VALUE FROM THE TX POINTER LIST
5255      : SUBORDINATE ROUTINES USED:
5256      : GETIND -          GET INDEX FROM TRIBLS
5257      : CALLING SEQUENCE:
5258      : JSR          PC,ULTPLS
5259      : --
5260
5261      046662 004737 047152      ULTPLS: JSR          PC,GETIND          :GET INDEX
5262      046666 013703 015760      MOV          INDW,R3          :MOVE WORD INDEX TO R3
5263      046672 016337 015512 017340  MOV          CPTTLS(R3),CPTR :GET PTR FROM LIST
5264      046700 000207              RTS          PC              :RETURN TO CALLER
5265
  
```

5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284
5285
5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305

```
.SBTTL          LOAD DVTCT LIST (TRANSMIT COUNT)
:++
: FUNCTIONAL DESCRIPTION:      LDTCLS - LOAD TX COUNT LIST
:                             THIS ROUTINE LOADS A VALUE FROM DVTCT TO
:                             THE TX COUNT LIST (DVTCLS). INDEXED BY TRIBN.
:
: INPUTS:          TRIBN      -      ADDRESS OF TRIB IN USE
:                 DVTCT      -      CURRENT TX COUNT FOR TRIB
: OUTPUTS:        DVTCLS     -      SLOT LOADED WITH DVTCT
: SUBORDINATE ROUTINES USED:
:                 GETIND     -      GET INDEX FROM TRIBLS
: CALLING SEQUENCE:
:                 JSR        PC,LDTCLS
:--
```

```
LDTCLS: JSR      PC,GETIND      ;GET INDEX
        MOV      INDEX,R3      ;MOVE BYTE INDEX TO R3
        MOVB    DVTCT,DVTCLS(R3);LOAD LIST
        RTS     PC              ;RETURN TO CALLER
```

```
.SBTTL          UNLOAD DVTCT LIST (TX COUNT)
:++
: FUNCTIONAL DESCRIPTION:      ULTCLS - UNLOAD TX COUNT LIST
:                             THIS ROUTINE TAKES DATA FROM DVTCLS AND MOVES
:                             IT TO DVTCT
: INPUTS:          TRIBN      -      ADDRESS OF TRIBN IN USE
:                 DVTCT      -      VLAUE
: OUTPUTS:
: SUBORDINATE ROUTINES USED:
:                 GETIND     -      GET INDEX VALUE FROM TRIBLS
: CALLING SEQUENCE:
:                 JSR        PC,ULTCLS
:--
```

```
ULTCLS: JSR      PC,GETIND      ;GET INDEX
        MOV      INDEX,R3      ;MOVE BYTE INDEX TO R3
        MOVB    DVTCLS(R3),DVTCT
        RTS     PC              ;RETURN TO CALLER
```

```

5306 .SBTTL GET ALL RX POINTERS FROM LIST TO CPTRR
5307
5308
5309 :++
5310 : FUNCTIONAL DESCRIPTION: GARPFL - GET ALL RX POINTERS FROM LIST
5311 : THIS ROUTINE CHECKS ALL RX POINTERS FOR VALID TRIBS
5312 : IN CPTRLS AND MAKES SURE THEY ARE ALL ZERO.
5313 : OUTPUTS: CPTRR - ZERO IF ALL CPTRLS IS ZERO
5314 : NON ZERO IF NOT.
5315 :
5316 : SUBORDINATE ROUTINES USED:
5317 : GTVIND - GET VALID INDEX
5318 : ULRPLS - UNLOAD CPTRR LIST TO CPTRR
5319 : CALLING SEQUENCE:
5320 : JSR PC,GARPFL
5321 :--
5322 046742 013737 015756 017362 GARPFL: MOV TRIBN,TEMP5
5323 046750 012737 177777 015762 MOV #-1,INDEX
5324 046756 004737 046460 GARP1: JSR PC,GTVIND :GET VALID INDEX
5325 046762 022737 000040 015762 CMP #32,INDEX :COMPARE INDEX
5326 046770 001405 BEQ GARPEX :EXIT IF DONE
5327 046772 004737 046524 JSR PC,ULRPLS :LOAD CPTRR WITH VALUE
5328 046776 005737 017336 TST CPTRR :TEST THE VALUE
5329 047002 001765 BEQ GARP1 :IF ZERO CHECK NEXT
5330 047004 013737 017362 015756 GARPEX: MOV TEMPS,TRIBN
5331 047012 000207 RTS PC :RETURN TO CALLER WHEN DONE
5332

```

5333
5334
5335
5336
5337
5338
5339
5340
5341
5342
5343
5344
5345
5346
5347
5348
5349
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359

.SBTTL GET ALL TX COUNTS FROM LIST TO DVTCT
:RETURN WITH DVTCT=1 IF ANY COUNT HAS SOME IN IT
:IF ALL COUNTS ARE ZERO EXIT
:
:++
: FUNCTIONAL DESCRIPTION: GATCFL - GET ALL TX COUNTS FROM LIST
: THIS ROUTINE GETS AND CHECKS ALL TX COUNTS TO BE ZERO
: OUTPUTS: DVTCT - ZERO IF LIST IS ZERO
: NON ZERO IF NOT
: SUBORDINATE ROUTINES USED:
: GTVIND - GET NEXT VALID INDEX
: CALLING SEQUENCE:
: JSR PC,GATCFL
:--

GATCFL: MOV TRIBN,TEMP5
MOV #-1,INDEX
CLR DVTCT ;CLEAR COUNT
GATC1: JSR PC,GTVIND ;GET VALID INDEX
CMP #32,INDEX ;IS INDEX =32 ALL DONE
BEQ GATCEX ;IF SO EXIT
MOV INDEX,R3
TSTB DVTCLS(R3) ;IS THIS COUNT 0
BEQ GATC1 ;IF THIS ONE IS ZERO
5347 047014 013737 015756 017362 GATCFL: MOV TRIBN,TEMP5
5348 047022 012737 177777 015762 MOV #-1,INDEX
5349 047030 005037 017256 CLR DVTCT ;CLEAR COUNT
5350 047034 004737 046460 GATC1: JSR PC,GTVIND ;GET VALID INDEX
5351 047040 022737 000040 015762 CMP #32,INDEX ;IS INDEX =32 ALL DONE
5352 047046 001410 BEQ GATCEX ;IF SO EXIT
5353 047050 013703 015762 MOV INDEX,R3
5354 047054 105763 015652 TSTB DVTCLS(R3) ;IS THIS COUNT 0
5355 047060 001765 BEQ GATC1 ;IF THIS ONE IS ZERO
5356 047062 012737 000001 017256 GATCFL: MOV TRIBN,TEMP5
5357 047070 013737 017362 015756 GATCEX: MOV TEMP5,TRIBN ;LOAD COUNT WITH A 1
5358 047076 000207 RTS PC ;RETURN TO CALLERR
5359

5360 .SBTTL GET NEXT TX POINTER FROM LIST

5361
 5362 :++
 5363 : FUNCTIONAL DESCRIPTION: GNTXPR - GET NEXT TX POINTER
 5364 : THIS ROUTINE GETS THE NEXT TX POINTER TO CPTR
 5365 : OUTPUTS: CPTR - POINTER FOR NEXT TRANSMIT MMSG
 5366 : SUBORDINATE ROUTINES USED:
 5367 : GTVIND - GET VALID INDEX
 5368 : CALLING SEQUENCE:
 5369 : JSR PC,GNTXPR
 5370 :--

5371 047100 022737 000040 015762 GNTXPR: CMP #32.,INDEX ;IS INDEX = DONE
 5372 047106 001003 BNE GNTX1
 5373 047110 012737 177777 015762 GNTX2: MOV #-1,INDEX
 5374 047116 004737 046460 GNTX1: JSR PC,GTVIND
 5375 047122 022737 000040 015762 CMP #32.,INDEX
 5376 047130 001767 BEQ GNTX2
 5377 047132 004737 046722 JSR PC,ULTCLS ;GET COUNT FROM LIST
 5378 047136 005737 017256 TST DVTCT ;TEST COUNT
 5379 047142 001756 BEQ GNTXPR
 5380 047144 004737 046662 JSR PC,ULTPLS ;UNLOAD POINTER
 5381 047150 000207 RTS ;RETURN TO CALLER
 5382

5383 .SBTTL GET INDEX BYTE AND WORD

5384 :++
 5385 : FUNCTIONAL DESCRIPTION: GETIND - GET INDEX FOR WORD AND BYTE
 5386 : THIS ROUTINE GETS INDEX LOADED WITH INDEX AND INDW WITH INDEX
 5387 : FOR WORD. IF TRIBLS ENTRY IS EQUAL TO TRIBN
 5388 : OUTPUTS: INDEX - BYTE INDEX
 5389 : INDW - WORD INDEX
 5390 : CALLING SEQUENCE:
 5391 : JSR PC,GETIND
 5392 :--
 5393

5394
 5395 047152 012703 177777 GETIND: MOV #-1,R3 ;LOAD R3 WITH -1
 5396 047156 005203 GETI1: INC R3 ;BUMP R3
 5397 047160 022703 000040 CMP #32.,R3 ;ARE WE ALL DONE
 5398 047164 001772 BEQ GETIND ;IF SO GO BACK
 5399 047166 126337 015712 015756 CMPB TRIBLS(R3),TRIBN ;ELSE COMPARE FOR THIS TRIB
 5400 047174 001370 BNE GETI1 ;BRANCH IF NO MATCH
 5401 047176 010337 015762 GETI2: MOV R3,INDEX ;STORE OFF BYTE INDEX
 5402 047202 006303 ASL R3 ;MAKE UP WORD INDEX
 5403 047204 010337 015760 MOV R3,INDW ;STORE OFF WORD INDEX
 5404 047210 000207 RTS ;RETURN TO CALLER

5405
 5406
 5407
 5408
 5409
 5410
 5411
 5412
 5413
 5414
 5415
 5416
 5417
 5418
 5419 047212 010246
 5420 047214 010346
 5421 047216 010446
 5422
 5423 047220 012704 000040
 5424 047224 012703 016220
 5425 047230 012702 016166
 5426 047234 012223
 5427 047236 022702 016206
 5428 047242 001374
 5429 047244 005304
 5430 047246 001370
 5431
 5432
 5433
 5434 047250 012703 017220
 5435 047254 012702 016206
 5436 047260 012223
 5437 047262 022702 016220
 5438 047266 001374
 5439 047270 012604
 5440 047272 012603
 5441 047274 012602
 5442 047276 000207
 5443
 5444

```

.SBTTL          WRITE DEFAULTS TO TRIB AND GLOBAL SLOTS
:++
: FUNCTIONAL DESCRIPTION:      WRDEFP - WRITE  DEFAULT POLL PARAMETERS
:
: THIS ROUTINE WRITES ALL POLLIS WITH DEFAULTS AND ALSO
: WRITE S THE GLOBAL LIST WITH DEFAULTS
: INPUTS:
:
: CALLING SEQUENCE:
:--
          ;WRITE DEFAULT POLL PARMS FOR TRIBS
WRDEFP: MOV     R2,-(SP)          ;SAVE R2,R3,R4 ON THE STACK
        MOV     R3,-(SP)
        MOV     R4,-(SP)
        MOV     #32,R4
        MOV     #POLLIS,R3
WRDE5B: MOV     #POLDEF,R2
WRDE5A: MOV     (R2)+,(R3)+
        CMP     #GLBDEF,R2      ;ARE WE THRU ONE SET?
        BNE    WRDE5A
        DEC     R4
        BNE    WRDE5B

          ;WRITE DEFAULTS FOR GLOBAL
WRDE5D: MOV     #GLBPLS,R3
        MOV     #GLBDEF,R2
        MOV     (R2)+,(R3)+
        CMP     #GLBEND,R2
        BNE    WRDE5D
        MOV     (SP)+,R4      ;RESTORE R4,R3,R2
        MOV     (SP)+,R3
        MOV     (SP)+,R2
        RTS     PC           ;RETURN TO CALLING ROUTINE
  
```

5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457 047300 013702 017336
5458 047304 011237 017354
5459 047310 012237 017270
5460 047314 011237 017356
5461 047320 011237 017272
5462 047324 010237 017336
5463 047330 004737 063736
5464 047334 004737 042054
5465 047340 000207
5466

```
.SBTTL          LOG AND QUE REC BUFFERS
:++
: FUNCTIONAL DESCRIPTION:      LOGAQR - QUE AND LOG RX BUFFERS
:                               THIS ROUTINE QUEUES THE REC BUFFER POINTED TO BY
:                               CPTRR
: INPUTS:                      CPTRR - POINTS TO POINTER TABLE ENTRY
: IMPLICIT OUTPUTS:           BUFFER QUEUED FOR THIS ENTRY
: CALLING SEQUENCE:           JSR      PC,LOGAQR
:--

LOGAQR: MOV      CPTRR,R2          ;LOAD R2 FROM POINTER
        MOV      (R2),TEMP2       ;SET UP ADDRESS FOR LOGGING
        MOV      (R2)+,DVRXA      ;SET UP ADDRESS FOR DEVICE
        MOV      (R2),TEMP3       ;SET UP CHAR COUNT FOR LOGGING
        MOV      (R2),DVRCC       ;SET UP COUNT FOR DEVICE
        MOV      R2,CPTRR         ;RESTORE POINTER
        JSR      PC,DVRXQ         ;QUEUE REC BUFFER
        JSR      PC,LOGRXQ        ;LOG RXQ
        RTS      PC              ;RETURN TO CALLER
```


5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487

.SBTTL SHOW MODE OF OPERATION, LOOP TYPE AND QUALIFIERS

FUNCTIONAL DESCRIPTION:
SHWOP - SHOW MODE OF OPERATION, LOOP, QUALIFIERS
PRINTED ON THE OPERATOR'S CONSOLE.

INPUTS:
DEV1= MODE TYPE (MODTYP)
DEV2= MAINT LOOP TYPE (MLTYP)
DEV3= 'RUN PASS' COUNT (RPASS) - COUNT DOWN
DEV4= PARAMETERS WORD (PARAM)

IMPLICIT INPUTS:
MODES= TABLE OF ADDRESSES OF MODE NAME STRINGS
LOOPS= TABLE OF ADDRESSES OF LOOP TYPE NAMES

CALLING SEQUENCE:
JSR PC,SHWOP

5488
5489
5490
5491
5492
5493
5494
5495
5496
5497
5498
5499
5500
5501
5502
5503
5504
5505
5506
5507
5508
5509
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522

047342 013702 020650
047346 006302
047350 016237 003344 017350
047356 013702 020652
047362 006302
047364 012737 026764 017356
047372 005702
047374 001003
047376 012737 026763 017356
047404 016237 003362 017352
047412 013737 020654 017354
047420
047420 013746 017354
047424 013746 017352
047430 013746 017356
047434 013746 017350
047440 012746 027527
047444 012746 000005
047450 010600
047452 104416
047454 062706 000014
047460 005002
047462 012737 027043 017350
047470 032737 000001 020656
047476 001003
047500 012737 027041 017350
047506 012737 027054 017352
047514 032737 000002 020656
047522 001003
047524 012737 027052 017352
047532 012737 027064 017354
047540 032737 000004 020656
047546 001003
047550 012737 027062 017354

SHWOP: MOV DEV1,R2 ;GET THE MODE TYPE IN R2
 ASL R2 ;MAKE IT A WORD TABLE OFFSET
 MOV MODES(R2),TEMP ;GET ADDRESS OF MODE-IN-ASCII
 MOV DEV2,R2 ;GET MAINTENANCE LOOP TYPE
 ASL R2
 MOV #LP00,TEMP3 ;LOAD TEMP3 TO POINT TO '/LOOP='
 TST R2 ;SEE IF /LOOP=XXXXX OR NONE
 BNE 10\$;BR IF /LOOP= OF SOME KIND
 MOV #LP0,TEMP3 ;IF NO LOOP THEN DON'T PRINT '/LOOP='
 10\$: MOV LOOPS(R2),TEMP1 ;GET ADDRESS OF LOOP-IN-ASCII
 MOV DEV3,TEMP2 ;GET NUMBER OF PASSES
 PRINTS #SHF0,TEMP,TEMP3,TEMP1,TEMP2
 MOV TEMP2,-(SP)
 MOV TEMP1,-(SP)
 MOV TEMP3,-(SP)
 MOV TEMP,-(SP)
 MOV #SHF0,-(SP)
 MOV #5,-(SP)
 MOV SP,R0
 TRAP C\$PNTS
 ADD #14,SP
 CLR R2 ;NOW SET UP FOR QUALIFIERS IN ASCII
 MOV #PST,TEMP
 BIT #STATB,DEV4 ;SEE IF /STATUS OR /NOSTATUS
 BNE 1\$;BR IF /STATUS
 MOV #PNST,TEMP
 MOV #PCK,TEMP1
 BIT #DATCKB,DEV4 ;SEE IF /CHECK OR /NOCHECK
 BNE 2\$;BR IF /CHECK
 MOV #PNCK,TEMP1
 MOV #PEC,TEMP2
 BIT #ECHOB,DEV4 ;SEE IF /ECHO OR /NOECHO
 BNE 3\$;BR IF /ECHO
 MOV #PNEC,TEMP2

SHOW MODE OF OPERATION, LOOP TYPE AND QUALIFIERS

```

5523
5524
5525 047556 012737 027073 017362 3$: MOV #PMS,TEMP5
5526 047564 032737 000010 020656 BIT #MOCHK,DEV4 ;SEE IF MODEM OR /NOMODEM
5527 047572 001003 BNE 5$ ;BRANCH IF MODEM
5528 047574 012737 027071 017362 MOV #PNMS,TEMP5
5529
5530 047602 5$: PRINTS #SHF1,TEMP,TEMP1,TEMP2,TEMP5 ;,TEMP3,TEMP4 **RFU**
5531 047602 013746 017362 MOV TEMP5,-(SP)
5532 047606 013746 017354 MOV TEMP2,-(SP)
5533 047612 013746 017352 MOV TEMP1,-(SP)
5534 047616 013746 017350 MOV TEMP,-(SP)
5535 047622 012746 027565 MOV #SHF1,-(SP)
5536 047626 012746 000005 MOV #5,-(SP)
5537 047632 010600 MOV SP,R0
5538 047634 104416 TRAP C$PNTS
5539 047636 062706 000014 ADD #14,SP
5540 047642 000207 RTS PC ;RETURN
5541
5542

```

```

5543      .SBTTL          TRAVERSE COMMAND LINE SUBROUTINES
5544
5545      :++
5546      :               P$TRV SUBROUTINE
5547      :
5548      : PARSE THE COMMAND LINE SUBROUTINE
5549      : TAKE ACTIONS (VIA ACTION TREE) AS PARCING LINE
5550      : PARSING DIRECTIONS FROM 'CLI PARSING NODES'
5551      : REGS USED:
5552      :
5553      :               R1,R5=SCRATCH                      P$NUM=NUMERIC CODE FROM DATA
5554      :               R2=ACTION CODE PARAMETER FROM TREE
5555      :               R3=PARSE TREE POINTER
5556      :               R4=INPUT STRING POINTER
5557      : CALLING SEQUENCE:
5558      :               JSR      PC,P$TRV
5559      : --
5560
5561      047644  -          P$TRV:
5562      047644  013704  003374      MOV      P$BUFA,R4
5563      047650  013703  003376      MOV      P$TREE,R3
5564      047654  105714          P$TR5:  TSTB      (R4)                ;SEE IF ANY CHARS LEFT IN INPUT STRING
5565      047656  001441          BEQ      P$EXIT                ;BR IF NO
5566      047660  121327  000013      CMPB      (R3),#11.          ;SEE IF SPECIAL CLI CHAR CODE OR ASCII
5567      047664  003023          BGT      20$                ;BR IF REGULAR ASCII CHAR.
5568      047666  111305          MOVB      (R3),R5            ;GET SPECIAL CHAR CODE INTO R5
5569      047670  006305          ASL      R5
5570      047672  016505  047706      MOV      10$(R5),R5         ;BUILD TRAVERSE ROUTINE ADDRESS
5571      047676  062705  047706      ADD      #10$,R5
5572      047702  004715          JSR      PC,(R5)            ;JSR TO SPECIAL CLI TRAVERSE ROUTINE
5573      047704  000700          BR       P$TR5             ;GO SEE IF MORE OF STRING LEFT
5574
5575
5576      047706  000114          10$:   .WORD    TRVERR-10$      ;TRAVERSE TABLE FOR 'CLI FUNCTIONS'
5577      047710  000134          .WORD    TRVEXI-10$         ;1
5578      047712  000152          .WORD    TRVBR-10$         ;2
5579      047714  000162          .WORD    TRVBIF-10$        ;3
5580      047716  000204          .WORD    TRVSPA-10$        ;4
5581      047720  000270          .WORD    TRVNUM-10$        ;5
5582      047722  000604          .WORD    TRVALP-10$        ;6
5583      047724  000650          .WORD    TRVALN-10$        ;7
5584      047726  000270          .WORD    TRVOCT-10$        ;8
5585      047730  000256          .WORD    TRVDEC-10$        ;9
5586      047732  000736          .WORD    TRVSTR-10$       ;10
5587
5588      ;NOT A SPECIAL CODE
5589
5590      047734  121314          20$:   CMPB      (R3),(R4)        ;SEE IF FIRST CHAR OF STRING IS A MATCH
5591      047736  001403          BEQ      22$                ;BR IF A MATCH
5592      047740  004737  050004          JSR      PC,TRVBRC          ;IF NOT A MATCH, GO TAKE MISS BRANCH
5593      047744  000743          BR       P$TR5             ; THEN GO BACK PT'G TO MISS NODE
5594      047746  004737  047764          22$:   JSR      PC,TRVACT          ;IF A MATCH, GO DO ACTION DEFINED BY
5595      047752  062703  000004          ADD      #4,R3             ; ACTION CODE IN CLI NODE, THEN
5596      ; ADJUST PTR TO NEXT CLI NODE
5597      047756  005204          INC      R4                ;ADJUST BUF PTR TO NEXT CHAR IF MATCH
5598      047760  000735          BR       P$TR5

```

```

5599
5600 047762 000207 P$EXIT: RTS PC ;RETURN FROM PARSER
5601
5602 -----
5603
5604 ;GOTO USER ACTION ROUTINE
5605 047764 116302 000001 TRVACT: MOV 1(R3),R2 ;GET ACTION CODE FROM CLI NODE
5606 047770 042702 177400 BIC #177400,R2 ;CLEAR ANY SIGN EXTENSION
5607 047774 013705 003400 MOV P$ACT,R5 ;GET ADDRESS OF CLI ACTION ROUTINE
5608 050000 004715 JSR PC,(R5) ;GO DO ACTION DEFINED BY CODE
5609 050002 000207 RTS PC ;RETURN TO CALLING CODE
5610
5611 ;TAKE BRANCH IN TREE
5612 050004 016305 000002 TRVBRC: MOV 2(R3),R5 ;GET BRANCH DISPLACEMENT FROM TREE
5613 050010 060503 ADD R5,R3 ;AND POINT R3 TO THE 'MISS' NODE
5614 050012 000207 RTS PC ;RETURN TO P$TRV
5615
5616 ;NO BRANCH TAKEN
5617 050014 062703 000004 TRVNOB: ADD #4,R3 ;THINGS OK, UPDATE R3 TO POINT TO NEXT
5618 050020 000207 RTS PC ;NODE AND RETURN TO P$TRV
5619
5620 -----
5621 050022 004737 047764 TRVERR: JSR PC,TRVACT ;TAKE ERROR ACTION
5622 050026 112737 177777 003411 MOVB #-1,P$GDBD ;SET ERROR RETURN FLAG
5623 050034 005726 TST (SP)+ ;GET RID OF 'JSR PUSH TO TRVERR'
5624 050036 000137 047762 JMP P$EXIT ;RETURN DIRECT TO EXIT OF P$TRV ROUTINE
5625
5626 050042 004737 047764 TRVEXI: JSR PC,TRVACT ;TAKE EXIT ACTION
5627 050046 105037 003411 CLRB P$GDBD ;SET GOOD/BAD FLAG TO 'SUCCESS (0)''
5628 050052 005726 TST (SP)+ ;GET RID OF 'JSR PUSH TO TRVEXI''
5629 050054 000137 047762 JMP P$EXIT ;RETURN DIRECT TO EXIT OF P$TRV ROUTINE
5630
5631 050060 004737 047764 TRVBR: JSR PC,TRVACT ;GO TAKE BRANCH ACTION
5632 050064 000137 050004 JMP TRVBRC
5633
5634 050070 004737 047764 TRVBIF: JSR PC,TRVACT
5635 050074 105737 003411 TSTB P$GDBD ;SEE IF P$GDBD SET OR CLEARED BY ACTION
5636 050100 001402 BEQ 1$ ;IF CLEAR FALL THRU TO NEXT NODE
5637 050102 000137 050004 JMP TRVBRC ;ELSE TAKE THE 'MISS' BRANCH
5638 050106 000137 050014 1$: JMP TRVNOB ;JUST UPDATE TO NEXT NODE IF THINGS OK
5639
5640 050112 005005 TRVSPA: CLR R5 ;CLEAR 'SPACE OR TAB FOUND' FLAG
5641 050114 121427 000011 1$: CMPB (R4),#11 ;SEE IF CHAR. IN CMD LINE= TAB
5642 050120 001003 BNE 2$ ;BR IF NO, NOT A TAB
5643 050122 005204 INC R4 ;INC INPUT STRING POINTER
5644 050124 005205 INC R5 ;INDICATE A TAB FOUND
5645 050126 000772 BR 1$ ;GO CHECK NEXT CHAR
5646
5647 050130 121427 000040 2$: CMPB (R4),#40 ;SEE IF CHAR. IN CMD LINE= SPACE
5648 050134 001003 BNE 10$ ;BR IF NO, NON-SPACE OR NON-TAB CHAR.
5649 050136 005204 INC R4 ;INC INPUT STRING POINTER
5650 050140 005205 INC R5 ;INDICATE A SPACE FOUND
5651 050142 000764 BR 1$ ;GO CHECK NEXT CHAR
5652 050144 005705 10$: TST R5 ;SEE IF ANY SPACES OR TABS FOUND
5653 050146 001404 BEQ 15$ ;BR IF NO, TAKE NO ACTION
5654 050150 004737 047764 JSR PC,TRVACT ;GO TAKE ACTION IF ANY FOUND
    
```

```

5655 050154 000137 050014          JMP      TRVNOB          ;JUST GO UPDATE R3 TO NEXT NODE IF OK
5656 050160 000137 050004          15$:    JMP      TRVBRC          ;TAKE BRANCH (MISS) IF NONE FOUND
5657
5658
5659 050164 012737 000012 003406 TRVDEC: MOV      #10.,P$RADX      ;USE DECIMAL AS RADIX AND ASSUME +
5660 050172 000137 050204          JMP      TRVNMA
5661 050176          TRVOCT: ;(SAME AS TRVNUM SINCE DEFAULT RADIX IS OCTAL)
5662 050176 012737 000010 003406 TRVNUM: MOV      #8.,P$RADX      ;USE OCTAL AS RADIX AND ASSUME +
5663 050204 005005          TRVNMA: CLR      R5          ;CLEAR DIGIT COUNTER
5664 050206 121427 000053          CMPB    (R4),#'+'          ;SEE IF THERE'S A + SIGN THERE
5665 050212 001001          BNE     10$              ; BR IF NO
5666 050214 000406          BR      11$              ; ELSE P$RADX ALREADY SAYS +, JUST BR
5667 050216 121427 000055          10$:    CMPB    (R4),#'-'          ;SEE IF THERE'S A - SIGN THERE
5668 050222 001004          BNE     1$                ; BR IF NO
5669 050224 112737 177777 003407 MOVB    #-1,P$RADX+1      ;SET 'MINUS FLAG' (HI BYTE OF P$RADX)
5670 050232 005204          11$:    INC      R4          ;BUMP R4 TO POINT TO FIRST CHAR
5671
5672 050234 121427 000060          1$:    CMPB    (R4),#60          ;SEE IF CHAR. LESS THAN A '0'
5673 050240 002434          BLT     2$                ;BR IF YES (NOT NUMERIC)
5674 050242 121427 000067          CMPB    (R4),#67          ;SEE IF CHAR. GREATER THAN A '7'
5675 050246 003426          BLE     13$              ; BR IF YES
5676 050250 123727 003406 000012 CMPB    P$RADX,#10.        ;SEE IF IN DECIMAL MODE
5677 050256 001417          BEQ     12$              ; BR IF YES (CAN USE HIGHER LIMIT)
5678 050260 121427 000071          CMPB    (R4),#71          ;SEE IF DIGIT WAS A 8 OR 9
5679 050264 003022          BGT     2$                ;BR IF NON-NUMERIC
5680 050266          PRINTF #CLIBRX          ;ELSE WAS A 8 OR 9 WHEN IN OCTAL RADIX
5681 050266 012746 023455          MOV     #CLIBRX,-(SP)
5682 050272 012746 000001          MOV     #1,-(SP)
5683 050276 010600          MOV     SP,R0
5684 050300 104417          TRAP   C$PNTF
5685 050302 062706 000004          ADD     #4,SP
5686 050306 112737 177777 003411 MOVB    #-1,P$GDBD          ;SET ERROR RETURN FLAG
5687 050314 000474          BR      5$                ; PRINT ERROR AND TAKE MISS
5688
5689 050316 121427 000071          12$:    CMPB    (R4),#71          ;SEE IF CHAR. GREATER THAN A '9'
5690 050322 003003          BGT     2$                ;BR IF YES (NOT NUMERIC)
5691 050324 005204          13$:    INC     R4          ;UPDATE CMD LINE PTR TO NEXT CHAR.
5692 050326 005205          INC     R5          ;INDICATE A NUMERIC FOUND
5693 050330 000741          BR      1$                ;GO LOOK AT NEXT CHAR.
5694
5695 050332 005705          2$:    TST     R5          ;SEE IF FOUND ANY NUMERICS
5696 050334 001464          BEQ     5$                ;BR IF NO, TAKE 'MISS' BRANCH
5697 050336 010401          MOV     R4,R1          ;GET POINTER TO START OF NUMERIC STRING
5698 050340 160501          SUB     R5,R1
5699 050342 005037 003404          CLR     P$NUM          ;CLEAR LOC. WHERE VALUE WILL BE STORED
5700 050346 112102          3$:    MOVB    (R1)+,R2          ;GET ASCII CHAR AND CONVERT IT TO A #
5701 050350 162702 000060          SUB     #60,R2
5702 050354 006337 003404          ASL     P$NUM          ;SHIFT CURRENT VALUE TO MAKE ROOM
5703 050360 103437          BCS     7$                ;ERROR IF NUMBER TOO BIG
5704 050362 013737 003404 003402 MOV     P$NUM,P$CNT      ;SAVE FOR LATER IN CASE DECIMAL RADIX
5705 050370 006337 003404          ASL     P$NUM
5706 050374 103431          BCS     7$                ;ERROR IF NUMBER TOO BIG
5707 050376 006337 003404          ASL     P$NUM
5708 050402 103426          BCS     7$                ;ERROR IF NUMBER TOO BIG
5709 050404 123727 003406 000012 CMPB    P$RADX,#10.        ;SEE IF DECIMAL RADIX
5710 050412 001004          BNE     4$                ;BR IF NOT EQUAL

```

```

5711 050414 063737 003402 003404      ADD      P$CNT,P$NUM
5712 050422 103416                    BCS      7$      ;ERROR IF NUMBER TOO BIG
5713 050424 060237 003404 4$:      ADD      R2,P$NUM
5714 050430 103413                    BCS      7$      ;ERROR IF NUMBER TOO BIG
5715 050432 005305                    DEC      R5
5716 050434 001344                    BNE      3$
5717 050436 105737 003407      TSTB     P$RADX+1 ;SEE IF NUM WAS PRECEDED BY A - SIGN
5718 050442 001402                    BEQ      15$     ; BR IF NO
5719 050444 005437 003404      NEG      P$NUM   ; ELSE NEGATE THE NUMBER BEFORE LEAVING
5720 050450 004737 047764      15$:     JSR      PC,TRVACT ;SINCE NUMERIC FOUND, GO TAKE ACTION
5721 050454 000137 050014      JMP      TRVNOB  ;GO POINT R3 TO NEXT NODE
5722
5723 050460 012746 023433      7$:     PRINTF #CLINBG ;PRINT NUMBER TOO BIG ERR
5724 050460 012746 000001                    MOV      #CLINBG,-(SP)
5725 050464 012746 000001                    MOV      #1,-(SP)
5726 050470 010600                    MOV      SP,R0
5727 050472 104417                    TRAP     C$PNTF
5728 050474 062706 000004      ADD      #4,SP
5729 050500 112737 177777 003411 5$:     MOVVB   #-1,P$GDBD ;SET ERROR RETURN FLAG
5730 050506 000137 050004      JMP      TRVBRC ;TAKE 'MISS' BRANCH
5731
5732
5733 050512 005005      TRVALP: CLR      R5 ;CLEAR ALPHA FOUND FLAG
5734 050514 121427 000101 1$:     CMPB   (R4),#101 ;SEE IF CHAR. LESS THAN A 'A'
5735 050520 002406                    BLT     2$      ;BR IF YES (NOT ALPHA)
5736 050522 121427 000132      CMPB   (R4),#132 ;SEE IF CHAR. GREATER THAN A 'Z'
5737 050526 003003                    BGT     2$      ;BR IF YES (NOT ALPHA)
5738 050530 005204                    INC     R4      ;UPDATE CMD LINE PTR TO NEXT CHAR
5739 050532 005205                    INC     R5      ;INDICATE AN ALPHA WAS FOUND
5740 050534 000767                    BR      1$      ;GO LOOK AT NEXT CHAR.
5741 050536 005705      2$:     TST     R5 ;SEE IF ANY ALPHA'S WERE FOUND
5742 050540 001404                    BEQ     3$      ;BR IF NO
5743 050542 004737 047764      JSR     PC,TRVACT ;IF ANY FOUND TAKE ACTION
5744 050546 000137 050014      JMP     TRVNOB  ;THEN UPDATE R3 TO NEXT NODE -NO BRANCH
5745 050552 000137 050004      3$:     JMP     TRVBRC  ;NONE FOUND, TAKE MISS BRANCH
5746
5747 050556 005005      TRVALN: CLR      R5 ;CLEAR ALPHANUM FOUND FLAG
5748 050560 121427 000060 10$:    CMPB   (R4),#60 ;SEE IF CHAR. LESS THAN A '0'
5749 050564 002417                    BLT     2$      ;BR IF YES (NOT NUMERIC OR ALPHA)
5750 050566 121427 000072      CMPB   (R4),#72 ;SEE IF CHAR. GREATER THAN A '9'
5751 050572 003003                    BGT     1$      ;BR IF YES (NOT NUMERIC)
5752 050574 005204                    INC     R4      ;UPDATE CMD LINE PTR TO NEXT CHAR.
5753 050576 005205                    INC     R5      ;INDICATE A NUMERIC FOUND
5754 050600 000767                    BR      10$     ;GO LOOK AT NEXT CHAR.
5755 050602 121427 000101 1$:     CMPB   (R4),#101 ;SEE IF CHAR. LESS THAN A 'A'
5756 050606 002406                    BLT     2$      ;BR IF YES (NOT ALPHA)
5757 050610 121427 000132      CMPB   (R4),#132 ;SEE IF CHAR. GREATER THAN A 'Z'
5758 050614 003003                    BGT     2$      ;BR IF YES (NOT ALPHA)
5759 050616 005204                    INC     R4      ;UPDATE CMD LINE PTR TO NEXT CHAR
5760 050620 005205                    INC     R5      ;INDICATE AN ALPHA FOUND
5761 050622 000756                    BR      10$     ;GO LOOK AT NEXT CHAR.
5762 050624 005705      2$:     TST     R5 ;SEE IF ANY ALPHANUM'S WERE FOUND
5763 050626 001404                    BEQ     3$      ;BR IF NO
5764 050630 004737 047764      JSR     PC,TRVACT ;IF ANY FOUND TAKE ACTION
5765 050634 000137 050014      JMP     TRVNOB  ;THEN UPDATE R3 TO NEXT NODE -NO BRANCH
5766 050640 000137 050004      3$:     JMP     TRVBRC  ;NONE FOUND, TAKE MISS BRANCH

```

5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796

050644 010401
050646 010305
050650 062705 000006
050654 005037 003402
050660 105715
050662 001411
050664 105711
050666 001407
050670 121115
050672 001005
050674 005237 003402
050700 005201
050702 005205
050704 000765
050706 005737 003402
050712 001406
050714 010104
050716 004737 047764
050722 066303 000004
050726 000207
050730 000137 050004

TRVSTR: MOV R4,R1
MOV R3,R5
ADD #6,R5
CLR P\$CNT
2\$: TSTB (R5)
BEQ 10\$
TSTB (R1)
BEQ 10\$
CMPB (R1),(R5)
BNE 10\$
INC P\$CNT
INC R1
INC R5
BR 2\$
10\$: TST P\$CNT
BEQ 15\$
MOV R1,R4
JSR PC,TRVACT
ADD 4(R3),R3
RTS PC
15\$: JMP TRVBRC

:POINT R1 TO CMD STRING
:POINT R5 TO MATCH STRING FROM CLI NODE
:CLEAR CHAR MATCH COUNT
:SEE IF END OF MATCH STRING YET
:BR IF YES
:SEE IF END OF CMD LINE YET
:BR IF YES
:SEE IF CHARACTERS MATCH
:BR IF NO
:MATCH -INCREMENT MATCH COUNT
:UPDATE STRING POINTERS
:BR TO CONTINUE CHECKING CHARS.
:WHEN DONE SEE IF ANY MATCHES FOUND
:BR IF NO, GO TAKE THE MISS BRANCH
:POINT CMD POINTER TO END OF STRING &
:IF A MATCH FOUND, GO DO MATCH ACTION
:UPDATE R3 TO NEXT NODE (NO BRANCH)
:(NO RETURN THRU TRVNOB SINCE DIFFERENT
: DISPLACEMENT DUE TO MATCH STRING)
: GO TAKE BRANCH
: (PARSED OK), -1 IF ILL CMD.....

5797
5798
5799
5800
5801
5802
5803
5804
5805
5806
5807
5808
5809
5810
5811
5812
5813
5814
5815
5816
5817

050734
050734

050734 004737 042552

050740
050740
050740 104425

.SBTTL REPORT CODING SECTION

:+
: THE REPORT CODING SECTION CONTAINS THE
: 'PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.
:--

BGNRPT

JSR PC,REPORT

ENDRPT

L\$RPT::

;CALL SUBROUTINE TO DUMP EVENT LOG
; AND BASE TABLE

L10011: TRAP C\$RPT


```
5818 .SBTTL PROTECTION TABLE
5819
5820 :++
5821 : THIS TABLE IS USED BY THE RUNTIME SERVICES
5822 : TO PROTECT THE LOAD MEDIA.
5823 :--
5824
5825 050742 BGNPROT
5826 050742 L$PROT::
5827
5828 050742 177777 -1 ;OFFSET INTO P-TABLE FOR CSR ADDRESS
5829 050744 177777 -1 ;OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
5830 050746 177777 -1 ;OFFSET INTO P-TABLE FOR DRIVE NUMBER
5831
5832 050750 ENDPROT
5833
```

5834
5835
5836
5837
5838
5839
5840
5841
5842
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861
5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889

.SBTTL INITIALIZE SECTION

;++
: THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
: AT THE BEGINNING OF EACH PASS.
:--

BGNINIT

LSINIT::

```

005737 017376      TST      DCLFLG      ;IS DOCLEAN SET
001403              BEQ      INIT1      ;BRANCH IF NOT
005037 017376      CLR      DCLFLG      ;IF SET CLEAR IT
104444              DOCLN
012737 177777 017400 INIT1: MOV      #-1,RESFLG    ;SET RESTART FLAG
                                REDEF     #EF.START    ;IF HERE CAUSE OF START,DO SOME INIT
                                TRAP      C$DCLN
012700 000040              MOV      #EF.START,RO
                                TRAP      C$REFG
103417              BCOMPLETE      START
                                REDEF     #EF.RESTART   ;IF HERE CAUSE OF RESTART, DO SOME INIT
                                BCS       START
                                MOV      #EF.RESTART,RO
                                TRAP      C$REFG
012700 000037              BCOMPLETE      RESTRT
                                REDEF     #EF.CONTINUE  ;SEE IF WE'RE HERE CAUSE OF A CONTINUE
                                BCS       RESTRT
                                MOV      #EF.CONTINUE,RO
                                TRAP      C$REFG
103514              BNCOMPLETE      S1
                                BCC       S1
                                JMP      ENDIT
000137 051644      S1:  REDEF     #EF.NEW
                                MOV      #EF.NEW,RO
                                TRAP      C$REFG
103002              BCOMPLETE      NEW
                                ;IF YES, BR AROUND LOGUNIT # SETUP
051022 000137 051644      JMP      ENDIT
012700 000035      S1:  REDEF     #EF.NEW
                                MOV      #EF.NEW,RO
                                TRAP      C$REFG
103522              BCOMPLETE      NEW
                                ;IF YES, BR AROUND LOGUNIT # SETUP
051036 000524      BR      GETPRM
                                BCS       NEW
005037 017400      START: CLR      RESFLG
                                ;CLEAR RESTART FLAG SINCE HERE ON START
                                BRESET
                                ;INIT ENTIRE BUS
                                TRAP      C$RESET
051046 005037 017440      CLR      CLKVEC
                                ;CLEAR CLK VECTOR PTR. AS A FLAG IN
                                ; NO CLOCK IS FOUND.
                                ;SETUP R2 AS A PTR. TO CLOCK INFO BLOCK
                                ;LOOK FOR A LINE CLOCK
                                MOV      #L,RO
                                TRAP      C$CLK
012702 017434      MOV      #CLKCSR,R2
                                ;SETUP R2 AS A PTR. TO CLOCK INFO BLOCK
                                ;LOOK FOR A LINE CLOCK
                                MOV      RO,R1
                                TRAP      C$CLK
051056 000114      BNCOMPLETE      S2
                                ; IF NONE THERE GO LOOK FOR A P-CLOCK
                                BCC       S2
004737 041652      JSR      PC,CLKSET
                                ; GO SET UP CLOCK INFO TABLE & CLK VEC.
012737 000100 017444      MOV      #LCLKEN,CLKEN
                                ;SETUP THE ENABLE LINE CLOCK DATA
000457              BR      RESTRT

```

```

5890 051104          S2:   CLOCK   P,R1          ;LOOK FOR A P-CLOCK SINCE NO LINE CLOCK
5891 051104 012700 000120          ;MOV #P,R0
5892 051110 104462          ;TRAP C$CLCK
5893 051112 010001          ;MOV RO,R1
5894 051114          BNCOMplete   S3          ; IF NONE THERE GO SEE IF THIS IS LSI
5895 051114 103017          ;BCC S3
5896 051116 004737 041652          JSR   PC,CLKSET          ; ELSE GO SET UP CLOCK INFO & VECTOR
5897 051122 062737 000002 017434  ADD   #2,CLKCSR          ;POINT CLKCSR TO P-CLK COUNT SET REG.
5898 051130 012777 001600 146276  MOV   #PCLKCT,@CLKCSR   ;LOAD CLK SET REG. WITH COUNT VALUE
5899 051136 162737 000002 017434  SUB   #2,CLKCSR          ;POINT CLKCSR BAC TO P-CLK CSR
5900 051144 012737 000111 017444  MOV   #PCLKEN,CLKEN     ;SETUP THE ENABLE THE P-CLK DATA
5901 051152 000433          BR    RESTRT
5902
5903 051154          S3:   READBUS          ;READ BUS TYPE TO SEE IF ON AN LSI
5904 051154 104407          ;TRAP C$RDBU
5905 051156          BNCOMplete   S4          ;BR IF NOT, NO CHANCE OF A CLOCK
5906 051156 103021          ;BCC S4
5907 051160 012737 000100 017440  MOV   #100,CLKVEC       ;LOAD 100 AS CLK VECTOR
5908 051166 005037 017436          CLR   CLKBR             ;LOAD 0 AS CLK INT. LEVEL
5909 051172 012737 017444 017434  MOV   #CLKEN,CLKCSR     ;KLUDGE UP THE CSR & ENABLE DATA LOCS
5910 051200          GMANID  L5060,CLKHZ,D,377,50.,60.,YES
5911 051200 104443          ;TRAP C$GMAN
5912 051202 000406          BR    10000$
5913 051204 017442          .WORD CLKHZ
5914 051206 000052          .WORD T$CODE
5915 051210 027117          .WORD L5060
5916 051212 000377          .WORD 377
5917 051214 000062          .WORD T$LOLIM
5918 051216 000074          .WORD T$HILIM
5919 051220          ;10000$:
5920 051220 000410          BR    RESTRT
5921
5922
5923 051222          S4:   PRINTF  #BDCLK
5924 051222 012746 027230          ;MOV #BDCLK,-(SP)
5925 051226 012746 000001          ;MOV #1,-(SP)
5926 051232 010600          ;MOV SP,RO
5927 051234 104417          ;TRAP C$PNTF
5928 051236 062706 000004          ;ADD #4,SP
5929 051242 005037 017446          RESTRT: CLR  TIMMIN      ;CLEAR TIME SINCE START LOCATIONS
5930 051246 005037 017450          CLR  TIMSEC
5931 051252 013737 017442 017452  MOV   CLKHZ,TIMTCK     ;LOAD TICKS/SEC
5932 051260 012702 017464          MOV   #EVTLOG,R2      ;INIT EVENT TABLE TO ALL 1'S AFTER EACH
5933 051264 010237 017462          MOV   R2,EVTPTR      ; START OR RES AND INIT TABLE POINTER
5934 051270 012722 177777          1$:  MOV   #-1,(R2)+
5935 051274 020227 020520          CMP   R2,#EVTEND     ;SEE IF REACHED END OF TABLE
5936 051300 001373          BNE   1$             ;LOOP UNTIL DONE
5937
5938 051302 012737 177777 017372  NEW:  MOV   #-1,LOGUNT  ;INITIALIZE LOGICAL UNIT #
5939
5940 051310 005237 017372          GETPRM: INC  LOGUNT     ;POINT TO NEXT LOGICAL UNIT
5941 051314 023737 017372 002012  CMP   LOGUNT,L$UNIT   ;SEE IF PAST MAX. LOG. UNIT #
5942 051322 002367          BGE   NEW            ;BR IF YES, AND START OVER
5943
5944 051324          GPHARD LOGUNT,R1    ;GET THE P-TABLE FOR THIS LOG. UNIT
5945 051324 013700 017372          ;MOV LOGUNT,RO

```

```

5946 051330 104442
5947 051332 00001
5948 051334
5949 051334 103365
5950
5951 051336 011137 017406
5952
5953
5954
5955
5956 051342 016137 000002 023050
5957 051350 016137 000002 023052
5958 051356 005237 023052
5959 051362 016137 000002 023054
5960 051370 062737 000002 023054
5961 051376 016137 000002 023056
5962 051404 062737 000003 023056
5963 051412 016137 000002 023060
5964 051420 062737 000004 023060
5965 051426 016137 000002 023062
5966 051434 062737 000005 023062
5967 051442 016137 000002 023064
5968 051450 062737 000006 023064
5969 051456 016137 000002 023066
5970 051464 062737 000007 023066
5971
5972 051472 016137 000004 023070
5973 051500 016137 000004 023072
5974 051506 062737 000004 023072
5975 051514 016137 000006 023074
5976 051522 016137 000010 023100
5977 051530 016137 000012 023076
5978 051536 032737 000003 023076
5979 051544 001417
5980 051546 012737 000454 016202
5981 051554 012737 001130 016204
5982 051562 012737 001130 016210
5983 051570 012737 000024 016212
5984 051576 012737 001750 016214
5985 051604 005037 023102 11$:
5986 051610 032737 000001 023100
5987 051616 001407
5988 051620 052737 000004 023102
5989 051626 032737 000002 023100
5990 051634 001003
5991 051636 052737 000002 023102 1$:
5992 051644
5993 051644
5994 051644 012746 000340
5995 051650 012746 041676
5996 051654 013746 017440
5997 051660 012746 000003
5998 051664 104437
5999 051666 062706 000010
6000
6001
  
```

```

TRAP C$GPHRD
MOV RO,R1
;IF NO P-TABLE AVAIL., GO GET NEXT ONE
BCC GETPRM
BNCOMPLETE GETPRM
MOV (R1),FHDPLX ;PUT FULL OR HALF DUPLEX ANSWER IN LOC.
;DEVICE DEPENDENT PART OF GETTING INFO FROM P-TABLE
MOV 2(R1),SELO ;STORE AWAY CSR ADDRESSES
MOV 2(R1),BSEL1
INC BSEL1
MOV 2(R1),SEL2
ADD #2,SEL2
MOV 2(R1),BSEL3
ADD #3,BSEL3
MOV 2(R1),SEL4
ADD #4,SEL4
MOV 2(R1),BSEL5
ADD #5,BSEL5
MOV 2(R1),SEL6
ADD #6,SEL6
MOV 2(R1),BSEL7
ADD #7,BSEL7
MOV 4(R1),INVEC ;STORE AWAY INPUT INTERRUPT VECTOR
MOV 4(R1),OUTVEC
ADD #4,OUTVEC ;BUILD OUTPUT INTERRUPT VECTOR
MOV 6(R1),INTPRI ;STORE AWAY INTERRUPT PRIORITY
MOV 10(R1),DEVPAR ;SORE AWAY PARAMS
MOV 12(R1),OPTYP ;STORE AWAY DEVICE OPTION TYPE
BIT #3,OPTYP ;IS THIS A DMV
BEQ 11$
MOV #300.,DMVDF1
MOV #600.,DMVDF2
MOV #600.,DMVDF3
MOV #24,DMVDF4
MOV #1000.,DMVDF5 ;SET UP DMV DEFAULTS
11$: CLR STATYP ;CLEAR STATION TYPE
BIT #MTP,DEVPAR ;IS THIS MULTIPOINT
BEQ 1$ ;BRANCH IF PT TO PT
BIS #BIT2,STATYP ;IF MULTIPOINT SET BIT
BIT #TRBB,DEVPAR ;IS THIS A TRIB
BNE ENDIT ;BRANCH IF CONTROL
1$: BIS #BIT1,STATYP ;SET STATION TYPE
ENDIT:
SETVEC CLKVEC,#CLKINT,#340 ;SETUP CLOCK VECTOR
MOV #340,-(SP)
MOV #CLKINT,-(SP)
MOV CLKVEC,-(SP)
MOV #3,-(SP)
TRAP C$SVEC
ADD #10,SP
;DEVICE DEPENDENT VECTOR SETUP
  
```

```

6002
6003 051672          SETVEC  INVEC,#DVINS,INTPRI      ;SETUP INPUT INTERRUPT VECTOR
6004 051672 013746 023074          MOV      INTPRI,-(SP)
6005 051676 012746 066530          MOV      #DVINS,-(SP)
6006 051702 013746 023070          MOV      INVEC,-(SP)
6007 051706 012746 000003          MOV      #3,-(SP)
6008 051712 104437          TRAP    C$$SVEC
6009 051714 062706 000010          ADD     #10,SP
6010 051720          SETVEC  OUTVEC,#DVOUTS,INTPRI      ;SETUP OUTPUT INTERRUPT VECTOR
6011 051720 013746 023074          MOV      INTPRI,-(SP)
6012 051724 012746 066546          MOV      #DVOUTS,-(SP)
6013 051730 013746 023072          MOV      OUTVEC,-(SP)
6014 051734 012746 000003          MOV      #3,-(SP)
6015 051740 104437          TRAP    C$$SVEC
6016 051742 062706 000010          ADD     #10,SP
6017
6018 051746          SETPRI  #PRI00          ;SET THE 'RUN' PRIORITY TO 0
6019 051746 012700 000000          MOV     #PRI00,R0
6020 051752 104441          TRAP   C$$SPRI
6021 051754          EXIT   INIT
6022 051754 104432          TRAP   C$EXIT
6023 051756 000002          .WORD  L10013-
6024
6025
6026          .EVEN
6027
6028 051760          ENDINIT
6029 051760
6030 051760 104411          L10013: TRAP   C$INIT
  
```

6031
6032
6033
6034
6035
6036
6037
6038
6039
6040 051762
6041 051762
6042
6043
6044 051762
6045 051762
6046 051762 104461

.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

L\$AUTO::

ENDAUTO

L10014: TRAP C\$AUTO

6047
6048
6049
6050
6051
6052
6053
6054
6055
6056
6057
6058
6059
6060
6061
6062
6063
6064
6065
6066
6067
6068
6069
6070
6071
6072
6073

051764
051764

051764 012737 177777 017320
051772 004737 064726
051776 005037 017320
052002 005077 145426
052006
052006 012700 000340
052012 104441
052014
052014 104432
052016 000002

052020
052020
052020 104412

.SBTTL CLEANUP CODING SECTION

+++
: THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
: AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.

BGNCLN

MOV #-1,CLNSET ;SET THE CLEANUP FLAG
JSR PC;HLTTRB ;HALT ALL TRIBS
CLR CLNSET
CLR @CLKCSR ;DISABLE CLOCK
SETPRI #PRI07 ;SET PROCESSOR PRIORITY BACK TO 7

L\$CLEAN::

MOV #PRI07,RO
TRAP C\$SPRI

TRAP C\$EXIT
.WORD L10015-

EXIT CLN

.EVEN

ENDCLN

L10015:
TRAP C\$CLEAN

6074
6075
6076
6077
6078
6079
6080
6081 052022
6082 052022
6083
6084
6085 052022
6086 052022 000167
6087 052024 000000
6088
6089
6090
6091
6092 052026
6093 052026
6094 052026 104453

.SBTTL DROP UNIT SECTION

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

BGNDU

L\$DU::

EXIT DU

.WORD JSJMP
.WORD L10016-2-

.EVEN

ENDDU

L10016: TRAP C\$DU

6095
6096
6097
6098
6099
6100
6101
6102
6103
6104
6105
6106
6107
6108
6109
6110
6111
6112
6113
6114
6115
6116
6117
6118

.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.
:--

BGNAU

L\$AU::

EXIT AU

.WORD JSJMP
.WORD L10017-2-

.EVEN

ENDAU

L10017: TRAP C\$AU

052030
052030

052030
052030 000167
052032 000000

052034
052034
052034 104452

TEST 1: SETUP AND MODES OF OPERATION

.SBTTL TEST 1: SETUP AND MODES OF OPERATION

;++
: TEST TO DETECT FAULTS IN THE DATA COMMUNICATION LINK. THIS TEST WILL
: THE PROVIDE COVERAGE NECESSARY TO ISOLATE FAILURES TO THE COMPUTER
: EQUIPMENT, THE COMMUNICATION LINK, OR THE MODEM.
:--

6119
6120
6121
6122
6123
6124
6125
6126
6127
6128
6129
6130 052036
6131 052036
6132
6133
6134
6135
6136 05207 013777 017444 145370
6137
6138 052044
6139 052044 005001
6140 052046 012737 000001 017454
6141 052054 005737 017454
6142 0060 001412
6143 062 005301
6144 052064 001373
6145 052066
6146 052066 012746 027254
6147 052072 012746 000001
6148 052076 010600
6149 052100 104417
6150 052102 062706 000004
6151
6152 052106 005737 017400
6153 052112 001120
6154
6155
6156
6157 052114 005037 017344
6158 052120 005037 017262
6159 052124 005037 017244
6160 052130 012737 011416 017236
6161
6162 052136 005037 017234
6163 052142 012737 011512 017240
6164
6165 052150 012737 000005 017332
6166 052156 013737 002162 017334
6167 052164 012737 003416 017264
6168 052172 012737 004416 017246
6169
6170 052200 013737 017264 017342
6171 052206 013737 017236 017340
6172 052214 004737 045444
6173 052220 012737 000001 017260
6174

BGNTST

T1::

.SBTTL PROGRAM SETUP SECTION

```
MOV CLKEN,@CLKCSR ;ENABLE THE CLOCK
GTXRXB:
GTRA2: CLR R1
MOV #1,TIMER1 ;SET TIMER TO COUNT 1 TICK
1$: TST TIMER1 ;CHECK FOR IT TO BE COUNTED OFF
BEQ GTRA3 ;BRANCH IF CLOCK EXISTS (COUNTED A TICK)
DEC R1
BNE 1$ ;KEEP CHECKING UNTIL R1 DOES FULL COUNTDOWN
PRINTF #NOCLK ;PRINT BAD CLK MSG AND WARN OF HANG IF TIMEOUT
MOV #NOCLK,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP ($PNTF
ADD #4,SP
GTRA3: TST RESFLG ;SEE IF HERE AFTER A RESTART.
BNE GTRA5 ;BR IF HERE CAUSE OF A RESTART
; CLEAR COUNTS AND SET UP DEFAULTS
GTRA4: CLR TOTCC ;CLEAR TOTAL CHAR. COUNT TEMP. LOC.
CLR TTOTCC ; CLEAR TOTAL CHAR. COUNT FOR TX BUFF
CLR CTOTCC ; CLEAR TOTAL CHAR. COUNT FOR CMP BUFF
MOV #PTRTAB, TXPTR ;INIT TRANSMIT MESSAGE POINTER
CLR RXPTR ; ZERO RX POINTER
MOV #PTR13, CMPPTR ;INIT COMP POINTER
MOV #5,MSGTYP ;SET UP DEFAULT MSG TYPE (QUICK FOX - ITEP MSG)
MOV MSG5C,CURCC ;SET UP DEFAULT CHAR COUNT
MOV #TXBUF,TCURAD ;SET UP CURRENT ADD TO START OF TX BUFFER
MOV #CMPBUF,CCURAD ;SET UP CURRENT ADD TO START OF CMP BUFFER
MOV TCURAD,CURADD ;SETUP CURRENT ADDR TO START OF TXBUF
MOV TXPTR,CPTR ;SETUP CURRENT POINTER TABLE POINTER FOR TXBUF
JSR PC,BLDBUF ; GO BUILD POINTER TABLE AND BUFFER
MOV #1,TXMTOT ;BUMP TOTAL MESSAGE COUNT
```

```

6175 052226 013737 017240 017340      MOV      CMPPTR,CPTR      ;SET UP START OF COMPARE POINTER TABLE
6176 052234 013737 017246 017342      MOV      CCURAD,CURADD   ;SET UP CURRENT ADDR. TO START OF CMPBUF
6177 052242 012737 000005 017332      MOV      #5,MSGTYP
6178 052250 013737 002162 017334      MOV      MSG5C,CURCC
6179 052256 004737 045444                JSR      PC,BLDBUF      ;PUT DEFAULT MESSAGE INTO CMPBUF
6180 052262 012737 000001 017242      MOV      #1,CMPTOT      ;BUMP THE COMP MMSG COUNT
6181 052270 012737 000003 017402      MOV      #ACT,MODTYP    ;SET DEFAULT MODE= ACTIVE
6182 052276 005037 017404                CLR      MLTYP          ;SET DEFAULT MAINTENANCE LOOP MODE =NONE
6183 052302 012737 000001 017412      MOV      #1,RPASS      ;SET UP DEFAULT 'RUN PASS' COUNT TO 1
6184 052310 012737 000002 017410      MOV      #2,PARAM      ;SET UP PROG. PARAMETERS - DATA CHECKING ENABL
6185                                     ;OPERATOR STATUS MSGS. PRINT OFF
6186 052316 012737 000061 003252      MOV      #KTRB,KEYWD1  ;SET UP KEYWRD.
6187 052324 004737 055444                JSR      PC,ACTKAL      ;ZERO TRIB LIST
6188
6189 052330 004737 047212                JSR      PC,WRDEFP      ;GO WRITE DEFAULTS TO TRIBS
6190
6191 052334                                     PRINTF   #HLPO
6192 052334 012746 024036                MOV      #HLPO,-(SP)
6193 052340 012746 000001                MOV      #1,-(SP)
6194 052344 010600                MOV      SP,RO
6195 052346 104417                TRAP    C$PNTF
6196 052350 062706 000004                ADD     #4,SP
6197 052354 010637 017364      GTRAS:  MOV      SP,SAVSP      ;SAVE OFF STACK
6198 052360 013737 017402 020650      MOV      MODTYP,DEV1
6199 052366 013737 017404 020652      MOV      MLTYP,DEV2
6200 052374 013737 017412 020654      MOV      RPASS,DEV3
6201 052402 013737 017410 020656      MOV      PARAM,DEV4
6202 052410 004737 047342      JSR      PC,SHWOP      ;PRINT TO OPERATOR THE CURRENT MODE.....
6203
6204 052414      MANUAL      ;SEE IF MANUAL INTERVENTION ALLOWED
6205 052414 104450      TRAP    C$MANI
6206 052416      BCOMPLETE   GETCL ; BR IF YES (UAM=0 AND NOT CHAINED)
6207 052416 103412      BCS     GETCL
6208 052420 005737 017412      TST     RPASS      ;SEE IF THIS IS FIRST 'DCLT PASS'
6209 052424 001002      BNE     1$        ; BR IF NOT COMPLETED 1 PASS
6210 052426      EXIT     TST      ; IF DONE 1 PASS IN UNATTENDED MODE - EXIT
6211 052426 104432      TRAP    C$EXIT
6212 052430 014126      .WORD   L10020-.
6213 052432 012737 000001 017404 1$:  MOV      #TTL,MLTYP    ;SET UP DEFAULT FOR UNATTENDED MODE
6214 052440 000137 057030      JMP     GTR9        ; 'R M=ACT/LO=I/PAS=1/NOST/CH' AND RUN
6215
6216      .SBTTL      COMMAND LINE FETCH & INTERPRETATION SECTION
6217
6218 052444 105037 003411      GETCL:  CLRB    P$GDBD      ;CLEAR CMD LINE PARSING ERROR FLAGS
6219 052450 105037 003410      CLRB    P$NNUF
6220 052454      GMANID   CLISP,CMDBUF,A,0,1,72.,NO ;GET A COMMAND LINE FROM OPR.
6221 052454 104443      TRAP    C$GMAN
6222 052456 000406      BR      10000$
6223 052460 003130      .WORD   CMDBUF
6224 052462 000142      .WORD   T$CODE
6225 052464 023352      .WORD   CLISP
6226 052466 000000      .WORD   0
6227 052470 000001      .WORD   T$LOLIM
6228 052472 000110      .WORD   T$HILIM
6229 052474      10000$:
6230 052474 012737 003130 003374      MOV      #CMDBUF,P$BUFA

```

COMMAND LINE FETCH & INTERPRETATION SECTION

```

6231 052502 012737 021166 003376      MOV      #CLITRE,P$TREE
6232 052510 012737 053410 003400      MOV      #CLIACT,P$ACT
6233 052516 005037 003254      CLR      QUALFG          ;CLEAR QUALIFIER FLAG LOCATION
6234 052522 004737 047644      JSR      PC,P$TRV        ;GO PARSE COMMAND LINE
6235 052526 105737 003411      TSTB    P$GDBD          ;SEE IF PARSED OK OR AN ERROR
6236 052532 001412
6237 052534
6238 052534 012746 023360
6239 052540 012746 000001
6240 052544 010600
6241 052546 104417
6242 052550 062706 000004
6243 052554 000137 052444
6244 052560 105737 003410      1$:     JMP      GETCL
6245 052564 001412      TSTB    P$NNUF          ;SEE IF INCOMPLETE COMMAND TYPED
6246 052566
6247 052566 012746 023410      BEQ     10$
6248 052572 012746 000001      PRINTF  #CLINUF
6249 052576 010600
6250 052600 104417
6251 052602 062706 000004
6252 052606 000137 052444      JMP     GETCL
6253
6254 052612 023727 003252 000067 10$:    CMP     KEYWD1,#SETET    ;WAS 'SET E=T' ENTERED ?
6255 052620 001711      BEQ     GETCL          ;YES,BRANCH
6256 052622 023727 003252 000004      CMP     KEYWD1,#RUN    ;SEE IF RUN WAS TYPED
6257 052630 001002      BNE     11$           ; BR IF NO
6258 052632 000137 057030      JMP     GTR9          ; START EXEC. IF YES
6259 052636 023727 003252 000052 11$:    CMP     KEYWD1,#DMPS   ;IS IT DUMP
6260 052644 001004      BNE     14$
6261 052646 004737 045210      JSR     PC,DUMPSR     ;GO TO DUMPSR
6262 052652 000137 052444      JMP     GETCL         ;AND GO BACK
6263 052656 023727 003252 000066 14$:    CMP     KEYWD1,#EXIT   ;IS IT EXIT
6264 052664 001005      BNE     40$           ;BRANCH IF NOT
6265 052666 012737 177777 017376      MOV     #-1,DCLFLG    ;SET DO CLEAN FLAG
6266 052674
6267 052674 104432
6268 052676 013660
6269 052700 023727 003252 000010 40$:    CMP     KEYWD1,#SETEXP ;SEE IF SET EXPECTED
6270 052706 001001      BNE     4$            ; BR IF YES (A SETEXP WAS TYPED)
6271 052710 000525      BR      2$
6272 052712 023727 003252 000011 4$:     CMP     KEYWD1,#SETTRN ;SEE IF SET TX
6273 052720 001407      BEQ     5$            ; BR IF YES
6274 052722 105737 003412      TSTB    WRFLG
6275 052726 001402      BEQ     77$
6276 052730 004737 045774      JSR     PC,DOGLOB     ;DO GLOBAL
6277 052734 000137 052444      77$:    JMP     GETCL
6278
6279 052740 013737 017262 017344 5$:     MOV     TTOTCC,TOTCC
6280 052746 023727 017344 001000      CMP     TOTCC,#BUFLIM ;SEE IF BUFFER ALREADY FULL
6281 052754 002414      BLT     15$           ; BR IF NOT FULL (BUFLIM # OF CHARS.)
6282 052756
6283 052756 012746 027375      PRINTF  #MSGTRN,#BUFEX ; ELSE TELL OPR. AND DON'T BUILD MSG.
6284 052762 012746 027413
6285 052766 012746 000002
6286 052772 010600

```

CZCLMBO DMP/V-11 DCLT
CZCLMB.P11 28-JUL-81

MACY11 30A(1052) 28-JUL-81 13:35
13:34

COMMAND LINE FETCH & INTERPRETATION SECTION

SEQ 0155

6287	052774	104417									TRAP	C\$PNTF
6288	052776	062706	000006								ADD	#6,SP
6289	053002	000137	052444									
6290	053006	005737	017262	15\$:	JMP	GETCL						
6291	053012	001002			TST	TTOTCC						
6292	053014	005037	017260		BNE	6\$						
6293	053020	012737	011416	017236	6\$:	CLR	TXMTOT					
6294	053026	013701	017260		MOV	#PTRTAB, TXPTR						
6295	053032	020127	000017		MOV	TXMTOT, R1						
6296	053036	002414			CMP	R1, #MSGLIM						
6297	053040				BLT	17\$						
6298	053040	012746	027335		PRINTF	#MSGTRN, #TABEX						
6299	053044	012746	027413								MOV	#TABEX, -(SP)
6300	053050	012746	000002								MOV	#MSGTRN, -(SP)
6301	053054	010600									MOV	#2, -(SP)
6302	053056	104417									MOV	SP, R0
6303	053060	062706	000006								TRAP	C\$PNTF
6304	053064	000137	052444								ADD	#6,SP
6305	053070	006301		17\$:	JMP	GETCL						
6306	053072	006301			ASL	R1						
6307	053074	060137	017236		ASL	R1						
6308	053100	013737	017236	017340	ADD	R1, TXPTR						
6309	053106	013737	017264	017342	MOV	TXPTR, CPTR						
6310	053114	004737	045346		MOV	TCURAD, CURADD						
6311	053120	004737	045444		JSR	PC, ADDCC						
6312	053124	013737	017340	017236	JSR	PC, BLDBUF						
6313	053132	013737	017344	017262	MOV	CPTR, TXPTR						
6314	053140	013737	017342	017264	MOV	TOTCC, TTOTCC						
6315	053146	005237	017260		MOV	CURADD, TCURAD						
6316	053152	005337	003256		INC	TXMTOT						
6317	053156	001270			DEC	QUALVL						
6318	053160	000137	052444		BNE	5\$						
6319					JMP	GETCL						
6320	053164	013737	017244	017344	2\$:	MOV	CTOTCC, TOTCC					
6321	053172	023727	017344	001000	CMP	TOTCC, #BUFLIM						
6322	053200	002414			BLT	16\$						
6323	053202				PRINTF	#MSGTRN, #BUFEX						
6324	053202	012746	027375									
6325	053206	012746	027413								MOV	#BUFEX, -(SP)
6326	053212	012746	000002								MOV	#MSGTRN, -(SP)
6327	053216	010600									MOV	#2, -(SP)
6328	053220	104417									MOV	SP, R0
6329	053222	062706	000006								TRAP	C\$PNTF
6330	053226	000137	052444								ADD	#6,SP
6331	053232	005737	017244	16\$:	JMP	GETCL						
6332	053236	001002			TST	CTOTCC						
6333	053240	005037	017242		BNE	7\$						
6334	053244			7\$:	CLR	CMPTOT						
6335	053244	012737	011512	017240	MOV	#PTR13, CMPPTR						
6336	053252	013701	017242		MOV	CMPTOT, R1						
6337												
6338	053256	020127	000017		CMP	R1, #MSGLIM						
6339	053262	002414			BLT	18\$						
6340	053264				PRINTF	#MSGTRN, #TABEX						
6341	053264	012746	027335									
6342	053270	012746	027413								MOV	#TABEX, -(SP)
											MOV	#MSGTRN, -(SP)

; THEN GO GET A NEW COMMAND
; IF FIRST 'SET' THEN GET RID OF DEFAULT

; GET POSITION OF END OF TX LIST

; SEE IF MSG COUNT EXCEEDED.
; BR IF NO
; ELSE TELL OPR. AND DON'T BUILD MSG.

; THEN GO GET A NEW COMMAND.
; # OF MSGS *4 = NEXT FREE PTR BLOCK

; SETUP CHAR. COUNT, CURRENT ADDR, & PTR

; ADD IN CHAR. COUNT AND CHECK TOTAL
; GO BUILD MESSAGE IN BUFFER AND PTRS.

; UPDATE CHAR. COUNT, CURR ADDR, & PTR

; DEC THE COPY COUNT

; SETUP CHAR. COUNT, CURR. ADDR. & PTR
; SEE IF BUFFER ALREADY FULL
; BR IF NOT FULL (BUFLIM # OF CHARS.)
; ELSE TELL OPR. AND DON'T BUILD MSG.

; THEN GO GET A NEW COMMAND
; IF FIRST 'SET' THEN GET RID OF DEFAULT

; INIT COMPARE MESSAGE POINTER

; SEE IF MSG COUNT EXCEEDED.
; BR IF NO
; ELSE TELL OPR. AND DON'T BUILD MSG.

6343 053274 012746 000002
6344 053300 010600
6345 053302 104417
6346 053304 062706 000006
6347 053310 000137 052444
6348 053314 006301
6349 053316 006301
6350 053320 060137 017240
6351 053324 013737 017240 017340
6352 053332 013737 017246 017342
6353 053340 004737 045346
6354 053344 004737 045444
6355 053350 013737 017340 017240
6356 053356 005237 017242
6357 053362 013737 017342 017246
6358 053370 013737 017344 017244
6359 053376 005337 003256
6360 053402 001270
6361 053404 000137 052444
6362
6363
6364
6365
6366

18\$:

JMP GETCL
ASL R1
ASL R1
ADD R1,CMPPTR
MOV CMPPTR,CPTR
MOV CCURAD,CURADD
JSR PC,ADDCC
JSR PC,BLDBUF
MOV CPTR,CMPPTR
INC CMPTOT
MOV CURADD,CCURAD
MOV TOTCC,CTOTCC
DEC QUALVL
BNE 2\$
JMP GETCL

MOV #2,-(SP)
MOV SP,RO
TRAP C\$PNTF
ADD #6,SP

; THEN GO GET A NEW COMMAND.
;# OF MSGS *4 = NEXT FREE PTR BLOCK

;ADD IN XHAR. COUNT AND CHECK TOTAL

;UPDATE CHAR. COUNT, CURR ADDR. & PTR

;IF COPY WAS GIVEN, PUT MSG IN BUFF
; AGAIN
;GO BACK UNTIL GET A 'RUN'

```

6367
6368
6369
6370
6371 053410
6372 053410 006302
6373 053412 016202 053426
6374 053416 062702 053426
6375 053422 004712
6376 053424 000207
6377
6378
6379 053426 000166
6380 053430 000170
6381 053432 000200
6382 053434 001566
6383 053436 000300
6384 053440 000210
6385 053442 000324
6386 053444 000416
6387 053446 000740
6388 053450 000750
6389 053452 000766
6390 053454 000776
6391 053456 001006
6392 053460 001100
6393 053462 001574
6394 053464 001120
6395 053466 001200
6396 053470 001206
6397 053472 001216
6398 053474 001226
6399 053476 001236
6400 053500 001246
6401 053502 001264
6402 053504 001352
6403 053506 001362
6404 053510 001402
6405 053512 001410
6406 053514 001420
6407 053516 001430
6408 053520 001440
6409 053522 001466
6410 053524 001476
6411 053526 001602
6412 053530 001616
6413 053532 001650
6414 053534 001660
6415 053536 001670
6416 053540 001700
6417 053542 001710
6418 053544 001720
6419 053546 000160
6420 053550 001156
6421 053552 000674
6422 053554 000724

```

.SBTTL ACTION TABLE AND ROUTINES
: USER MUST CLEAR/SET P\$GDBD IF USE 'CLIBIF' IN CONNECTION WITH ACTION
: R2 WILL HOLD ACTION CODE FROM PARSING (CLI) NODE
: CLIACT:

ASL	R2	:MULTIPLY ACTION CODE BY 2
MOV	10\$(R2),R2	:OFFSET VALUE
ADD	#10\$,R2	:ADD BASE VALUE
JSR	PC,(R2)	:GO DO ACTION
RTS	PC	:RETURN TO TRVACT:

10\$: .WORD ACTNUL-10\$:BRIEF DESCRIPTION OF ACTIONS TAKEN
:NULL
:WORD ACTCLR-10\$:CLEAR
:WORD ACTSHO-10\$:SHOW
:WORD ACTCHK-10\$:CHECK
:WORD ACTRUN-10\$:RUN
:WORD ACTHLP-10\$:HELP
:WORD ACTCSE-10\$:CLEAR OR SHOW EXPECTED
:WORD ACTCST-10\$:CLEAR OR SHOW TRANSMIT
:WORD ACTSTE-10\$:SET EXPECTED
:WORD ACTSTT-10\$:SET TRANSMIT
:WORD ACTSZE-10\$:SIZE
:WORD ACTCOP-10\$:COPY
:WORD ACTNUM-10\$:NUMERIC VALUE FOR SIZE OR COPY
:WORD ACTOPM-10\$:QUOTED MESSAGE FROM USER
:WORD ACTSTS-10\$:STATUS
:WORD ACTEQO-10\$:END OF QUOTED MESSAGE FROM USER
:WORD ACTMSO-10\$:ONES DATA
:WORD ACTMS1-10\$:ZEROS DATA
:WORD ACTMS2-10\$:1ALT
:WORD ACTMS3-10\$:OACT
:WORD ACTMS4-10\$:ITEP
:WORD ACTMS5-10\$:CCITT
:WORD ACTMS6-10\$:ALPHA
:WORD ACTATV-10\$:ACTIVE MODE
:WORD ACTPAS-10\$:PASSIVE MODE
:WORD ACTREC-10\$:RECEIVE MODE
:WORD ACTLIS-10\$:LISTEN MODE
:WORD ACTDLL-10\$:DOWNLINE LOAD
:WORD ACTTRA-10\$:TRANSMIT MODE
:WORD ACTTAL-10\$:TALK MODE
:WORD ACTNO-10\$:/NO
:WORD ACTECH-10\$:ECHO
:WORD ACTCRC-10\$:SET CRC BIT
:WORD ACTPRO-10\$:SET PROTOCOL BIT
:WORD ACTRPS-10\$:STATUS
:WORD ACTMOP-10\$:REMOTE STATION IN MAINTENACE LOOP MODE
:WORD ACTTLP-10\$:INTERNAL TTL
:WORD ACTCLP-10\$:CABLE LOOP
:WORD ACTLLP-10\$:LOCAL MODEM LOOP
:WORD ACTRLP-10\$:REMOTE MODEM LOOP
:WORD ACTNUF-10\$:MORE COMMAND LINE NEEDED
:WORD ACTBCR-10\$:BAD CHARACTER IN OPERATOR MESSAGE
:WORD ACTDMS-10\$:DUMP MEMORY START ADDRESS
:WORD ACTDME-10\$:DUMP MEMORY END ADDRESS

6423	053556	000716	.WORD	ACTDMQ-10\$;DUMP WORD
6424	053560	000264	.WORD	ACTPRT-10\$;PRINT
6425	053562	001610	.WORD	ACTMOS-10\$;MODEM STATUS CHANGE
6426	053564	002474	.WORD	ACTSLS-10\$;SHOW TRIB LIST
6427	053566	001776	.WORD	ACTETB-10\$;ESTABLISH TRIB
6428	053570	002006	.WORD	ACTKTB-10\$;KILL TRIB
6429	053572	002016	.WORD	ACTKAL-10\$;KILL ALL
6430	053574	002730	.WORD	ACTEKT-10\$;FLAG TRIB KILLED
6431	053576	003334	.WORD	ACTCKT-10\$;CHECK VALID TRIB
6432	053600	002100	.WORD	ACTEWS-10\$;POLL PARAMETERS
6433	053602	000254	.WORD	ACTEXT-10\$;EXIT
6434	053604	001310	.WORD	ACTSEX-10\$;SET E=T COMMAND REV B EC
6435					


```

6436
6437 053606 112737 177777 003410 ACTNUF: MOV# -1,P$NNUF ;SET FLAG TO SAY NEED MORE OF COMMAND
6438 053614 000207 ACTNUL: RTS PC ;RETURN TO PARSER
6439
6440 053616 012737 000001 003252 ACTCLR: MOV #CLEAR,KEYWD1 ;SET LOC TO SAY A CLEAR WAS TYPED
6441 053624 000207 RTS PC
6442
6443 053626 012737 000002 003252 ACTSHO: MOV #SHOW,KEYWD1 ;SET LOC. TO SAY A SHOW WAS TYPED
6444 053634 000207 RTS PC
6445
6446 053636 012702 003260 ACTHLP: MOV #HLPTAB,R2 ;SETUP R2 AS A POINTER TO HELP MSG TABLE
6447 053642 1$ : PRINTF #HLPF,(R2)+ ;PRINT HELP INFORMATION MESSAGES
6448 053642 012246
6449 053644 012746 024114 MOV (R2)+,-(SP)
6450 053650 012746 000002 MOV #HLPF,-(SP)
6451 053654 010600 MOV #2,-(SP)
6452 053656 104417 MOV SP,R0
6453 053660 062706 000006 TRAP C$PNTF
6454 053664 020227 003304 ADD #6,SP
6455 053670 001364
6456 053672 012737 000005 003252 CMP R2,#HLPEND ;SEE IF ALL INFO PRINTED YET
6457 053700 000207 BNE 1$ ;IF NO KEEP PRINTING
6458 053702 012737 000066 003252 ACTEXT: MOV #EXIT,KEYWD1 ;SET UP KEYWORD AND SCOOT OUT OF HERE
6459 053710 000207 RTS PC ;SET LOC. TO SAY A HELP WAS TYPED
6460 053712 012737 000055 003252 ACTPRT: MOV #PRNT,KEYWD1 ;CALL ROUTINE TO PRINT EVENT LOG AND BASE TABLE
6461 053720 004737 042552 JSR PC,REPORT
6462 053724 000207 RTS PC
6463
6464 053726 012737 000004 003252 ACTRUN: MOV #RUN,KEYWD1 ;SET RUN FLAG
6465 053734 112737 177777 003410 MOV# -1,P$NNUF ;SET FLAG TO SAY NEED MORE OF COMMAND
6466 053742 012737 000001 017412 MOV #1,RPASS ;SET DEFAULT RUN 'PASS' TO 1
6467 053750 000207 RTS PC
6468
6469 053752 012737 011512 017240 ACTCSE: MOV #PTR13,CMPPTR ;INIT COMPARE MESSAGE POINTER
6470 053760 013701 017240 MOV CMPPTR,R1
6471
6472 053764 013702 017242 MOV CMPTOT,R2
6473 053770 105037 003410 CLRB P$NNUF ;FLAG THAT HAVE VALID COMMAND AT THIS PT.
6474 053774 023727 003252 000002 CMP KEYWD1,#SHOW ;SEE IF A CLEAR OR SHOW WAS TYPED
6475 054002 001471 BEQ ACTSHW ;BR IF A SHOW WAS TYPED
6476 054004 012737 000001 017242 MOV #1,CMPTOT ;CLEAR COMPARE MESSAGE COUNT, CHAR. COUNT
6477 054012 005037 017244 CLR CTOTCC ; AND RESET POINTER
6478
6479 054016 012737 011512 017240 MOV #PTR13,CMPPTR ;INIT COMPARE MESSAGE POINTER
6480 054024 013737 017240 017340 MOV CMPPTR,CPTR ;SET UP TO FILL IN DEFAULT MESSAGE
6481 054032 012701 004416 MOV #CMPBUF,R1
6482 054036 010137 017246 MOV R1,CCURAD
6483 054042 000431 BR ACTCLB
6484
6485 054044 012701 011416 ACTCST: MOV #PTRTAB,R1
6486 054050 013702 017260 MOV TXMTOT,R2
6487 054054 105037 003410 CLRB P$NNUF ;FLAG THAT HAVE VALID COMMAND AT THIS PT.
6488 054060 023727 003252 000002 CMP KEYWD1,#SHOW ;SEE IF A CLEAR OR SHOW WAS TYPED
6489 054066 001437 BEQ ACTSHW ;BR IF A SHOW WAS TYPED
6490 054070 012737 000001 017260 MOV #1,TXMTOT ;CLEAR TRANSMIT MESSAGE COUNT, CHAR. COUNT
6491 054076 005037 017262 CLR TTOTCC ; AND RESET POINTER

```



```

6545
6546
6547 054366 012737 000010 003252 ACTSTE: MOV #SETEXP,KEYWD1
6548 054374 000403 BR ACTSTX
6549
6550 054376 012737 000011 003252 ACTSTT: MOV #SETTRN,KEYWD1
6551 054404 012737 000001 003256 ACTSTX: MOV #1,QUALVL ;SET UP DEFAULT COPY TO 1 (/COPY=0)
6552 054412 000207 RTS PC
6553
6554 054414 012737 000012 003254 ACTSIZE: MOV #SIZE,QUALFG
6555 054422 000207 RTS PC
6556
6557 054424 012737 000013 003254 ACTCOP: MOV #QCOPY,QUALFG
6558 054432 000207 RTS PC
6559
6560 054434 023727 003254 000012 ACTNUM: CMP QUALFG,#SIZE ;SEE IF A SIZE OR COPY TYPED
6561 054442 001023 BNE 1$ ;BR IF IT WAS A COPY
6562 054444 005737 003404 TST P$NUM ;CHECK TO BE SURE DIDN'T TRY SIZE=0
6563 054450 001014 BNE 3$ ; BR IF NO
6564 054452 PRINTF #CLISE0
6565 054452 012746 023647 MOV #CLISE0,-(SP)
6566 054456 012746 000001 MOV #1,-(SP)
6567 054462 010600 MOV SP,R0
6568 054464 104417 TRAP C$PNTF
6569 054466 062706 000004 ADD #4,SP
6570 054472 112737 177777 003411 MOVB #-1,P$GDBD ;SET ERROR-IN-CMD FLAG
6571 054500 000411 BR 2$
6572 054502 013737 003404 017334 3$: MOV P$NUM,CURCC ;IF A SIZE LOAD CURCC WITH BYTE COUNT
6573 054510 000405 BR 2$
6574 054512 013737 003404 003256 1$: MOV P$NUM,QUALVL ;IF A COPY, LOAD COPY COUNT
6575 054520 005237 003256 INC QUALVL ;INCREMENT SO FIRST DEC MAKES IT REAL #
6576 054524 000522 2$: BR ACTMEX
6577
6578 054526 012737 000007 017332 ACTOPM: MOV #7,MSGTYP
6579 054534 010437 017350 MOV R4,TEMP ;KEEP TRACK OF START OF QUOTED TEXT
6580 054540 005237 017350 INC TEMP ; SO CAN CALC OPCNT AT END OF QUOTES
6581 054544 000207 RTS PC
6582
6583 054546 010402 ACTEQO: MOV R4,R2
6584 054550 163702 017350 SUB TEMP,R2
6585 054554 010237 017334 MOV R2,CURCC ;CALC BYTE COUNT FOR QUOTED TEXT
6586 054560 010237 002166 MOV R2,OPCNT
6587 054564 013701 017350 MOV TEMP,R1
6588 054570 012705 002524 MOV #OPBUF,R5
6589 054574 112125 1$: MOVB (R1)+,(R5)+ ;COPY QUOTED TEXT TO OPBUF
6590 054576 005302 DEC R2
6591 054600 001375 BNE 1$
6592 054602 000473 BR ACTMEX
6593
6594 054604 ACTBCR: PRINTF #CLIBCR ;BAD CHAR. IN OPR. QUOTED STRING
6595 054604 012746 023602 MOV #CLIBCR,-(SP)
6596 054610 012746 000001 MOV #1,-(SP)
6597 054614 010600 MOV SP,R0
6598 054616 104417 TRAP C$PNTF
6599 054620 062706 000004 ADD #4,SP
6600 054624 000207 RTS PC

```

```

6601
6602 054626 005037 017332      ACTMS0: CLR      MSGTYP
6603 054632 000435                BR      ACTME1
6604 054634 012737 000001 017332 ACTMS1: MOV      #1,MSGTYP
6605 054642 000431                BR      ACTME1
6606 054644 012737 000002 017332 ACTMS2: MOV      #2,MSGTYP
6607 054652 000425                BR      ACTME1
6608 054654 012737 000003 017332 ACTMS3: MOV      #3,MSGTYP
6609 054662 000421                BR      ACTME1
6610 054664 012737 000004 017332 ACTMS4: MOV      #4,MSGTYP
6611 054672 000415                BR      ACTME1
6612 054674 012737 000005 017332 ACTMS5: MOV      #5,MSGTYP
6613 054702 013737 002162 017334      MOV      MSG5C,CURCC
6614 054710 000430                BR      ACTMEX
6615 054712 012737 000006 017332 ACTMS6: MOV      #6,MSGTYP
6616 054720 013737 002164 017334      MOV      MSG6C,CURCC
6617
6618 054726 012737 000100 017334 ACTME1: MOV      #64.,CURCC
6619 054734 000416                BR      ACTMEX
6620
6621
6622
6623 054736 022737 000010 003252 ACTSEX: ;REV B BY EC
6624 054744 001404                CMP      #SETEXP,KEYWD1
6625 054746 112737 177777 003411      BEQ      10$
6626 054754 000406                MOV      #-1,P$GDBD
6627 054756 004737 045570                BR      ACTMEX
6628 054762 012737 000067 003252 10$: JSR      PC,FACSIMILE
6629 054770 000400                MOV      #SETET,KEYWD1
6630
6631
6632
6633 054772 105037 003410      ACTMEX: CLRB     P$NNUF
6634 054776 000207                RTS      PC
6635

```

;SETUP DEFAULT SIZE FOR THIS TYPE

;SETUP DEFAULT SIZE FOR THIS TYPE

;SETUP DEFAULT SIZE FOR MSG0-4
;EXIT

;DID WE GET HERE FROM 'SET E =' COMMAND?
;YES,BRANCH
;SET ERROR FLAG
;GO TO EXIT
;GO COPY TRANMIT BUFFER TO EXPECT BUFFER
;SET FLAG TO BE USED IN T1::
;GO TO EXIT

;CLEAR NOT-ENOUGH FLAG

6636	055000	012737	000003	017402	ACTATV: MOV	#ACT,MODTYP	
6637	055006	000432			BR	ACTM2X	
6638							
6639	055010	012737	000002	017402	ACTPAS: MOV	#PAS,MODTYP	
6640	055016	105037	003410		CLRB	P\$NNUF	:CLEAR NOT-ENOUGH FLAG
6641	055022	005037	017404		CLR	MLTYP	:CLEAR MAINT LOOP TYPE
6642	055026	000207			RTS	PC	
6643							
6644	055030	005037	017402		ACTREC: CLR	MODTYP	
6645	055034	000417			BR	ACTM2X	
6646							
6647	055036	012737	000006	017402	ACTLIS: MOV	#LIS,MODTYP	
6648	055044	000413			BR	ACTM2X	
6649							
6650	055046	012737	000004	017402	ACTDLL: MOV	#DOW,MODTYP	
6651	055054	000407			BR	ACTM2X	
6652							
6653	055056	012737	000001	017402	ACTTRA: MOV	#TRA,MODTYP	
6654	055064	000403			BR	ACTM2X	
6655							
6656	055066	012737	000005	017402	ACTTAL: MOV	#TAL,MODTYP	
6657							
6658	055074	042737	000004	017410	ACTM2X: BIC	#ECHOB,PARAM	:DISABLE /ECHO (ALL BUT PASSIVE MODE)
6659	055102	105037	003410		CLRB	P\$NNUF	:CLEAR NOT-ENOUGH FLAG
6660	055106	005037	017404		CLR	MLTYP	:CLEAR MAINT LOOP TYPE
6661	055112	000207			RTS	PC	
6662							

6663	055114	012737	000036	003254	ACTNO:	MOV	#NO,QUALFG		
6664	055122	000207				RTS	PC		
6665									
6666	055124	022737	000036	003254	ACTECH:	CMP	#NO,QUALFG		
6667	055132	001422				BEQ	1\$		
6668	055134	052737	000004	017410		BIS	#ECHOB,PARAM		
6669	055142	022737	000002	017402		CMP	#PAS,MODTYP		:BE SURE IN PASSIVE MODE IF
6670	055150	001416				BEQ	2\$:IF TRYING TO SET /ECHO
6671	055152					PRINTF	#CLINPS		
6672	055152	012746	023537					MOV	#CLINPS,-(SP)
6673	055156	012746	000001					MOV	#1,-(SP)
6674	055162	010600						MOV	SP,RO
6675	055164	104417						TRAP	C\$PNTF
6676	055166	062706	000004					ADD	#4,SP
6677	055172	112737	177777	003411		MOVB	#-1,P\$GDBD		
6678	055200	042737	000004	017410	1\$:	BIC	#ECHOB,PARAM		
6679	055206	005037	003254		2\$:	CLR	QUALFG		:CLEAR 'NO' OUT OF QUALIFIER FLAG
6680	055212	000501				BR	ACTLXX		
6681									
6682	055214	012701	000002		ACTCHK:	MOV	#DATCKB,R1		:SET DATA CHECK BIT
6683	055220	000413				BR	ACTQFG		
6684									
6685	055222	012701	000001		ACTSTS:	MOV	#STATB,R1		:SET THE STATUS BIT
6686	055226	000410				BR	ACTQFG		
6687									
6688	055230	012701	000020		ACTCRC:	MOV	#CRCB,R1		:SET THE CRC BIT
6689	055234	000405				BR	ACTQFG		
6690									
6691	055236	012701	000010		ACTMOS:	MOV	#MOCHK,R1		:SET THE MODEM BIT
6692	055242	000402				BR	ACTQFG		
6693									
6694	055244	012701	000040		ACTPRO:	MOV	#PROTOB,R1		:SET THE PROTOCOL BIT
6695									
6696	055250	050137	017410		ACTQFG:	BIS	R1,PARAM		
6697	055254	022737	000036	003254		CMP	#NO,QUALFG		
6698	055262	001002				BNE	1\$		
6699	055264	040137	017410			BIC	R1,PARAM		
6700	055270	005037	003254		1\$:	CLR	QUALFG		:CLEAR 'NO' OUT OF QUALIFIER FLAG
6701	055274	000450				BR	ACTLXX		
6702									
6703	055276	013737	003404	017412	ACTRPS:	MOV	P\$NUM,RPASS		:GET NUMBER OF 'RUN PASSES'
6704	055304	000444				BR	ACTLXX		
6705									
6706	055306	012737	000005	017404	ACTMOP:	MOV	#5,MLTYP		
6707	055314	000417				BR	ACTLPX		
6708	055316	012737	000001	017404	ACTTLP:	MOV	#1,MLTYP		
6709	055324	000413				BR	ACTLPX		
6710	055326	012737	000002	017404	ACTCLP:	MOV	#2,MLTYP		
6711	055334	000407				BR	ACTLPX		
6712	055336	012737	000003	017404	ACTLLP:	MOV	#3,MLTYP		
6713	055344	000403				BR	ACTLPX		
6714	055346	012737	000004	017404	ACTRLP:	MOV	#4,MLTYP		
6715									
6716	055354	022737	000003	017402	ACTLPX:	CMP	#ACT,MODTYP		:BE SURE IN ACTIVE IF TRYING TO SET LOOP
6717	055362	001415				BEQ	ACTLXX		: BR IF INACTIVE
6718	055364	112737	177777	003411		MOVB	#-1,P\$GDBD		

6719	055372	005037	017404
6720	055376		
6721	055376	012746	023475
6722	055402	012746	000001
6723	055406	010600	
6724	055410	104417	
6725	055412	062706	000004
6726	055416	105037	003410
6727	055422	000207	
6728			

CLR MLTYP
PRINTF #CLIBDL

;CLEAR ANY LOOP TYPE THAT MAY HAVE GOT SET

MOV	#CLIBDL,-(SP)
MOV	#1,-(SP)
MOV	SP,R0
TRAP	C\$PNTF
ADD	#4,SP

ACTLXX: CLR B P\$NNUF
RTS PC

;CLEAR NOT-ENOUGH FLAG

```

6729 055424 012737 000060 003252 ACTETB: MOV #ETRB,KEYWD1 ; RECORD THAT ESTABLISH TYPED
6730 055432 000207 RTS PC ;RETURN TO CALL
6731
6732 055434 012737 000061 003252 ACTKTB: MOV #KTRB,KEYWD1 ; RECORD THAT KILLTRIB TYPED
6733 055442 000207 RTS PC ;RETURN TO CALL
6734
6735 055444 105037 003410 ACTKAL: CLRB P$NNUF ; CLEAR INCOMPLETE INFO FLAG
6736 055450 022737 000061 003252 CMP #KTRB,KEYWD1 ; BE SURE "ALL" IS AFTER A "KILL"
6737 055456 001403 BEQ 11$ ; BR IF YES
6738 055460 112737 177777 003411 MOVB #-1,P$GDBD ; ELSE ERROR IN CMD
6739 055466 105737 003411 11$: TSTB P$GDBD ; SEE IF WAS AN ERROR FROM ..KTB
6740 055472 001401 BEQ 10$ ; BR IF NO
6741 055474 000413 BR 2$ ; ELSE EXIT
6742 055476 005037 015754 10$: CLR TRBTOT ; ZERO TOTAL # OF TRIB ADDRESSES
6743 055502 012702 015712 MOV #TRIBLS,R2 ; PT R2 TO TRIB ADDRESS TABLE
6744 055506 012705 000020 MOV #16.,R5 ; SETUP R5 AS COUNTER
6745 055512 005022 1$: CLR (R2)+ ; CLEAR 32 BYTES OF TABLE
6746 055514 005305 DEC R5
6747 055516 001375 BNE 1$
6748 055520 004737 047212 JSR PC,WRDEFP ;WRITE DEFAULTS TO POLL PARMS
6749 055524 000207 2$: RTS PC ;RETURN TO CALL
6750
6751 055526 010246 ACTEWS: MOV R2,-(SP) ;SAVE R2,R3,R4 ON THE STACK
6752 055530 010346 MOV R3,-(SP)
6753 055532 010446 MOV R4,-(SP)
6754 055534 005737 003414 TST VALTRB ;VALID TRIB? REV B EC
6755 055540 001517 BEQ ACTW7B ;NO,BRANCH REV B EC
6756 055542 112737 177777 003412 ACTWS9: MOVB #-1,WRFLG ;SET WRITE GLOBAL FLAG
6757 055550 PRINTF #POLPM,INDW ;PRINT POLL PARAMS FOR TRIB #
6758 055550 013746 015760 MOV INDW,-(SP)
6759 055554 012746 025232 MOV #POLPM,-(SP)
6760 055560 012746 000002 MOV #2,-(SP)
6761 055564 010600 MOV SP,R0
6762 055566 104417 TRAP C$PNTF
6763 055570 062706 000006 ADD #6,SP
6764 055574 005037 017354 CLR TEMP2
6765 055600 012737 000020 017350 MOV #16.,TEMP ;USE 16 BYTES AS MULTIPLIER
6766 055606 013737 015762 017232 MOV INDEX,MPLY ;USE TRIB INDEX [BYTE]
6767 055614 004737 046434 JSR PC,MPLY ;ON RETURN TEMP2=START ADDR OF
6768 ;THIS TRIBS POLL PRAMS
6769 055620 012702 000027 MOV #27,R2 ;INIT INDEX OF POLL PARAMS
6770 055624 032737 000002 023100 BIT #TRBB,DEVPAR ;IS THIS TRIB
6771 055632 001002 BNE ACTWS5 ;BRANCH IF NOT A TRIB
6772 055634 012702 000034 MOV #34,R2 ;ONLY 35 IS GOOD FOR TRIBS
6773 055640 005202 ACTWS5: INC R2
6774 055642 116205 020760 MOVB TSSIND(R2),R5 ;R5 = 0 FOR WORD 2 FOR BYTE
6775 055646 010204 MOV R2,R4
6776 055650 006304 ASL R4 ;MAKE R4 WORD INDEX
6777 055652 010403 MOV R4,R3
6778 055654 042703 177760 BIC #^C<17>,R3 ;MAKE R3 POLPAM INDEX
6779 055660 063703 017354 ADD TEMP2,R3
6780 055664 016337 016220 017350 MOV POLLIS(R3),TEMP ;GETS DEFAULT
6781 055672 016437 020660 017366 MOV TSSLST(R4),CONOTM
6782 055700 000175 055704 JMP @ACTWS1(R5) ;GO TO CORRECT ACTION
6783 055704 055710 ACTWS1: .WORD ACTWS2
6784 055706 056010 .WORD ACTWS3

```


6785 055710
6786 055710 013746 017350
6787 055714 013746 017366
6788 055720 012746 000002
6789 055724 010600
6790 055726 104417
6791 055730 062706 000006
6792 055734
6793 055734 104443
6794 055736 000406
6795 055740 017350
6796 055742 000032
6797 055744 025053
6798 055746 177777
6799 055750 000000
6800 055752 177777
6801 055754
6802 055754 013763 017350 016220
6803 055762 032737 000002 023100
6804 055770 001403
6805
6806 055772 022702 000037
6807 055776 001320
6808 056000 012604
6809 056002 012603
6810 056004 012602
6811 056006 000207
6812
6813
6814
6815 056010
6816 056010 005046
6817 056012 153716 017351
6818 056016 005046
6819 056020 153716 017350
6820 056024 013746 017366
6821 056030 012746 000003
6822 056034 010600
6823 056036 104417
6824 056040 062706 000010
6825 056044
6826 056044 104443
6827 056046 000406
6828 056050 017350
6829 056052 000032
6830 056054 025107
6831 056056 000377
6832 056060 000000
6833 056062 000377
6834 056064
6835 056064 113737 017351 017356
6836 056072
6837 056072 104443
6838 056074 000406
6839 056076 017356
6840 056100 000032

ACTWS2: PRINTF CONOTM,TEMP

GMANID EQUQ,TEMP,0,-1,0,-1,YES ;GET INPUT

MOV TEMP,-(SP)
MOV CONOTM,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTF
ADD #6,SP
TRAP C\$GMAN
BR 10001\$
.WORD TEMP
.WORD T\$CODE
.WORD EQUQ
.WORD -1
.WORD T\$LOLIM
.WORD T\$HILIM

10001\$:

ACTWS7: MOV TEMP,POLLIS(R3) ;PUT ANSWER BACK
BIT #TRBB,DEVPAR ;IS THIS TRIB
BEQ ACTW7B ;BRANCH IF TRIB

ACTW7A: CMP #37,R2 ;ALL DONE
BNE ACTWS5
ACTW7B: MOV (SP)+,R4 ;RESTORE R4,R3,R2
MOV (SP)+,R3
MOV (SP)+,R2

RTS PC ;RETURN TO CALLING ROUTINE

;GET INPUT FOR LO AND HI BYTES

ACTWS3: PRINTF CONOTM,<B,TEMP><B,TEMP+1>

CLR -(SP)
BISB TEMP+1,(SP)
CLR -(SP)
BISB TEMP,(SP)
MOV CONOTM,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTF
ADD #10,SP

GMANID EQUQ1,TEMP,0,377,0,377,YES

TRAP C\$GMAN
BR 10002\$
.WORD TEMP
.WORD T\$CODE
.WORD EQUQ1
.WORD 377
.WORD T\$LOLIM
.WORD T\$HILIM

10002\$:

MOVB TEMP+1,TEMP3
GMANID EQUQ2,TEMP3,0,377,0,377,YES

TRAP C\$GMAN
BR 10003\$
.WORD TEMP3
.WORD T\$CODE

```

6841 056102 025147
6842 056104 000377
6843 056106 000000
6844 056110 000377
6845 056112
6846 056112 113737 017356 017351      MOVB  TEMP3,TEMP+1
6847 056120 000715                      BR    ACTWS7
6848
6849 056122 105037 003410      ACTSLS: CLRB  P$NNUF      ; CLEAR THE INCOMPLETE CMD FLAG
6850 056126 012737 000002 003252      MOV    #SHOW,KEYWD1    ; SET UP TO LOOK LIKE A SHOW CMD
6851 056134 105737 003411      TSTB  P$GDBD          ; SEE IF WAS AN ERROR FROM ..KTB
6852 056140 001401      BEQ    10$            ; BR IF NO
6853 056142 000504      BR    5$              ; ELSE EXIT
6854 056144 005037 017300      10$:  CLR    LNCNT      ; INIT ADDR/LINE COUNTER
6855 056150 005737 015754      TST   TRBTOT         ; SEE IF LIST EMPTY
6856 056154 001011      BNE   1$              ; BR IF NO
6857 056156      PRINTS #SHTRE        ; PRINT THE TRIB LIST IS EMPTY
6858 056156 012746 025725      MOV    #SHTRE,-(SP)
6859 056162 012746 000001      MOV    #1,-(SP)
6860 056166 010600      MOV    SP,R0
6861 056170 104416      TRAP  C$PNTS
6862 056172 062706 000004      ADD   #4,SP
6863 056176 000456
6864 056200 012702 015712      1$:  BR    4$
6865 056204 012705 000040      MOV    #TRIBLS,R2    ; POINT R2 TO THE TRIB ADDR LIST
6866 056210      MOV    #32.,R5       ; SETUP R5 AS A COUNTER
6867 056210 012746 025764      PRINTS #SHTRH        ; PRINT TRIB LIST HEADER
6868 056214 012746 000001      MOV    #SHTRH,-(SP)
6869 056220 010600      MOV    #1,-(SP)
6870 056222 104416      MOV    SP,R0
6871 056224 062706 000004      TRAP  C$PNTS
6872 056230 105712      ADD   #4,SP
6873 056232 001435      2$:  TSTB  (R2)          ; SEE IF A NULL ENTRY
6874 056234 111237 017350      BEQ    3$            ; BR IF YES
6875 056240      MOVB  (R2),TEMP
6876 056240 005046      PRINTS #SHTAP,<B,TEMP>
6877 056242 153716 017350      CLR    -(SP)
6878 056246 012746 026015      BISB  TEMP,(SP)
6879 056252 012746 000002      MOV    #SHTAP,-(SP)
6880 056256 010600      MOV    #2,-(SP)
6881 056260 104416      MOV    SP,R0
6882 056262 062706 000006      TRAP  C$PNTS
6883 056266 005237 017300      ADD   #6,SP
6884 056272 022737 000010 017300      INC    LNCNT          ; INCREMENT PRINT COUNTER
6885 056300 001012      CMP    #8.,LNCNT     ; SEE IF TIME FOR A CR YET
6886 056302      BNE   3$
6887 056302 012746 031564      PRINTS #CR
6888 056306 012746 000001      MOV    #CR,-(SP)
6889 056312 010600      MOV    #1,-(SP)
6890 056314 104416      MOV    SP,R0
6891 056316 062706 000004      TRAP  C$PNTS
6892 056322 005037 017300      ADD   #4,SP
6893 056326 005202      3$:  CLR    LNCNT
6894 056330 005305      INC    R2            ; INCREMENT TABLE ADDRESS
6895 056332 001336      DEC    R5            ; SEE IF CHECKED ALL OF LIST
6896 056334      BNE   2$            ; BR BACK IF NO
6896 056334      PRINTS #CR          ; ELSE PRINT A PARTING CR

```



```

6953 056574 012737 000040 017350 ACTEKE: MOV #32.,TEMP ; SETUP TO ENTER A TRIB ADDRESS
6954
6955 056602 032737 000003 023076 BIT #3,OPTYP
6956 056610 001403 BEQ 1$ ;BRANCH IF DMP
6957 056612 012737 000014 017350 MOV #12.,TEMP
6958 056620 023737 015754 017350 1$: CMP TRBTOT,TEMP ; SEE IF LIST ALREADY FULL
6959 056626 002412 BLT 2$ ; BR IF NOT FULL YET
6960 056630 PRINTF #SHTFL,R1 ;PRINT ERROR IS LIST FULL
6961 056630 010146 MOV R1,-(SP)
6962 056632 012746 026025 MOV #SHTFL,-(SP)
6963 056636 012746 000002 MOV #2,-(SP)
6964 056642 010600 MOV SP,R0
6965 056644 104417 TRAP C$PNTF
6966 056646 062706 000006 ADD #6,SP
6967 056652 000442
6968 056654 012702 015712 2$: BR ACTEXX
6969 056660 013705 017350 MOV #TRIBLS,R2 ; NOW CHECK TO SEE ADDR IS UNIQUE
6970 056664 122201 MOV TEMP,R5
6971 056666 001423 3$: CMPB (R2)+,R1 ; CHECK EACH ADDR AGAINST NEW ONE
6972 056670 005305 BEQ 5$ ; BR IF EQUAL
6973 056672 001374 DEC R5
6974 BNE 3$ ; LOOP TIL ENTIRE TABLE CHECKED
6975 056674 012702 015712 MOV #TRIBLS,R2 ; ONCE CHECKED LIST
6976 056700 105722 4$: TSTB (R2)+ ; LOOK FOR EMPTY SLOT TO LOAD
6977 056702 001376 BNE 4$
6978 056704 110142 MOVB R1,-(R2) ; LOAD TRIB ADDR IN EMPTY SLOT
6979 056706 005237 015754 INC TRBTOT ; INC TOTAL # OF TRIB ADDRESSES
6980 056712 162702 015712 SUB #TRIBLS,R2 ;SUBTRACT START OF LIST FROM POINT TO
6981 ;GET INDEX
6982 056716 012737 177777 003414 MOV #-1,VALTRB ;SET VALID TRIB FLAG REV B EC
6983 056724 010237 015762 MOV R2,INDEX ;MOVE R2 TO INDEX
6984 056730 010137 015760 MOV R1,INDW ;MOVE TRIB NUMBER TO INDW
6985 056734 000411 BR ACTEXX
6986 056736 5$: PRINTF #SHTUN,R1 ; PRINT ADDR NOT UNIQUE ERROR
6987 056736 010146 MOV R1,-(SP)
6988 056740 012746 026113 MOV #SHTUN,-(SP)
6989 056744 012746 000002 MOV #2,-(SP)
6990 056750 010600 MOV SP,R0
6991 056752 104417 TRAP C$PNTF
6992 056754 062706 000006 ADD #6,SP
6993
6994 056760 ACTEXX:
6995 056760 000207 RTS PC ;RETURN TO CALL
6996
6997 056762 ACTCKT:
6998 056762 112737 177777 003410 MOVB #-1,P$NNUF ; SET INCOMPLETE INFO FLAG
6999 056770 032737 000001 023100 BIT #MTP,DEVPAR ; SEE IF IN PT-PT OR MULTIPT MODE
7000 056776 001013 BNE 1$ ; BR IF IN MULTIPT MODE
7001 057000 PRINTF #CLIPPE ; TRIB CMDS INVALID IN PT-PT MODE
7002 057000 012746 023700 MOV #CLIPPE,-(SP)
7003 057004 012746 000001 MOV #1,-(SP)
7004 057010 010600 MOV SP,R0
7005 057012 104417 TRAP C$PNTF
7006 057014 062706 000004 ADD #4,SP
7007 057020 112737 177777 003411 1$: MOVB #-1,P$GDBD ; SET THE ERROR IN CMD FLAG
7008 057026 000207 RTS PC ;RETURN TO CALL

```

7009
7010
7011

```

7012
7013 ; RX ALLOCATE CODE
7014
7015 057030 032737 000002 017410 GTR9: BIT #DATCKB,PARAM ;IS THIS DATA CHECK
7016 057036 001421 BEQ 44$ ;BRANCH IF NO
7017 057040 005737 017404 TST MLTYP
7018 057044 001416 BEQ 44$ ;BRANCH IF NOT LOOP
7019 057046 023737 017242 017260 CMP CMPTCT, TXMTOT ;ARE TX AND EX EQUAL
7020 057054 001412 BEQ 44$ ;BRANCH IF YES
7021 057056 PRINTF #CLIPW
7022 057056 012746 023746 MOV #CLIPW, -(SP)
7023 057062 012746 000001 MOV #1, -(SP)
7024 057066 010600 MOV SP, R0
7025 057070 104417 TRAP C$PNTF
7026 057072 062706 000004 ADD #4, SP
7027 057076 000137 052444 JMP GETCL
7028 057102 032737 000001 023100 44$: BIT #MTP, DEVPAR ;IS THIS MULTIPOINT
7029 057110 001004 BNE 3$ ;BRANCH IF MULTIPOINT
7030 057112 112737 000001 015712 MOVB #1, TRIBLS ;MAKE TRIBLS =1
7031 057120 000570 BR 2$
7032 057122 005737 015754 3$: TST TRBTOT ;IS TRIB TOTAL
7033 057126 001013 BNE 4$ ;ZERO?..BR IF NOT
7034 057130 PRINTF #SHTLPA ;PRINT ERROR MUST ESTABLISH TRIB
7035 057130 012746 026314 MOV #SHTLPA, -(SP)
7036 057134 012746 000001 MOV #1, -(SP)
7037 057140 010600 MOV SP, R0
7038 057142 104417 TRAP C$PNTF
7039 057144 062706 000004 ADD #4, SP
7040
7041 057150 112737 177777 003411 MOVB #-1, P$GDBD ;SET ERROR FLAG
7042 057156 023727 017404 000001 4$: CMP MLTYP, #TTL ;IS LOOP CABLE OR REMOTE
7043 057164 003413 BLE 5$ ;BRANCH IF INT OR NONE
7044 057166 PRINTF #SHTLP ;PRINT ERROR LOOP MUST BE INT FOR MTP
7045 057166 012746 026227 MOV #SHTLP, -(SP)
7046 057172 012746 000001 MOV #1, -(SP)
7047 057176 010600 MOV SP, R0
7048 057200 104417 TRAP C$PNTF
7049 057202 062706 000004 ADD #4, SP
7050 057204 112737 177777 003411 MOVB #-1, P$GDBD ;SET ERROR FLAG
7051 057214 02737 000001 017404 5$: CMP #TTL, MLTYP ;IS IT INTERNAL
7052 057222 0057 BNE 10$ ;IF NOT THEN CHECK COMPARE TOTALS
7053 057224 032737 000002 023100 BIT #TRBB, DEVPAR ;IS THIS CONTROL OR TRIB
7054 057232 001013 BNE 6$ ;BRANCH IF CONTROL
7055 057234 PRINTF #SHTLPB ;PRINT ERROR MUST BE CONTROL
7056 057234 012746 026367 MOV #SHTLPB, -(SP)
7057 057240 012746 000001 MOV #1, -(SP)
7058 057244 010600 MOV SP, R0
7059 057246 104417 TRAP C$PNTF
7060 057250 062706 000004 ADD #4, SP
7061 057254 112737 177777 003411 MOVB #-1, P$GDBD ;SET ERROR FLAG
7062 057262 022737 000001 015754 6$: CMP #1, TRBTOT ;IS TRIB TOATAL = 1
7063 057270 001011 BNE 7$ ;BRANCH IF MORE
7064 057272 012737 177777 015762 MOV #-1, INDEX
7065 057300 004737 046460 JSR PC, GTVIND ;GET TRIBN WITH ADDRESS
7066 057304 022737 000001 015756 CMP #1, TRIBN
7067 057312 001423 BEQ 10$ ;OK IF ADD 1

```

```

7068 057314           7$: PRINTF #SHTLPC
7069 057314 012746 026431          MOV #SHTLPC, -(SP)
7070 057320 012746 000001          MOV #1, -(SP)
7071 057324 010600          MOV SP, R0
7072 057326 104417          TRAP C$PNTF
7073 057330 062706 000004          ADD #4, SP
7074 057334           PRINTF #SHTLPD          :PRINT ERROR ---
7075 057334 012746 026502          MOV #SHTLPD, -(SP)
7076 057340 012746 000001          MOV #1, -(SP)
7077 057344 010600          MOV SP, R0
7078 057346 104417          TRAP C$PNTF
7079 057350 062706 000004          ADD #4, SP
7080 057354 112737 177777 003411  10$: MOVB #-1, P$GDBD          :SET ERROR FLAG
7081 057362 105737 003411          TSTB P$GDBD          :TEST ERROR FLAG
7082 057366 001043          BNE 12$             :BRANCH IF ERROR
7083 057370 013737 017244 017334  MOV CTOTCC, CURCC   :MAKE CURRENT COUNT= COMPARE COUNT
7084 057376 005737 017334          TST CURCC          :TEST TOTAL COMPARE COUNT
7085 057402 001003          BNE 1$             :BRANCH IF NON DEFAULT
7086 057404 012737 000072 017334  1$: MOV #58, CURCC       :SET UP DEFAULT
7087 057412 032737 000002 017410  BIT #DATCKB, PARAM
7088 057420 001430          BEQ 2$             :BRANCH IF NOT CHECKING
7089 057422 013737 015754 017232  MOV TRBTOT, MPLY
7090 057430 013737 017334 017350  MOV CURCC, TEMP
7091 057436 005037 017354          CLR TEMP2
7092 057442 004737 046434          JSR PC, MTPLY      :MULTIPLY TRBTOT BY CURCC
7093                                     :RESULT IN TEMP2
7094 057446 022737 004000 017354  CMP #RBFLIM, TEMP2  :IS IT MUCH TO MUCH
7095 057454 002012          BGE 2$             :NO EVERTHING IS HUNKY DORY
7096 057456          PRINTF #SHTBR      :ERROR
7097 057456 012746 026603          MOV #SHTBR, -(SP)
7098 057462 012746 000001          MOV #1, -(SP)
7099 057466 010600          MOV SP, R0
7100 057470 104417          TRAP C$PNTF
7101 057472 062706 000004          ADD #4, SP
7102 057476 000137 052444          12$: JMP GETCL          :GO BACK TO GET NEW COMMAND
7103 057502 012737 011416 017236  2$: MOV #PTRTAB, TXPTR  :INIT TRANSMIT MESSAGE POINTER
7104 057510 012737 011512 017240  MOV #PTR13, CMPPTR  :INIT COMPARE MESSAGE POINTER
7105 057516 012737 011606 017234  MOV #PTR23, RXPTR   :INIT RECEIVE MESSAGE POINTER
7106
7107 057524 013737 017242 017276  MOV CMPTOT, RXMTOT  :MAKE COMPARE AND RX MESSAGE COUNTS EQUAL
7108 057532 032737 000002 017410  BIT #DATCKB, PARAM  :IS IT DATA CHECK
7109 057540 001003          BNE GTREX          :BRANCH IF CHECKING
7110 057542 012737 000001 017276  MOV #1, RXMTOT      :IF NOCHK MAKE RXCOUNT =1
7111 057550 005037 017320          GTREX: CLR CLNSET
7112 057554 005037 015764          CLR CTX
7113 057560 005037 015766          CLR CRX
7114 057564 012737 016012 015770  MOV #RXSTAK, RSPTRS
7115 057572 012737 016012 015772  MOV #RXSTAK, RSPTRE
7116 057600 012737 015776 015774  MOV #TXSTAK, TSPTR  :SET UP INT STAK POINTERS
7117 057606 005037 017414          CLR FLAG          :CLEAR FLAG
7118 057612 005037 017302          CLR OPVAR         :CLEAR NO BUFFER COUNTER
7119 057616 005037 017304          CLR OPVAR1        :CLEAR OPVAR1
7120 057622 005037 017306          CLR PSCNT         :CLEAR PASS COUNT
7121 057626 005037 017310          CLR ERRCNT        :CLEAR ERROR COUNT
7122 057632 005037 017300          CLR LNCNT         :CLEAR COUNTER THAT IS USED FOR STATUS
7123 057636 004737 042120          JSR PC, LOGDVI     :LOG ABOUT TO INIT DEVICE

```

```

7124 057642 004737 062720      JSR      PC,DVINIT      ;INIT DEVICE
7125
7126 057646 012737 177777 015762 GTRX2:  MOV      #-1,INDEX      ;MAKE INDEX =-1
7127 057654 013737 017334 017350 GTRX2C: MOV      CURCC,TEMP
7128 057662 032737 000001 023100      BIT      #MTP,DEVPAR
7129 057670 001404      BEQ      GTRX22          ;IF NOT MULTI GO TO 22
7130 057672 032737 000002 017410      BIT      #DATCKB,PARAM ;IS THERE DATA CHECKING
7131 057700 001005      BNE      GTRX2A          ;BRANCH IF CHECKING
7132 057702 012737 001000 017334 GTRX22: MOV      #BUFLIM,CURCC ;SET UP CHAR COUNT TO 'BUFLIM'
7133 057710 005037 017350      CLR      TEMP
7134 057714 004737 046460      GTRX2A: JSR      PC,GTVIND    ;GET VALID INDEX
7135 057720 022737 000040 015762      CMP      #32.,INDEX     ;IS IT 32
7136 057726 001423      BEQ      GTRX2B          ;YES.. ALL DONE GO EXECUTE MODE
7137
7138      ;GET RXBUFF PTR FIGURE
7139
7140 057730 012737 005416 017354      MOV      #RXBUF,TEMP2   ;TEMP = 0 FOR PTP OR MTP/W NO CHK
7141 057736 013737 015762 017232      MOV      INDEX,MPLY     ;INDEX X TEMP+ RXBUF ADDR =
7142 057744 004737 046434      JSR      PC,MTPLY       ;NEW RXBUF ADDR.
7143 057750 013737 017354 017342      MOV      TEMP2,CURADD   ;SET UP RX BUFFER ADDRESS
7144
7145      ;GET CURRENT POINTER FIGURE
7146
7147 057756 004737 046544      JSR      PC,GRPTCP
7148
7149      ;GO LOAD '33' TO BUFFER
7150
7151 057762 012737 000010 017332      MOV      #10,MSGTYP     ;SET UP FOR 33 TO FILL RX BUFFERS
7152 057770 004737 045444      JSR      PC,BLDBUF      ;CLEAR RX BUFFER
7153 057774 000727      BR       GTRX2C          ;GO BACK FOR MORE
7154 057776 013702 017402      GTRX2B: MOV      MODTYP,R2
7155 060002 006302      ASL      R2
7156 060004 000172 017416      JMP      @MODE(R2)      ;MODE DISPATCH
7157
    
```


7158
7159
7160
7161
7162
7163
7164
7165
7166
7167
7168
7169
7170
7171
7172
7173
7174
7175
7176
7177
7178
7179

```

.SBTTL          RECEIVE MODE SECTION
++
: FUNCTIONAL DESCRIPTION:
: RECEIVE-ONLY (OR ONE-WAY-IN) ROUTINE
: IN THIS MODE OF TESTING THE DEVICE'S RECEIVER IS ENABLED IN EXPECTATION
: OF RECEIVING A MESSAGE. AFTER RECEIVING AN 'EXPECTED' NUMBER OF
: MESSAGES, THE DATA RECEIVED CAN BE COMPARED AGAINST A LIST OF 'EXPECT
: TO RECEIVE' MESSAGES IF DATA-CHECKING IS ENABLED.
:
: SUBORDINATE ROUTINES USED:
: 'ALLTR'
:
: CALLING SEQUENCE:
:--
: JMP @MODE(R2) ;DISPATCH TO MODE BASED ON MODE TYPE IN R2
:
: RXONLY: BIS #QRX+RERX,FLAG ;SET UP RX QUE
: JSR PC,LCPRLS ;LOAD CPTLS (RX PTRS)
: JSR PC,RXQUAL ;GO QUE ALL VALID RX'S
: RXON3: CLR CPTR
: JMP ALLTR ;GO RX.

```

060010	052737	000104	017414
060016	004737	046232	
060022	004737	046162	
060026	005037	017340	
060032	000137	060162	

```

7180          .SBTTL          TRANSMIT MODE SECTION
7181
7182          :++
7183          : FUNCTIONAL DESCRIPTION:
7184          : TRANSMIT-ONLY (OR ONE-WAY-OUT) ROUTINE
7185          : IN THIS MODE OF TESTING A LIST OF MESSAGES IS TRANSMITTED WITHOUT
7186          : EXPECTING ANY DATA TO BE RECEIVED. A REPETITION COUNT CAN BE
7187          : SPECIFIED TO REPETITIVELY TRANSMIT THE LIST.
7188
7189          : SUBORDINATE ROUTINES USED:
7190          :         'ALLTR'
7191
7192          : CALLING SEQUENCE:
7193          :         JMP         @MODE(R2)          ;DISPATCH TO MODE BASED ON MODE TYPE IN R2
7194          :--
7195
7196 060036 042737 000002 017410 TXONLY: BIC      #DATCKB,PARAM ;SET NOCHECK.
7197 060044 004737 046362          TXON2: JSR      PC,LCPTLS ;LOAD TX POINTERS AND TX COUNTS
7198 060050 052737 000210 017414          BIS      #QTX+#ETX,FLAG ;SET THE QUE TX FLAG
7199 060056 004737 046336          JSR      PC,CLRPLS ;CLEAR RXPRT LIST
7200 060062 012737 000040 015762          MOV      #32.,INDEX
7201 060070 000137 060162          JMP      ALLTR      ;GO TX.
7202

```

7203
7204
7205
7206
7207
7208
7209
7210
7211
7212
7213
7214
7215
7216
7217
7218
7219
7220
7221
7222
7223
7224
7225
7226

.SBTTL PASSIVE MODE SECTION

..++
: FUNCTIONAL DESCRIPTION:
: PASSIVE MODE SECTION
: IN THIS MODE OF TESTING, THE DEVICE'S RECEIVER IS ENABLED IN
: EXPECTATION OF RECEIVING A MESSAGE. THEN EVERY TIME A MESSAGE IS
: RECEIVED, A MESSAGE IS TRANSMITTED. DATA CHECKING CAN BE DONE ON THE
: RECEIVED DATA.

: SUBORDINATE ROUTINES USED:

.. 'ALLTR'

: CALLING SEQUENCE:

.. JMP @MODE(R2) ;DISPATCH TO MODE BASED ON MODE TYPE IN R2

PLCK: JSR PC,LCPTLS ;LOAD TX POINTERS AND TX COUNTS
JSR PC,LCPRLS ;SET UP CPTRR TO REC POINTERS
BIS #QRX+PERX,FLAG ;SET UP Q AND EXPECT RX
JSR PC,RXQUAL ;QUE ALL
JMP ALLTR ;AND GO RX FIRST MSG.

060074 004737 046362
060100 004737 046232
060104 052737 000104 017414
060112 004737 046162
060116 000137 060162

7227
 7228
 7229
 7230
 7231
 7232
 7233
 7234
 7235
 7236
 7237
 7238
 7239
 7240
 7241
 7242
 7243
 7244
 7245
 7246 060122
 7247 060122 032737 000002 017410
 7248 060130 001003
 7249 060132 013737 017260 017276
 7250 060140 004737 046362
 7251 060144 004737 046232
 7252 060150 052737 000314 017414
 7253 060156 004737 046162
 7254

```
.SBTTL      ACTIVE MODE SECTION

:++
: FUNCTIONAL DESCRIPTION:
: ACTIVE MODE SECTION
: IN THIS MODE OF TESTING A LIST OF MESSAGES IS TRANSMITTED AND
: MESSAGES ARE EXPECTED TO BE RECEIVED. RECEIVED DATA CAN BE COMPARED
: AGAINST 'EXPECTED' DATA IF DATA-CHECKING IS ENABLED.
: NOTE: IF BOTH ENDS OF THE LINK ARE IN ACTIVE MODE, THEN THE
: LINK MUST BE A FULL DUPLEX LINK!

: SUBORDINATE ROUTINES USED:
:
: 'ALLTR'

: CALLING SEQUENCE:
: JMP @MODE(R2) ;DISPATCH TO MODE BASED ON MODE TYPE IN R2
:--

ALCK:
BIT #DATCKB,PARAM ;IS IT DATA CHECK
BNE 1$ ;BRANCH IF CHECK
MOV TXMTOT,RXMTOT ;IF NOCH MAKE RX=TX
1$: JSR PC,LCPTLS ;LOAD TX POINTERS AND COUNTS
JSR PC,LCPRLS ;LOAD RX POINTERS
BIS #QRX+#QTX+#ETX+#ERX,FLAG
JSR PC,RXQUAL ;QUE UP 1 RX BUFFER FOR ALL VOID
```

7255
7256
7257
7258
7259
7260
7261
7262
7263
7264
7265
7266
7267
7268
7269
7270
7271
7272
7273
7274
7275
7276
7277
7278
7279
7280
7281
7282
7283
7284
7285
7286
7287
7288
7289
7290
7291
7292
7293
7294
7295
7296
7297
7298
7299
7300
7301
7302
7303
7304
7305
7306
7307
7308
7309
7310

.SBTTL TRANSMIT - RECEIVE FOR ALL STANDARD MODES

..**
: FUNCTIONAL DESCRIPTION:
: THIS CODE PERFORMS THE FOLLOWING FUNCTIONS
: 1.) IF RX BUFFERS ARE TO BE QUED, TELL DEVICE
: CODE TO QUE THEM, LOG RECEIVE QUED.
: 2.) IF TX BUFFERS ARE TO BE QUED, TELL DEVICE
: CODE TO QUE THEM, LOG TRANSMIT QUED.
: 3.) WAIT FOR EITHER RECEIVE BUFFER OR TRANSMIT BUFFER OR
: BOTH TO COMPLETE
: 4.) IF RECEIVE COMPLETE LOG IT UPDATE RX TABLE IF DATA
: CHECKING.
: 5.) IF TRANSMIT COMPLETE LOG IT.
: 6.) WHEN BOTH TRANSMIT AND RECEIVE LISTS ARE DONE
: GO TO THE COMPARE BUFFER CODE

: SUBORDINATE ROUTINES USED:
: 'DVRXQ' -QUE RECEIVE BUFFER SPACE TO DEVICE
: 'LOGRXQ' -LOG RECEIVE BUFFER SPACE TO EVENT LOG
: 'LOGTXQ' -LOG TRANSMIT BUFFER QUED TO EVENT LOG
: 'DVTXRX' -QUE TRANSMIT BUFFER AND WAIT FOR RX
: OR TX TO COMPLETE
: 'LOGRXC' -LOG RECEIVE BUFFER COMPLETED TO EVENT LOG
: 'LOGTXC' -LOG TRANSMIT BUFFER COMPLETED TO EVENT LOG

: USE OF FLAG BITS:
: QRX - SET ON INPUT TO ALLTR IF REC IS TO BE QUED TO
: DEVICE. CLEARED BY DVRXQ AND THEN SET BY DVTXRX
: WHEN RX BUFFER IS COMPLETED.
: QTX - SET ON INPUT TO ALLTR IF TRANSMIT IS TO BE QUED TO
: DEVICE. CLEARED ON ENTRY TO DVTXRX AND SET BY DVTXRX
: WHEN TX BUFFER IS COMPLETED.
: ETX - USED BY DVTXRX TO DETERMINE IF TX BUFFER COMPLETED IS
: EXPECTED.
: ERX - USED BY DVTXRX TO DETERMINE IF RX BUFFER COMPLETED IS
: EXPECTED.

: CALLING SEQUENCE:
: JMP ALLTR ;GO TO TRANSMIT-RECEIVE FOR ALL STANDARD MODES
:--

060162	032737	000004	017414	ALLTR:			
060162	001406			ALCK5:	BIT	#QRX, FLAG	
060170	004737	046524			BEQ	ALCK1	:IF NOT RX GO TO TX'S
060172	004737	047300		ALCK5B:	JSR	PC, ULRPLS	:GET RX INDEX
060176	004737	046504			JSR	PC, LOGAQP	:LOG AND QUE REC.
060202	004737	046504			JSR	PC, LDRPLS	:RESTORE RX PTR TO LIST
060206	032737	000010	017414	ALCK1:	BIT	#QTX, FLAG	
060214	001422				BEQ	ALCK2	:IF NO TX'S GO TO 2
060216	004737	047100			JSR	PC, GNTXPR	
060222	013702	017340			MOV	CPTR, R2	
060226	011237	017354			MOV	(R2), TEMP2	
060232	012237	017250			MOV	(R2)+, DVTXA	

```

7311 060236 011237 017356      MOV      (R2),TEMP3
7312 060242 012237 017252      MOV      (R2)+,DVTCC
7313 060246 010237 017340      MOV      R2,CPTR
7314 060252 004737 046642      JSR      PC,LDTPLS      ;RELOAD LIST
7315 060256 004737 042020      JSR      PC,LOGTXQ
7316
7317 060262 004737 064020      ALCK2:  JSR      PC,DVTXRX      ;GO TO TX AND RX SUB ROUT.
7318
7319 060266 032737 000004 017414      BIT      #QRX,FLAG      ;CHECK FOR REC. MSG.
7320 060274 001532
7321 060276 013737 017270 017354      MOV      DVRXA,TEMP2
7322 060304 013737 017272 017356      MOV      DVRCC,TEMP3
7323 060312 013737 017266 015756      MOV      DVRTB,TRIBN
7324 060320 004737 042072      JSR      PC,LOGRXC      ;LOG REC COMPLETE
7325 060324 032737 000004 017410  UPTABL: BIT      #ECHOB,PARAM      ;IS THIS ECHO MODE(PASSIVE)
7326 060332 001410      BEQ      UPTA4          ;IF NOT GO TO 4
7327 060334 004737 046662      JSR      PC,ULTPLS
7328 060340 013702 017340      MOV      CPTR,R2          ;ELSE SET R2 TO PRESENT TX TABLE
7329 060344 013722 017354      MOV      TEMP2,(R2)+      ;STORE OFF RX ADD
7330 060350 013712 017356      MOV      TEMP3,(R2)      ;AND CC
7331 060354 032737 000002 017410  UPTA4: BIT      #DATCKB,PARAM      ;IS DATA CHECKING ASKED FOR
7332 060362 001012      BNE      UPTA1          ;IF SO GO TO UPTA1
7333 060364 004737 047152      JSR      PC,GETIND      ;GET INDEX
7334 060370 004737 046264      JSR      PC,LCPL1      ;RESTORE POINTER
7335 060374 013737 017354 017336      MOV      TEMP2,CPTRR      ;RESTORE POINTER
7336 060402 004737 046504      JSR      PC,LDRPLS      ;LOAD COUNT AND LIST
7337 060406 000430      BR
7338
7339 060410 004737 046524      UPTA1:  JSR      PC,ULRPLS      ;GET PTR FROM LIST
7340 060414 013702 017336      MOV      CPTRR,R2
7341 060420 011237 017350      MOV      (R2),TEMP
7342 060424 163737 017356 017350      SUB      TEMP3,TEMP      ;LOAD TEMP WITH PREV. COUNT
7343 060432 013722 017356      MOV      TEMP3,(R2)+      ;LOAD TEMP WITH PREV.COUNT-CURRENT
7344 060436 063737 017356 017354      ADD      TEMP3,TEMP2
7345 060444 013722 017354      MOV      TEMP2,(R2)+      ;STORE OF NEW ADD
7346 060450 013712 017350      MOV      TEMP,(R2)      ;AND NEW CC
7347 060454 162702 000002      SUB      #2,R2          ;PUT POINTER BACK TO ADDR.
7348 060460 010237 017336      MOV      R2,CPTRR      ;AND RESTORE IT.
7349 060464 004737 046504      JSR      PC,LDRPLS
7350 060470
7351 060470 022737 000002 017402  UPTEX:  CMP      #PAS,MODTYP
7352 060476 001011      BNE      ALCK2A
7353 060500 005337 015762      DEC      INDEX          ;IF NOT PASSIVE LOOP THEN GO TO 2A
7354 060504 042737 000004 017414      BIC      #QRX,FLAG      ;IF PASSIVE NEXT TXQ WILL BE FOR THIS TRIB
7355 060512 052737 000210 017414      BIS      #QTX+#ETX,FLAG ;CLEAR BOTH EXPECTED AND COMPLETED FLAGS
7356 060520 000632      BR          ;SET THE TX FLAGS
7357
7358 060522 004737 046622      ALCK2A: JSR      PC,ULRCLS      ;GET COUNT
7359 060526 005337 017274      DEC      DVRCT          ;DEC REC COUNT
7360 060532 004737 046602      JSR      PC,LDRCLS      ;RESTORE COUNT
7361 060536 005737 017274      TST     DVRCT          ;IS IT ALL DONE
7362 060542 001007      BNE      ALCK3          ;NO. GO CHECK TX
7363 060544 042737 000004 017414      BIC      #QRX,FLAG      ;CLEAR THE RX FLAG
7364 060552 005037 017336      CLR     CPTRR          ;YES. CLEAR POINTER
7365 060556 004737 046504      JSR      PC,LDRPLS      ;AND RELOAD LIST
7366 060562 032737 000010 017414  ALCK3:  BIT      #QTX,FLAG      ;IS IT TX

```

```

7367 060570 001467          BEQ      ALCK4          ;IF NOT TX THEN GO BACK
7368 060572 013737 017250 017354  MOV     DVTXA,TEMP2
7369 060600 013737 017252 017356  MOV     DVTCC,TEMP3      ;LOG TX COMPLETED
7370 060606 013737 017254 015756  MOV     DVTTB,TRIBN
7371 060614 004737 042036          JSR     PC,LOGTXC
7372 060620 004737 046722          JSR     PC,ULTCLS        ;GET COUNT TO DVTCT
7373 060624 005337 017256          DEC     DVTCT           ;DEC TX COUNT
7374 060630 004737 046702          JSR     PC,LDTCLS        ;AND RELOAD LIST
7375 060634 022737 000002 017402  CMP     #PAS,MODTYP
7376 060642 001020          BNE     ALCK3A          ;IF NOT PASSIVE MODE GO TO 3A
7377 060644 042737 000010 017414  BIC     #QTX,FLAG       ;CLEAR THE TX FLAGS
7378 060652 005737 017256          TST     DVTCT
7379 060656 001403          BEQ     ALCK3D          ;IF NO MORE MESG TO RX FOR THIS TRIB
7380                                ;EXIT WITHOUT RESETTING QRX
7381 060660 052737 000104 017414  BIS     #QRX+ERX,FLAG   ;AND SET THE RX FLAGS
7382 060666 004737 047014          ALCK3D: JSR    PC,GATCFL
7383 060672 005737 017256          TST     DVTCT
7384 060676 001007          BNE     ALCK3C          ;IF MORE TX'S TO IT
7385 060700 000137 060774          JMP     CMPSR           ;ELSE COMPARE
7386 060704 004737 047014          ALCK3A: JSR    PC,GATCFL ;GET ALL TX COUNTS FROM LIST
7387 060710 005737 017256          TST     DVTCT           ;IS IT ALL DONE
7388 060714 001404          BEQ     ALCK3B          ;IF NOT GO BACK TO 5
7389 060716 004737 047152          ALCK3C: JSR    PC,GETIND
7390 060722 000137 060162          JMP     ALCK5
7391 060726 005037 017340          ALCK3B: CLR    CPTR
7392 060732 042737 000010 017414  BIC     #QTX,FLAG       ;CLEAR POINTER
7393 060740 032737 000002 017410  BIT     #DATCKB,PARAM   ;CLEAR TX FLAG
7394 060746 001405          BEQ     ALCK4A          ;IS IT DAT CHECK
7395 060750 004737 046742          ALCK4:  JSR     PC,GARPFL ;IF NOT THEN END WO CKING RX.
7396 060754 005737 017336          TST     CPTRR
7397 060760 001356          BNE     ALCK3C          ;IF SOME RX'S LEFT GO BACK
7398 060762 005737 017340          ALCK4A: TST     CPTR
7399 060766 001402          BEQ     CMPSR           ;BRANCH IF ANY TX'S LEFT
7400 060770 000137 060262          JMP     ALCK2
7401
7402
7403
7404

```

7405 .SBTTL DATA COMPARISON CODE

7406
7407
7408
7409
7410
7411
7412
7413
7414
7415
7416
7417
7418
7419
7420
7421
7422
7423
7424
7425
7426
7427
7428
7429
7430
7431
7432
7433
7434
7435
7436
7437
7438
7439
7440
7441
7442
7443
7444
7445
7446
7447
7448
7449
7450
7451
7452
7453
7454
7455
7456
7457
7458
7459
7460

++
FUNCTIONAL DESCRIPTION:

CMPSR - COMPARE CODE
 THIS CODE COMPARES THE RECEIVED DATA AGAINST THE
 EXPECTED AND FILLS THE EVENT LOG WITH 1 OF 3 MSGS.

NOTE: IF NO DATA CHECKING SKIP THIS CODE

- 1.) A DATA COMPARISON ENTRY WHICH REPORTS THE NUMBER OF COMPARISON ERRORS FOUND.
 - 2.) A DATA COMPARISON ENTRY WHICH REPORTS DIFFERENCES IN REC LENGTH TO COMPARE LENGTH.
 - 3.) A DATA COMPARISON STARTED ENTRY WHICH REPORTS ADDRESS OF RECEIVE BUFFER AND BYTE COUNT.
- THIS CODE ALSO REPORTS SOFT ERRORS FOR DATA COMPARISON (THE FIRST 5 ONLY), LENGTH ERROR, AND TOTAL NUMBER OF ERRORS

SUBORDINATE ROUTINES USED:

'LOGCMP' - SEE ITEM 3 ABOVE
 'LOGCML' - SEE ITEM 2 ABOVE
 'LOGCMD' - SEE ITEM 1 ABOVE

CALLING SEQUENCE:

JMP CMPSR ; JUMP TO DATA COMPARISON CODE

```

CMPSR: BIT #DATCKB,PARAM ;IS DATA CHECKING TO BE DONE
        BEQ CMPSEX ;IF NOT THEN EXIT
CMPNEW: MOV #-1,INDEX
        JSR PC,GTVIND
        CMP #32.,INDEX
        BEQ CMPSEX ;END IF NO MORE TRIBS
        JSR PC,GRPTCP
        MOV CMPPTR,CPTRR ; AND START OF COMPARE POINTS TO CPTRR
        MOV RXMTOT,DVRCT
CMPS3: MOV CPTR,R2 ;MOVE CURRET RX PT.TO R2
        MOV (R2),TEMP2 ;MOVE RX ADD TO EVENT LOG
        MOV (R2)+,R1 ;SET R1 TO START ADD OF RX
        MOV (R2)+,TEMP3 ;SET CHAR COUNT TO EVENT LOG
        MOV R2,CPTR ;RESTORE RX POINT
        MOV CPTRR,R2 ;PUT R2 AT COMPARE TABLE
        MOV (R2)+,R3 ;SET R3 TO COMPARE ADD
        MOV (R2)+,R4 ;SET R4 TO COMP CC
        MOV R2,CPTRR ;RESTORE POINTER
        MOV R4,TEMP4
        JSR PC,LOGCMP ;LOG COMPARE START.
  
```



```

7461
7462 061114 020437 017356      CMP      R4,TEMP3      ;IS COMPARE COUNT = TO RX COUNT
7463 061120 001410              BEQ      CMPS7          ;IF SO GO TO 7
7464 061122 005237 017310      INC      ERRCNT
7465 061126              ERRSOFT 1,EDDLE,ERR10 ;PRINT ERROR
7466 061126 104457              TRAP    C$ERSOFT
7467 061130 000001              .WORD  1
7468 061132 030220              .WORD  EDDLE
7469 061134 041426              .WORD  ERR10
7470 061136 004737 042204      JSR      PC,LOGCML     ;LOG LENGTH ERROR
7471
7472 061142 005037 017360      CMPS7:  CL      TEMP4   ;CLEAR BAD BYTE COUNTER
7473 061146 012737 000001 017346  MOV      #1,OFSET      ;SET OFSET BYTE COUNT TO 1
7474 061154 122123              CMPS1:  CMPB     (R1)+,(R3)+ ;COMPARE RX WITH EXPETED
7475 061156 001422              BEQ      CMPS6          ;IF EQUAL THEN GO TO 6
7476
7477 061160 005237 017360      CMPS2:  INC      TEMP4   ;INC BAD COUNT
7478 061164 023727 017360 000005  CMP      TEMP4,#5      ;IS IT MORE THEN 5
7479 061172 101014              BHI     CMPS6          ;IF SO GO FOR MORE
7480 061174 114337 017370      MOVB    -(R3),GOOD    ;STORE GOOD BYTE FOR ERROR
7481 061200 114137 017371      MOVB    -(R1),BAD     ;STORE BAD BYTE FOR ERROR
7482 061204 005237 017310      INC      ERRCNT
7483 061210              ERRSOFT 2,EDDDE,ERR1 ;REPORT COMPARISON FAILURE TO OPR.
7484 061210 104457              TRAP    C$ERSOFT
7485 061212 000002              .WORD  2
7486 061214 030255              .WORD  EDDDE
7487 061216 041336              .WORD  ERR1
7488 061220 005201              INC      R1
7489 061222 005203              INC      R3
7490 061224 005237 017346      CMPS6:  INC      OFSET   ;INC OFFSET
7491 061230 005304              DEC      R4            ;ELSE DEC CHAR COUNT AND SEE IF 0
7492 061232 001350              BNE     CMPS1          ;IF NOT GO BACK
7493 061234 005737 017360      TST     TEMP4         ;SEE IF ANY CMP ERRS FOR THIS MSG
7494 061240 001410              BEQ     CMPS5A         ;BR IF NONE
7495 061242 005237 017310      INC      ERRCNT
7496 061246              ERRSOFT 3,EDDDE,ERR2 ;REPORT # OF MISMATCHES FOR MESSAGE
7497 061246 104457              TRAP    C$ERSOFT
7498 061250 000003              .WORD  3
7499 061252 030255              .WORD  EDDDE
7500 061254 041400              .WORD  ERR2
7501 061256 004737 042222      CMPS5:  JSR      PC,LOGCMD ;LOG DATA ERROR IN COMPARE
7502 061262              CMPS5A:
7503 061262 005337 017274      DEC      DVRCT
7504 061266 001267              BNE     CMPS3
7505 061270 000137 061012      JMP     CMPNEW
7506
  
```

7507 .SBTTL INTERNAL END OF PASS CODE

7508
7509
7510
7511
7512
7513
7514
7515
7516
7517
7518
7519
7520
7521
7522
7523
7524
7525
7526
7527
7528
7529
7530
7531
7532
7533
7534
7535
7536
7537
7538
7539
7540

..**
: FUNCTIONAL DESCRIPTION:
: THIS CODE INCREMENTS THE PASS COUNT FOR THE
: EVENT LOG. LOGS THE END OF PASS EVENT
: IF 'RPASS' IS A MINUS ONE RETURN TO MODE
: DISPATCHER. IF NOT -1 THEN DECREMENT RPASS
: AND IF 'RPASS' IS THEN = TO 0 GO TO DCLT PROMPT
: IN NOT = TO 0 THEN GO BACK TO MODE DISPATCHER

: SUBORDINATE ROUTINES USED:
:-----
: 'LOGEOP' - LOG END OF PASS TO EVENT LOG

CMPSEX: INC PSCNT ;BUMP PASS COUNT
MOV OPVAR1,TEMP5 ;LOG TX THRES
MOV OPVAR,TEMP4 ;LOG RX THRESH
MOV PSCNT,TEMP2 ;LOG PASS COUNT
MOV ERRCNT,TEMP3
JSR PC,LOGEOP ;LOG END OF PASS
CMP #-1,RPASS ;SEE IF RPASS=-1
BEQ 1\$;IF IT IS DON'T DECREMENT, LOOP FOREVER
DEC RPASS ;DEC PASS COUNT
BEQ 2\$;IF DONE EXIT TEST
1\$: JMP GTRX2 ;ELSE GO BACK AND DISPATCH
2\$: JSR PC,HLTTRB ;GO HALT ALL TRIBS BEFORE GOING BACK
JMP GTRA5 ;WHEN RPASS=0 GO BACK TO 'DCLT>'

7541
7542
7543
7544
7545
7546
7547
7548
7549
7550
7551
7552
7553
7554
7555
7556
7557
7558
7559
7560
7561
7562
7563
7564
7565
7566
7567
7568
7569
7570
7571
7572
7573
7574
7575
7576
7577
7578
7579
7580
7581
7582
7583
7584
7585
7586
7587
7588
7589
7590
7591
7592
7593
7594
7595
7596

.SBTTL DOWN-LINE-LOAD SECTION

..**
: FUNCTIONAL DESCRIPTION:
: DOWN-LINE-LOAD SECTION
: IN THIS MODE OF TESTING THE 'HOST' OR ORIGINATING STATION
: REQUESTS THE 'SATELLITE' OR BOOT STATION TO ENTER MOP MODE.
: THE BOOT STATION THEN SENDS A 'REQUEST PROGRAM MESSAGE'.
: THE 'HOST' THEN SENDS A 'MEMORY LOAD WITH TRANSFER ADDRESS'
: THAT CONTAINS IMAGE DATA TO BE LOADED BY THE BOOT STATION'S
: DMP-11 MICROCODE STARTING AT LOC. 0. THIS IMAGE DATA WILL CONTAIN A
: PROGRAM THAT WILL PRINT A MSG THAT DOWN-LINE-LOAD WAS SUCESSFUL.

: SUBORDINATE ROUTINES USED:
: 'DLTXRX' - SPECIAL TX RX ROUTINE FOR DLL
: 'DVRXQ' - QUE RX BUFFER SPACE TO DEVICE
: 'LOGRXQ' - LOG RX SPACE QUED TO EVENT LOG
: 'LOGTXQ' - LOG TX BUFFER QUED TO EVENT LOG
: 'DVTXRX' - QUE TX BUFFER AND WAIT FOR RX OR TX TO COMPLETE
: 'LOGTXC' - LOG TX COMPLETED TO EVENT LOG
: 'LOGRXC' - LOG RX COMPLETED TO EVENT LOG

: CALLING SEQUENCE:
: JMP @MODE(R2) ;DISPATCH TO MODE BASED ON MODE TYPE IN R2
:--

```
DLL:  MOV    #-1,INDEX
      JSR    PC,GTVIND      ;GET VALID INDEX ALSO FIRST TRIBN
      MOV    TRIBN,TEMP3    ;MOV TRIBN TO TEMP3 FOR MTP DEFAULT
      BIT    #MTP,DEVPAR    ;IS THIS MULITPOINT
      BNE    1$             ;IF SO BRANCH
      GMANID DLLQ1,TEMP3,0,377,0,377,NO
```

TRAP CS\$GMAN
BR 10004\$
.WORD TEMP3
.WORD T\$CODE
.WORD DLLQ1
.WORD 377
.WORD T\$LOLIM
.WORD T\$HILIM

10004\$:

```
1$:  MOVB   TEMP3,PASS1
      MOVB   TEMP3,PASS2
      MOVB   TEMP3,PASS3
      MOVB   TEMP3,PASS4
      RIS    #ERX,FLAG      ;SET EXPECTED TO RX
      BIC    #DATCKB,PARAM  ;CLEAR NOCHECK
      MOV    #DLLM1,CURADD  ;SET THE DOWN LINE LOAD MSG TO #1
      MOV    DLLM1C,CURCC   ;SET THE CC
      JSR    PC,DLTXRX      ;GO TO THE DOWN LINE TX RX ROUTINE
```

;RETURN WHEN TX AND RX ARE COMPLETED

```

7597
7598 061522 012737 002654 017342      MOV      #DLLM2,CURADD      ;SET THE DOWN LINE LOAD MSG TO #2
7599 061530 013737 002174 017334      MOV      DLLM2C,CURCC      ;SET CC
7600 061536 042737 001000 017414      BIC      #DLLGA,FLAG      ;CLEAR THE GO AHEAD FLAG
7601 061544 004737 061610                JSR      PC,DLTXRX        ;GO TO THE DOWN LINE TX RX ROUTINE
7602
7603                                ; RETURN WHEN TX AND RX ARE COMPLETED
7604 061550      DLLPRI:
7605 061550      PRINTF #DLLCM
7606 061550 012746 027156                MOV      #DLLCM,-(SP)
7607 061554 012746 000001                MOV      #1,-(SP)
7608 061560 010600                        MOV      SP,R0
7609 061562 104417                        TRAP    C$PNTF
7610 061564 062706 000004                ADD      #4,SP
7611 061570 000137 052354                JMP      GTRA5
7612
7613 061574      DLLEA:
7614 061574      ERRSOFT 13,DLLAB,ERR14
7615 061574 104457                        TRAP    C$ERSOFT
7616 061576 000015                        .WORD  13
7617 061600 040020                        .WORD  DLLAB
7618 061602 041510                        .WORD  ERR14
7619
7620 061604 000137 052354                JMP      GTRA5          ;PRINT ABORT AND EXIT
7621
7622
7623
7624 061610      DLTXX:
7625 061610 052737 000004 017414      BIS      #QRX,FLAG      ;SET THE QUE RX FLAG
7626 061616 012737 005416 017270      MOV      #RXBUF,DVRXA   ;SET THE DEVICE RX BUFFER TO RXBUF
7627 061624 012737 005416 017354      MOV      #RXBUF,TEMP2   ;SET UP FOR LOG
7628 061632 012737 000400 017272      MOV      #256.,DVRCC    ;SET UP FOR CC OF 256
7629 061640 012737 000400 017356      MOV      #256.,TEMP3    ;SET UP FOR LOG
7630 061646 004737 063736                JSR      PC,DVRXQ        ; GO QUE RX
7631 061652 004737 042054                JSR      PC,LOGRXQ       ;AND LOG IT...
7632
7633 061656 013737 017342 017250      MOV      CURADD,DVTXA   ;SET UP FOR TX
7634 061664 013737 017342 017354      MOV      CURADD,TEMP2   ;AND LOG
7635 061672 013737 017334 017252      MOV      CURCC,DVTCC    ;SE UP FOR TX COUNT
7636 061700 013737 017334 017356      MOV      CURCC,TEMP3    ;AND LOG IT
7637 061706 004737 042020                JSR      PC,LOGTXQ       ;LOG THE TX QUEUED
7638 061712 052737 000210 017414      BIS      #QTX+#ETX,FLAG ;SET UP TO QUE AND EXPECTED
7639 061720 004737 064020                JSR      PC,DVTXRX      ;GO TO DEVICE ROUTINE
7640 061724 032737 001000 017414      BIT      #DLLGA,FLAG    ;TEST FOR GO AHEAD BIT
7641 061732 001047                        BNE     DLLE1           ;IF SET GO TO ONE
7642 061734 032737 000010 017414      BIT      #QTX,FLAG      ;ELSE CHECK FOR TX DONE
7643 061742 001020                        BNE     DLLE6           ;IF DONE THEN BRANCH
7644                                ;ELSE ERROR
7645 061744 012737 041230 017366      MOV      #TXNC,CONOTM
7646 061752 013737 005416 017356      DLLE7:  MOV      RXBUF,TEMP3
7647 061760 013737 003416 017360      MOV      TXBUF,TEMP4
7648 061766 012737 040020 017354      DLLE7A: MOV      #DLLAB,TEMP2
7649 061774 004737 042102                JSR      PC,LGDVE        ;LOG ERROR
7650 062000 000137 061574                JMP      DLLEA          ;ABORT TEST
7651
7652 062004 013737 017250 017354      DLLE6:  MOV      DVTXA,TEMP2

```

```

7653 062012 013737 017252 017356      MOV      DVTCC,TEMP3
7654 062020 004737 042036              JSR      PC,LOGTXC      ;LOG TX DONE
7655 062024 042737 000210 017414      BIC      #QTX+#ETX,FLAG ;CLEAR QUE AND EXPECTED
7656 062032 052737 001000 017414      BIS      #DLLGA,FLAG   ;SET THE GO AHEAD BIT
7657 062040 023737 002174 017252      CMP      DLLM2C,DVTCC
7658 062046 001475              BEQ      DLLE5          ;EXIT IF SECOND MSG.
7659 062050 000723              BR       DLLE2          ;AND GO BACK TO 2
7660 062052 032737 000004 017414  DLLE1:  BIT      #QRX,FLAG    ;IS THE A RX COMPLETED
7661 062060 001004              BNE      DLLE8          ;IF SO GO TO 8
7662 062062 012737 041250 017366      MOV      #RXNC,CONOTM  ;ELSE SET UP ERROR AND ABORT.
7663 062070 000730              BR       DLLE7
7664 062072 013737 017270 017354  DLLE8:  MOV      DVRXA,TEMP2
7665 062100 013737 017272 017356      MOV      DVRCC,TEMP3
7666 062106 004737 042072              JSR      PC,LOGRXC      ;LOG RECEIVE COMPLETE
7667 062112 122737 000010 005416      CMPB    #10,RXBUF      ;CHECK FOR FIRST WORD OF RX
7668                                ;SEC BOOT MSG.
7669 062120 001404              BEQ      DLLE3
7670 062122 012737 041270 017366  DLLE4:  MOV      #RXM1,CONOTM ;SET UP MESG AND ABORT
7671 062130 000710              BR       DLLE7          ;ABORT TEST
7672
7673 062132 122737 000001 005420  DLLE3:  CMPB    #1,RXBUF+2    ;IS SECOND WORD 1 ?
7674 062140 001407              BEQ      DLLE5A        ;YES,BRANCH
7675 062142 012737 041313 017366      MOV      #RXM2,CONOTM
7676 062150 013737 005420 017356      MOV      RXBUF+2,TEMP3
7677 062156 000703              BR       DLLE7A        ;SET UP MESSAGE AND ABORT
7678
7679
7680                                ;;PRINT ID OF DEVICE REQUESTING LOAD REV B BY EC
7681 062160 012737 032060 017350  DLLE5A:  MOV      #UNKM,TEMP   ;SET UP FOR UNKNOWN DEVICE
7682 062166 113703 005417              MOVB    RXBUF+1,R3     ;GET DEVTYPE FROM MESSAGE
7683 062172 120327 000042              CMPB    R3,#34.        ;OUT OF LEGAL RANGE ?
7684 062176 101006              BHI     DLLE5B          ;YES,BRANCH
7685 062200 132703 000001              BITB    #1,R3          ;ODD ?
7686 062204 001003              BNE     DLLE5B          ;YES,BRANCH
7687 062206 016337 023220 017350      MOV      DLLIND(R3),TEMP ;GET ASCIZ MESSAGE FROM TABLE
7688
7689                                DLLE5B:  PRINTF  #SECRM,TEMP,R3 ;PRINT ID MESSAGE
7690 062214 010346
7691 062216 013746 017350              MOV     R3,-(SP)
7692 062222 012746 031722              MOV     TEMP,-(SP)
7693 062226 012746 000003              MOV     #SECRM,-(SP)
7694 062232 010600              MOV     #3,-(SP)
7695 062234 104417              MOV     SP,R0
7696 062236 062706 000010              TRAP   C$PNTF
7697                                ADD     #10,SP
7698
7699 062242 000207              DLLE5:  RTS     PC      ;RETURN TO CALLER
7700
7701
7702
    
```

7703
7704
7705
7706
7707
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728
7729
7730
7731
7732
7733
7734
7735
7736
7737
7738
7739
7740
7741
7742
7743
7744
7745
7746
7747
7748
7749
7750
7751
7752
7753
7754
7755
7756
7757
7758

.SBTTL TALK MODE SECTION

..**
: FUNCTIONAL DESCRIPTION:
: TALK MODE SECTION
: IN THIS MODE, THE "TALK" END OF THE LINK TRANSMITS OPERATOR
: SPECIFIED MESSAGES UNTIL A "EXIT" MESSAGE IS TYPE. AT THAT POINT,
: THIS END OF THE LINK GOES INTO "LISTEN" MODE.

: SUBORDINATE ROUTINES USED:
: "LOGTXQ" - LOG TX BUFFER QUED TO EVENT LOG
: "DVTXRX" - QUE TX BUFFER TO DEVICE AND WAIT FOR COMPLETE
: "LOGTXC" - LOG TX COMPLETE TO EVENT LOG

: CALLING SEQUENCE:
: JMP @MODE(R2) ;DISPATCH TO MODE BASED ON MODE TYPE IN R2

```

TALCK: MOV #-1,INDEX
        JSR PC,GTVIND ;GET FIRST TRIB
        BIC #DATCKB,PARA1 ;SET NOCHECK
1$: MOV #OPBUF,R2
     MOV #-1,(R2)+ ;CLEAR OUT OPBUFFER FIRST
     CMP #OPEND,R2
     BNE 1$
     GMANID OPRMM,OPBUF,A,0,1,72.,NO ;GET TALK MESSAGE
TRAP CS$GMAN
BR 10005$
.WORD OPBUF
.WORD T$CODE
.WORD OPRMM
.WORD 0
.WORD T$LOLIM
.WORD T$HILIM
10005$:
2$: CLR R2 ;NOW GET CHAR COUNT
     CMPB #377,OPBUF(R2)
     BEQ 3$
     INC R2
     BR 2$
3$: MOV R2,OPCNT
     MOV #OPBUF,DVTXA ;SET UP TX ADDR.
     MOV #OPBUF,TEMP2
     MOV OPCNT,TEMP3
     MOV OPCNT,DVTCC ;SET UP TX CC
     JSR PC,LOGTXQ
     BIS #QTX+#ETX,FLAG ;SET UP FLAGS
     CLR CPTRR ;CLEAR RX POINTER
     JSR PC,DVTXRX
     MOV DVTXA,TEMP2
     MOV DVTCC,TEMP3
     JSR PC,LOGTXC

```

062244 012737 177777 015762
062252 004737 046460
062256 042737 000002 017410
062264 012702 002524
062270 012722 177777
062274 022702 002646
062300 001373
062302 104443
062304 000406
062306 002524
062310 000142
062312 027112
062314 000000
062316 000001
062320 000110
062322 005002
062324 122762 000377 002524
062332 001402
062334 005202
062336 000772
062340 010237 002166
062344 012737 002524 017250
062352 012737 002524 017354
062360 013737 002166 017356
062366 013737 002166 017252
062374 004737 042020
062400 052737 000210 017414
062406 005037 017336
062412 004737 064020
062416 013737 017250 017354
062424 013737 017252 017356
062432 004737 042036

CZCLMBO DMP/V-11 DCLT MACY11 30A(1052) 28-JUL-81 13:35 PAGE 190
CZCLMB.P11 28-JUL-81 13:34 TALK MODE SECTION

SEQ 0189

7759	062436	022737	054105	002524	CMP	#'EX,OPBUF	:CHECK FOR EXIT
7760	062444	001277			BNE	TALCK	
7761	062446	022737	052111	002526	CMP	#'IT,OPBUF+2	
7762	062454	001273			BNE	TALCK	
7763	062456	042737	000210	017414	BIC	#QTX+#ETX,FLAG	:CLEAR THE TX BITS
7764	062464	012737	000006	017402	MOV	#LIS,MODTYP	:CHANGE TO LISTEN MODE
7765	062472	000137	057646		JMP	GTRX2	:AND GO BACK TO DISPATCH

7766
7767
7768
7769
7770
7771
7772
7773
7774
7775
7776
7777
7778
7779
7780
7781
7782
7783
7784
7785
7786
7787
7788
7789
7790
7791
7792
7793
7794
7795
7796
7797
7798
7799
7800
7801
7802
7803
7804
7805
7806
7807
7808
7809
7810
7811
7812
7813
7814
7815
7816
7817
7818
7819
7820
7821

.SBTTL LISTEN MODE SECTION

```

:++
: FUNCTIONAL DESCRIPTION:
: LISTEN MODE SECTION
: IN THIS MODE, THE 'LISTEN' END OF THE LINK PRINTS ALL OF THE MESSAGES
: RECEIVED BY THE DEVICE ON THE OPERATOR'S CONSOLE. IF THE MESSAGE
: RECEIVED IS AN 'EXIT' MESSAGE, THEN THE NODE ENTERS 'TALK' MODE.

```

SUBORDINATE ROUTINES USED:

```

'DVRXQ' - QUE RECEIVE BUFFER SPACE TO DEVICE
'LOGRXQ' - LOG RECEIVE BUFFER QUED TO EVENT LOG
'DVTXRX' - WAIT FOR RX TO COMPLETE
'LOGRXC' - LOG RX COMPLETE TO EVENT LOG

```

CALLING SEQUENCE:

```

JMP @MODE(R2) ;DISPATCH TO MODE BASED ON MODE TYPE IN R2

```

```

LISCK: MOV #-1,INDEX ;GET FIRST TRIB
      JSR PC,GTVIND ;CLEAR CHECK BIT
      BIC #DATCKB,PARAM ;PRINT PROMPT FOR OPR.
      PRINTF #LISP
      MOV #LISP,-(SP)
      MOV #1,-(SP)
      MOV SP,R0
      TRAP C$PNTF
      ADD #4,SP

LISCKA: MOV #OPBUF,DVRXA ;SET DEVICE UP TO REC AT OPBUF
      MOV #OPBUF,TEMP2
      MOV #82.,DVRCC ;SET UP CHAR COUNT TO 82.
      MOV #82.,TEMP3
      BIS #QRX+#ERX,FLAG ;SET UP FLAG
      CLR CPTR ;CLEAR THE TX.
      JSR PC,DVRXQ ;QUE RX
      JSR PC,LOGRXQ
      JSR PC,DVTXRX ;GO TO DEVICE RX. SUBROUTINE
      MOV DVRXA,TEMP2
      MOV DVRCC,TEMP3 ;SET UP ADDR.AND CC.
      JSR PC,LOGRXC ;LOG COMPLETED
      ADD DVRXA,DVRCC
      CLRB @DVRCC
      PRINTF #OPBFPT
      MOV #OPBFPT,-(SP)
      MOV #1,-(SP)
      MOV SP,R0
      TRAP C$PNTF
      ADD #4,SP

      CMP #'EX,OPBUF ;COMPARE FOR EX OF 'EXIT'
      BNE LISCKA ;IF NOT EXIT THEN GO BACK
      CMP #'IT,OPBUF+2 ;IF FIRST HALF OK CHECK NEXT PART
      BNE LISCKA ;IF NOT EXIT THE GO BACK

```


CZCLMBO DMP/V-11 DCLT MACY11 30A(1052) 28-JUL-81 13:35 PAGE 192
CZCLMB.P11 28-JUL-81 13:34 LISTEN MODE SECTION

SEQ 0191

7822 062706 012737 000005 017402
7823 062714 000137 057646
7824
7825

MOV #TAL,MODTYP
JMP GTRX2

:CHANGE MODE TO TALK
:RETURN TO DISPATCHER

7826 .SBTTL DEVICE FUNCTION SUBROUTINES

7827
7828
7829
7830

7831 .SBTTL DEVICE INIT SUBROUTINE

7832
7833
7834

7835 :++
7836 : FUNCTIONAL DESCRIPTION:
7837 : DVINIT- DEVICE INIT ROUTINE
7838 : THIS ROUTINE IS DEVICE DEPENDENT CODE THAT INITIS
7839 : THE DEVICE BEING TESTED.
7840 :
7841 : INPUTS: 'FHDPLX' INDICATES IF MODE IS FULL OR HALF DUPLEX. (1=FULL)
7842 : ADDRESS POINTERS (SELO,...) ALREADY POINT TO DEVICE'S REG.S
7843 :
7844 : SUBORDINATE ROUTINES USED:
7845 : 'LGDVE' - LOG DEVICE ERROR TO EVENT LOG
7846 :
7847 :
7848 : CALLING SEQUENCE:
7849 : JSR PC,DVINIT
7850 :--

7836
7837
7838
7839

7840
7841
7842
7843

7844
7845
7846
7847

7848
7849
7850
7851

7852 062720

DVINIT:

7853
7854
7855

7856
7857
7858
7859

7860
7861
7862
7863

7864
7865
7866
7867

7868
7869
7870
7871

7872
7873
7874
7875

7876
7877
7878
7879

7880
7881

012737 001000 017454
022737 000004 017402
001034
022737 000001 023100
001030
012777 060000 140074
005737 017454
001375
012737 037723 017354
017737 140054 017356
017737 140052 017360
004737 042102
005237 017310
104457
000016
037723
041456
000735
012777 040000 140014
022737 000004 023076
001005
112777 000200 140000
000240
000240
005777 137766
100426

MOV #1000,TIMER1 ;SET UP TIMER 1 FOR 1000(OCTAL) TICKS
CMP #DOW,MODTYP
BNE DVIN4 ;BRANCH IF NOT DLL
CMP #1,DEVPAR ;IS THIS TRIB
BNE DVIN4 ;BRANCH IF CONTROL OR PTP
MOV #60000,@SELO ;SET MCLR AND ENTER P MOP
DVIN4A: TST TIMER1
BNE DVIN4A ;IF TIMER RUNS OUT IT MEANS
MOV #DVEM8,TEMP2 ;SWITHCES ARE NOT SET CORRECTLY
MOV @SELO,TEMP3
MOV @SEL2,TEMP4
JSR PC,LGDVE
INC ERRCNT
ERRSOFT 14,DVEM8,ERR13

TRAP CSERSOFT
.WORD 14
.WORD DVEM8
.WORD ERR13

BR DVINIT
DVIN4: MOV #MCLR,@SELO ;DO A MASTER CLEAR
CMP #DMP6,OPTYP ;IS THIS A 8206
BNE DVIN6 ;IF NOT GO TO 6
MOVB #200,@BSEL1 ;SET RUN FOR 8206
NOP
NOP ;SLIGHT DELAY
DVIN6: TST @SELO ;IS RUN BIT SET
BMI DVIN1 ;IF YES GO TO 1 ELSE...
BREAK

```

7882 063064 104422
7883 063066 005737 017454
7884 063072 001371
7885
7886 063074 012737 037114 017354
7887 063102 017737 137742 017356
7888 063110 017737 137740 017360
7889 063116 004737 042102
7890 063122 005237 017310
7891 063126
7892 063126 104457
7893 063130 000004
7894 063132 037114
7895 063134 041456
7896
7897 063136 000670
7898
7899 063140 022777 000305 137716 DVIN1:
7900 063146 001422
7901 063150 012737 037175 017354
7902 063156 017737 137666 017356
7903 063164 017737 137674 017360
7904
7905 063172 004737 042102
7906 063176 005237 017310
7907 063202
7908 063202 104457
7909 063204 000005
7910 063206 037175
7911 063210 041456
7912 063212 000642
7913
7914
7915 063214 032737 000003 023076 DVIN13:
7916 063222 001427
7917 063224 022737 000001 017404
7918 063232 001023
7919 063234 012777 040400 137606
7920 063242
7921 063242 104422
7922 063244 000240
7923 063246 000240
7924 063250 132777 000200 137576
7925 063256 001771
7926 063260 042777 000400 137562
7927 063266 152777 000210 137556
7928 063274 012777 000006 137552
7929 063302 042737 000003 017414 DVIN12:
7930 063310 152777 000010 137534
7931 063316 112777 000221 137524
7932 063324 004737 065044
7933 063330 113777 023102 137526
7934 063336 005737 017406
7935 063342 001403
7936 063344 052777 000001 137512
7937 063352 112777 000002 137474 1$:
  
```

TST TIMER1 ;SEE IF TIME HAS EXPIRED
 BNE DVIN6 ;IF NOT GO BACK AND CHECK
 ;AGAIN ELSE...PRINT ERROR
 MOV #DVEMO,TEMP2
 MOV @SELO,TEMP3
 MOV @SEL2,TEMP4 ;LOAD UP ERRM. AND REG OUTPUTS
 JSR PC,LGDVE ;LOG TIME OUT WAITING FOR RUN
 INC ERRCNT
 ERRSOFT 4,DVEMO,ERR13
 TRAP CSBRK
 TRAP CSERSOFT
 .WORD 4
 .WORD DVEMO
 .WORD ERR13
 BR DVINIT ;GO BACK AND TRY MSTR CLR AGAIN IF ERROR
 CMP #305,@BSEL6 ;SEE IF MICRO DIAGS RAN OK
 BEQ DVIN13
 MOV #DVEM1,TEMP2 ;MOV ADDR OF ERR MSG TO TEMP2
 MOV @SELO,TEMP3
 MOV @SEL6,TEMP4 ;MOV SEL 6 TO TEMP4 IT CONTAINS
 ;THE ERROR CODE FROM MICRO DIAGS.
 JSR PC,LGDVE ;LOG DEVICE ERROR
 INC ERRCNT
 ERRSOFT 5,DVEM1,ERR13
 TRAP CSERSOFT
 .WORD 5
 .WORD DVEM1
 .WORD ERR13
 BR DVINIT ;GO BACK AND TRY TO RE INIT.
 ;GET HERE WHEN RUN IS SET AND MICRO DIAGS RAN OK.
 BIT #3,OPTYP ;IS THIS A DMV
 BEQ DVIN12 ;BRANCH IF NOT
 CMP #01,MLTYP ;IS IT INTERNAL LOOP
 BNE DVIN12 ;BRANCH IF NOT
 MOV #40400,@SELO ;SET MASTER CLEAR AND MREQ
 DIVN15: BREAK
 TRAP CSBRK
 NOP
 NOP
 BITB #BIT7,@BSEL2 ;IS RDI SET
 BEQ DIVN15 ;GO CHECK AGAIN
 BIC #BIT8,@SELO ;CLEAR MREQ
 BISB #210,@BSEL1 ;SET LULOOP AND RUN
 MOV #6,@SEL2 ;SET LU LOOP MAINT MODE
 BIC #3,FLAG ;CLEAR INPUT AND OUTPUT INT FLAGS
 BISB #BIT3,@BSEL1 ;SET LU LOOP BIT
 MOVB #221,@BSEL0 ;SET RQI IEO AND IEI
 JSR PC,TOORIO ;GO WAIT FOR INPUT INTERRUPT
 MOVB STATYP,@BSEL6 ;SET UP STATION TYPE
 TST FHDPLX ;IS THIS A HALF/DUPLEX
 BEQ 1\$;BRANCH IF SO.
 BIS #BIT0,@BSEL6 ;SET TO FULL DUPLEX
 MOVB #2,@BSEL2 ;DO MODE DEF

```

7938 063360 022737 000001 017404      CMP      #01,MLTYP      ;IS IT INTERNAL LOOP
7939 063366 001411                      BEQ      DVES1A        ;IF YES ESTABLISH TRIB
7940 063370 012737 000001 017460      MOV      #1,TIMERS     ;SET TIMER FOR 1 SEC
7941 063376 005737 017460      2$:     TST      TIMERS
7942 063402 001375                      BNE      2$           ;WAIT FOR TIMER
7943 063404 142777 000010 137440      BICB    #BIT3,@BSEL1  ;CLEAR LU LOOP
7944
7945
7946 063412                      DVES1A: ;WRITE GLOABL PARMAS
7947 063412 005037 015756      CLR      TRIBN        ;MAKE TRIBN 0
7948 063416 012737 017220 021020      MOV      #GLBPLS,TSSE ;TSSE POINTS TO LIST
7949 063424 004737 064554      JSR      PC,WRIPPG    ;WRITE POLL PARAMS
7950 063430 012737 000110 017350      MOV      #110,TEMP
7951 063436 022737 000003 017404      CMP      #MODLOC,MLTYP ;IS THIS MODEM LOCAL
7952 063444 001407                      BEQ      1$           ;BRANCH IF MODEM LOCAL
7953 063446 012737 000104 017350      MOV      #104,TEMP
7954 063454 022737 000004 017404      CMP      #MODREM,MLTYP ;IS THIS REM
7955 063462 001002                      BNE      2$           ;BRANCH IF NOT
7956 063464 004737 064662      1$:     JSR      PC,WRMCS    ;GO WRITE MODEM CONTROL
7957
7958 063470 012737 177777 015762      2$:     MOV      #-1,INDEX ;MAKE INDEX =-1
7959 063476 004737 046460      DVES1:  JSR      PC,GTVIND    ;GET VALID INDEX
7960 063502 022737 000040 015762      CMP      #32,,INDEX  ;DONE
7961 063510 001475                      BEQ      DVINEX       ;IF SO EXIT
7962
7963                      ;ESTABLISH TRIB
7964
7965 063512 152777 000200 137330      DVEST:  BISB    #RQI,@BSELO  ;DO REQUEST IN
7966 063520 004737 065044      JSR      PC,TOORIO   ;WAIT TIL PORT IS OURS
7967 063524 113777 015756 137324      MOVB    TRIBN,@BSEL3 ;SET UP TRIB NO.
7968 063532 012777 000001 137324      MOV      #01,@SEL6  ;ESTABLISH TRIB
7969 063540 112777 000001 137306      MOVB    #01,@BSEL2  ;CLEAR RDI AND DO COMMAND.
7970
7971                      ;WRITE POLL PARAMS IF NESC.
7972 063546 022737 000003 023100      CMP      #3,DEVPAR   ;IS THIS A MULTIPOINT CONTROL
7973 063554 001022                      BNE      POLLEN      ;BRANCH IF NOT.
7974 063556 004737 047152      JSR      PC,GETIND   ;GET VALID INDEX
7975 063562 013737 015762 017232      MOV      INDEX,MPLY  ;MOVE INDEX TO MULTIPLIER
7976 063570 012737 000020 017350      MOV      #16,,TEMP
7977 063576 012737 016220 017354      MOV      #POLLIS,TEMP2
7978 063604 004737 046434      JSR      PC,MTPLY    ;RETURN WITH ADDRESS
7979                      ;OF FIRST WORD IN TEMP2
7980
7981 063610 013737 017354 021020      MOV      TEMP2,TSSE
7982 063616 004737 064564      JSR      PC,WRIPP    ;WRITE POLL PARMAS
7983
7984                      ; ISTRT TRIB
7985
7986 063622 152777 000200 137220      POLLEN: BISB    #RQI,@BSELO  ;REQUEST IN
7987 063630 004737 065044      JSR      PC,TOORIO   ;WAIT TIL PORT IS OURS.
7988 063634 113777 015756 137214      MOVB    TRIBN,@BSEL3
7989 063642 012777 000004 137214      MOV      #04,@SEL6  ;MAKE IT MAINT MODE
7990 063650 022737 000004 017402      CMP      #DOW,MODTYP ;IS THIS DOWN LINE LOAD
7991 063656 001406                      BEQ      POLLE2
7992 063660 012777 000003 137176      MOV      #03,@SEL6  ;TO ISTRT
7993 063666 052737 000400 017414      BIS     #RUNST,FLAG  ;SET THE RUN STATE FLAG
  
```

7994	063674	112777	000001	137152	POLLE2: MOVB	#01,@BSEL2	:DO COMMAND
7995	063702	000675			BR	DVES1	:GO BACK
7996	063704	000207			DVINEX: RTS	PC	:RETURN TO CALLER
7997							
7998							
7999							
8000							
8001							

8002 .SBTTL DEVICE GET MODEM STATUS SUBROUTINE

8003
8004
8005
8006
8007
8008
8009
8010
8011
8012
8013
8014
8015
8016
8017
8018
8019
8020
8021
8022
8023
8024
8025
8026
8027
8028
8029
8030
8031
8032

++
: FUNCTIONAL DESCRIPTION:
: 'DVMODS' GET MODEM STATUS
:
: IMPLICIT INPUTS:
: THE BIT POSITION AND AVAILABILITY OF THE MODEM SIGNALS CTS,DSR,...RI..
: IN THE DEPENDENT PORTION OF THE GLOBAL EQUATES SECTION.
:
: OUTPUTS:
: CURRENT MODEM SIGNAL VALUES IN 'MODS'
:
: SUBORDINATE ROUTINES USED:
:
: CALLING SEQUENCE:
: JSR PC,DVMODS
:--

063706	152777	000200	137134	DVMODS: BISB	#RQI,@BSELO	:SET RQI
063714	004737	065044		JSR	PC,TOORIO	:GO TIME OUT CHECK
063720	012777	000020	137136	MCV	#20,@SEL6	:READ MODEM STATUS
063726	112777	000001	137120	MOVB	#01,@BSEL2	:DO CONTROL IN
063734	000207			RTS	PC	:RETURN TO CALLER

8033
8034
8035
8036
8037
8038
8039
8040
8041
8042
8043
8044
8045
8046
8047
8048
8049
8050
8051
8052
8053
8054
8055
8056
8057
8058
8059
8060
8061
8062
8063
8064
8065
8066
8067
8068
8069
8070
8071

.SBTTL DEVICE QUEUE RECEIVE SPACE SUBROUTINE

```

  :++
  : FUNCTIONAL DESCRIPTION:
  :   DVRXQ - THIS SUB ROUTINE QUES THE REC BUFFER SPACE TO THE
  :           DEVICE, THEN CLEARS THE QRX BIT OF THE FLAG WORD.
  :
  : INPUTS:
  :   DVRXA = ADDRESS OF RX BUFFER SPACE
  :   DVRCC = BYTE CHAR COUNT OF RX BUFFER
  :   QRX FLAG BIT = SET BY CALLING ROUTINE
  :
  : OUTPUTS:
  :   QRX FLAG BIT = CLEARED BY ROUTINE
  :
  : SUBORDINATE ROUTINES USED:
  :
  : CALLING SEQUENCE:
  :   JSR    PC,DVRXQ
  :--
  
```

```

DVRXQ:
  BIT    #QRX,FLAG
  BEQ    DVREX      ;IF NOT RX THEN EXIT
  BIC    #QRX,FLAG  ;ELSE QUE RX
  BIC    #QRX,FLAG  ;CLEAR FLAG FOR RX
  BISB   #RQI,@BSELO ;SET UP REQUEST
  JSR    PC,TOORIO  ;GO CHECK FOR IN OR OUT
  MOV    DVRXA,@SEL4
  MOV    DVRCC,@SEL6 ;LOAD CC AND ADDR
  MOVB   TRIBN,@BSEL3 ;SET UP TRIB NO.
  MOVB   #0,@BSEL2   ;DO COMMAND.
DVREX:  RTS    PC    ;RETURN TO CALLER
  
```

```

063736
063736 032737 000004 017414
063744 001424
063746 042737 000004 017414
063754 152777 000200 137066
063762 004737 065044
063766 013777 017270 137064
063774 013777 017272 137062
064002 113777 015756 137046
064010 112777 000000 137036
064016 000207
  
```

8072
8073
8074
8075
8076
8077
8078
8079
8080
8081
8082
8083
8084
8085
8086
8087
8088
8089
8090
8091
8092
8093
8094
8095
8096
8097
8098
8099
8100
8101
8102
8103
8104
8105
8106
8107
8108
8109
8110
8111
8112
8113
8114
8115
8116
8117
8118
8119
8120
8121
8122
8123
8124
8125
8126
8127

.SBTTL DEVICE TRANSMIT AND RECEIVE SUBROUTINE

..**
: FUNCTIONAL DESCRIPTION:
: DVTXRX-DEVICE TRANSMIT AND RECEIVE ROUTINE
: THIS CODE QUES THE TRANSMIT BUFFER TO THE DEVICE
: IF NEEDED. THE CODE THEN WAITS FOR A TX COMPLETE,
: RX COMPLETE OR BOTH. THE CODE REPORTS A TIME OUT
: ERROR IF NEITHER IS REPORTED BACK IN
: 60 SECONDS. AFTER REPORTING ERROR TIMER IS RE STARTED
: AND DEVICE WILL CONTINUE TO WAIT FOR INTERRUPT.

: INPUTS:
: 'DVTXA' = ADDRESS OF TRANSMIT MSG.
: 'DVTCC' = BYTE COUNT OF TRANSMIT MSG.
: 'QTX' BIT = SET IF TRANSMIT REQUESTED
: 'ETX' BIT = SET IF TRANSMIT EXPECTED
: 'ERX' BIT = SET IF RECEIVE EXPECTED

: OUTPUTS:
: 'DVTXA' = ADDRESS OF TX MSG. COMPLETED
: 'DVTCC' = BYTE COUNT OF TX MSG. COMPLETED
: 'QTX' = SET IF TX COMPLETED
: 'DVRXA' = ADDRESS OF RX MSG. COMPLETED
: 'DVRCC' = BYTE COUNT OF RX MSG. COMPLETED
: 'QRX' = SET IF RX COMPLETED

: SUBORDINATE ROUTINES USED:

: CALLING SEQUENCE:
: JSR PC,DVTXRX
:--

8108	064020	032737	000010	017414	DVTXRX: BIT	#QTX,FLAG	:ANY TX TO QUE
8109	064026	001424			BEC	DVTR3	:IF NOT GO WAIT FOR OUTPUT
8110	064030	042737	000010	017414	BIC	#QTX,FLAG	:CLEAR FLAG
8113	064036	152777	000200	137004	BISB	#RQI,@BSEL0	:SET REQUEST
8114	064044	004737	065044		JSR	PC,TOORIO	:GO CHECK FOR IN OR OUT
8115	064050	013777	017250	137002	MOV	DVTXA,@SEL4	
8116	064056	013777	017252	137000	MOV	DVTCC,@SEL6	
8117	064064	113777	015756	136764	MOVB	TRIBN,@BSEL3	:SET UP TRIB NO.
8118	064072	112777	000004	136754	MOVB	#4,@BSEL2	:DO COMMAND
8120	064100				DVTR3:		
8121	064100	012737	000074	017460	MOV	#60.,TIMERS	:SET TIMER FOR 60 SECS
8122	064106	005737	015766		TOINOT: TST	CRX	
8123	064112	001050			BNE	DVTR4	:BRANCH IF RX COMPLETED
8124	064114	005737	015764		TST	CTX	
8125	064120	001045			BNE	DVTR4	:BRANCH IF TX COMPLETED


```

8128 064122 005757 017460          TST     TIMERS          ;IS TIMER EXPIRED
8129 064126 001025                    BNE     TOIN1
8130 064130 012737 037255 017354    MOV     #DVEM2,TEMP2
8131 064136 017737 136706 017356    MOV     @SELO,TEMP3
8132 064144 017737 136704 017360    MOV     @SEL2,TEMP4
8133 064152 117737 136700 015756    MOVB   @BSEL3,TRIBN
8134 064160 004737 042102          JSR     PC,LGDVE
8135 064164 005237 017310          INC     ERRCNT
8136 064170                    ERRSOFT 6,DVEM2,ERR13
8137 064170 104457                    TRAP   C$ERSOFT
8138 064172 000006                    .WORD 6
8139 064174 037255                    .WORD DVEM2
8140 064176 041456                    .WORD ERR13
8141 064200 000737          BR     DVTR3          ;RETURN TO CHECK TIMER
8142
8143
8144 064202                    TOIN1:  BREAK
8145 064202 104422                    TRAP   C$BRK
8146 064204 032737 000002 017414    TOIN2:  BIT     #OTINT,FLAG
8147 064212 001735                    BEQ     TOINOT          ;IF NOT OUTPUT GO BACK AND
8148                    ;CHECK TIMER AGAIN
8149 064214 004737 065204          JSR     PC,OUTHDL      ;ELSE HANDLE OUTPUT AND RETURN
8150 064220 005737 015766          TST     CRX
8151 064224 001003          BNE     DVTR4          ;IF TX GO TO 4
8152 064226 005737 015764          TST     CTX
8153 064232 001725          BEQ     TOINOT          ;BRANCH IF NOT RX OR TX COMPLETED
8154 064234 005737 015764          DVTR4:  TST     CTX          ;IS IT TX COMPLETED
8155 064240 001456          BEQ     DVTR5          ;IF NOT TRY RX
8156 064242 032737 000200 017414    BIT     #ETX,FLAG      ;IF SO SHOULD IT BE
8157 064250 001023          BNE     DVTR4A         ;IF IT SHOULD GO TO 4A
8158 064252 012737 037512 017354    MOV     #DVEM5,TEMP2   ;ELSE LOG ERROR
8159 064260 013737 066516 017356    MOV     TSEL4,TEMP3
8160 064266 013737 066514 017360    MOV     TSEL6,TEMP4
8161 064274 013737 066520 015756    MOV     TSEL3,TRIBN
8162 064302 004737 042102          JSR     PC,LGDVE
8163 064306                    ERRSOFT 9,DVEM5,ERR13 ;REPORT ERROR
8164 064306 104457                    TRAP   C$ERSOFT
8165 064310 000011                    .WORD 9
8166 064312 037512                    .WORD DVEM5
8167 064314 041456                    .WORD ERR13
8168
8169 064316 000425          DVTR4A: BR     DVTR4B      ;THEN CLEAR COMPL.FLAG
8170 064320 013702 015774          MOV     TSPTR,R2
8171 064324 014237 017350          MOV     -(R2),TEMP      ;UNLOAD TRIBN
8172 064330 113737 017351 015756    MOVB   TEMP+1,TRIBN
8173 064336 105037 015757          CLRB   TRIBN+1
8174 064342 013737 015756 017254    MOV     TRIBN,DVTTB     ;UNLOAD TRIB NUMBER
8175 064350 014237 017250          MOV     -(R2),DVTXA     ;UNLOAD CC
8176 064354 014237 017252          MOV     -(R2),DVTCC     ;UNLOAD ADDRESS
8177 064360 010237 015774          MOV     R2,TSPTR
8178 064364 052737 000010 017414    BIS     #QTX,FLAG      ;AND SET TX COMPL FLAG
8179 064372 005337 015764          DVTR4B: DEC     CTX          ;AND COUNT DOWN FLAG
8180 064376 005737 015766          DVTR5:  TST     CRX
8181 064402 001463          BEQ     DVTREX          ;IF NOT THEN EXIT.
8182 064404 032737 000100 017414    BIT     #ERX,FLAG      ;TEST IS THIS SUPPOSED TO BE RX
8183 064412 001023          BNE     DVTR5A         ;IF YES PROCESS AS SUCH
    
```

```

8184 064414 012737 037567 017354      MOV      #DVEM6,TEMP2
8185 064422 013737 066522 017356      MOV      RSEL4,TEMP3
8186 064430 013737 066526 015756      MOV      RSEL3,TRIBN
8187 064436 013737 066524 017360      MOV      RSEL6,TEMP4      ;ELSE
8188 064444 004737 042102      JSR      PC,LGDVE         ;LOG ERROR
8189 064450      ERRSOFT 10,DVEM6,ERR13
8190 064450 104457
8191 064452 000012      TRAP    C$ERSOFT
8192 064454 037567      .WORD  10
8193 064456 041456      .WORD  DVEM6
8194
8195 064460 000432      BR      DVTRX1           ;AND EXIT
8196
8197 064462 013702 015770      DVTR5A: MOV     RSPTRS,R2
8198 064466 012237 017272      MOV     (R2)+,DVRCC
8199 064472 012237 017270      MOV     (R2)+,DVRXA     ;UNLOAD ADDR
8200 064476 012237 017350      MOV     (R2)+,TEMP
8201 064502 113737 017351 015756      MOV     TEMP+1,TRIBN
8202 064510 105037 015757      CLRB   TRIBN+1
8203 064514 013737 015756 017266      MOV     TRIBN,DVRTB     ;UNLOAD TRIBN
8204 064522 020227 016166      CMP     R2,#RXSKEN     ;IS IT AT THE END
8205 064526 001002      BNE    2$
8206 064530 012702 016012      MOV     #RXSTAK,R2     ;START OVER
8207 064534 010237 015770 2$:      MOV     R2,RSPTRS     ;RELOAD POINTER
8208 064540 052737 000004 017414      BIS     #QRX,FLAG
8209 064546 005337 015766      DVTRX1: DEC    CRX     ;COUNT DOWN CRX
8210
8211 064552 000207      DVTREX: RTS     PC     ;AND EXIT
8212

```

8213
8214
8215
8216
8217
8218
8219
8220
8221
8222
8223
8224
8225
8226
8227
8228
8229
8230
8231
8232
8233
8234
8235
8236
8237
8238
8239
8240
8241
8242
8243
8244

```

.SBTTL          DEVICE DEPENDENT SUBROUTINES
.SBTTL          WRITE POLL PAREMETERS
:++
: FUNCTIONAL DESCRIPTION:      WRIPP - WRITE POLL PARAMETERS
: WRITE ALL POLLING PARAMETRS FROM LIST
: POINTED TO BY TSSE FOR TRIB NUMBER IN TRIBN
:
: INPUTS:
:                               TRIBN - TRIB NUMBER OF WRITE
:                               TSSE - ADDRESS OF POLL LIST
:
: CALLING SEQUENCE:
:                               JSR      PC,WRIPP      ;FOR TRIBS
:                               JSR      PC,WRIPPG     ;FOR GLOBAL
:
:--
WRIPPG: MOV      #233,TSSA      ;LOAD TSSA WITH ADDR OF IS GLOBAL PP.
        BR       WRIP1        ;THEN GO TO 1
WRIPP:  MOV      #230,TSSA      ;LOAD TSSA WITH ADDR OF 1ST POLPAR.
WRIP1:  BISB    #RQI,@BSELO    ;DO REQUEST IN
        JSR     PC,TOORIO      ;WAIT TIL PORT IS OURS
        MOVB   TRIBN,@BSEL3    ;SET UP TRIBN
        MOV    @TSSE,@SEL4     ;MOVE DATA INTO SEL4
        MOVB   TSSA,@SEL6     ;SET UP POLL PARMATER
        MOVB   #01,@BSEL2     ;DO CONTROL IN WRITE TSS/GSS
        CMP    #237,TSSA
        BEQ    WRIP1EX        ;EXIT IF DONE
        INC    TSSA
        ADD    #2,TSSE
        BR     WRIP1          ;GO BACK FOR MORE
WRIP1EX: RTS      PC

```

```

064554 012737 000233 021022
064562 000403
064564 012737 000230 021022
064572 152777 000200 136250
064600 004737 065044
064604 113777 015756 136244
064612 017777 134202 136240
064620 113777 021022 136236
064626 112777 000001 136220
064634 022737 000237 021022
064642 001406
064644 005237 021022
064650 062737 000002 021020
064656 000745
064660 000207

```

```

8245 .SBTTL WRITE MODEM CONTROL
8246 :++
8247 : FUNCTIONAL DESCRIPTION: WRMCS - WRITE MODEM CONTROL SIGNALS
8248 :
8249 : WIRTE MODEM CONTROL SIGNALS FROM TEMP TO DMP
8250 : THIS ROUTINE IS IGNORED BY THE DMV
8251 :
8252 : INPUTS:
8253 : TEMP - CONTAINS CONTENTS FRO BSEL4
8254 :
8255 : CALLING SEQUENCE:
8256 : JSR PC,WRMCS ;WRITE MODEM CONTROL
8257 :--
8258
8259 064662 152777 000200 136160 WRMCS: BISB #RQI,@BSEL0 ;DO REQUEST IN
8260 064670 004737 065044 JSR PC,TOORIO ;WAIT TIL PORT IS OURS
8261 064674 113777 015756 136154 MOVB TRIBN,@BSEL3 ;SET UP TRIBN
8262 064702 013777 017350 136150 MOV TEMP,@SEL4
8263 064710 012777 000021 136146 MOV #21,@SEL6 ;DO WRITE MODEM
8264 064716 112777 000001 136130 MOVB #01,@BSEL2 ;CONTROL IN
8265 064724 000207 RTS PC ;THEN RETURN TO CALLER
8266
8267
8268
8269
8270
8271
8272
8273
8274
8275
8276
8277
8278
8279
8280
8281
8282
8283
8284 064726 022737 000001 023100 HLTTTB: CMP #1,DEVPAR ;IS THIS TRIB OR CONTROL
8285 064734 001442 BEQ HLTREX ;BRANCH IF TRIB
8286 064736 032737 000002 017410 BIT #DATCKB,PARAM
8287 064744 001006 BNE HLTTTR2 ;IF CHECK GO TO 2
8288 064746 012737 000002 017460 MOV #2,TIMERS ;SET UP FOR 2 SEC TIMER
8289 064754 005737 017460 HLTTTR3: TST TIMERS
8290 064760 001375 BNE HLTTTR3 ;WAIT FOR TIMER TO BE 0
8291 064762 HLTTTR2:
8292 064762 012737 177777 015762 MOV #-1,INDEX ;MAKE INDEX =-1
8293 064770 004737 046460 HLTTTR1: JSR PC,GTVIND ;GET VALID INDEX
8294 064774 022737 000040 015762 CMP #32,INDEX ;DONE
8295 065002 001417 BEQ HLTREX ;IF SO EXIT
8296
8297 ;HALT TRIB
8298
8299 065004 152777 000200 136036 BISB #RQI,@BSEL0 ;DO REQUEST IN
8300 065012 004737 065044 JSR PC,TOORIO ;WAIT TIL PORT IS OURS
  
```

```

.SBTTL HALT TRIB SUBROUTINE
:++
: FUNCTIONAL DESCRIPTION:
: HLTTRIB - HALT TRIB SUBROUTINE HALTS ALL TRIBS THAT
: ARE FOUND IN THE TRIBLSIT
:
: INPUTS: TRIBLS - CONTAINS VALID TRIBS
:
: SUBORDINATE ROUTINES USED:
:
: TOORIO - TIME OUT OR INPUT OR OUTPUTN INTERRUPT
:
: CALLING SEQUENCE:
: JSR PC,HLTTTB
:--
  
```

CZCLMBO DMP/V-11 DCLT MACY11 30A(1052) 28-JUL-81 13:35 PAGE 204
CZCLMB.P11 28-JUL-81 13:34 HALT TRIB SUBROUTINE

SEQ 0203

8301	065016	113777	015756	136032	MOVB	TRIBN,@BSEL3	:SET UP TRIB NO.
8302	065024	012777	000005	136032	MOV	#05,@SEL6	:HALT TRIB
8303	065032	112777	000001	136014	MOVB	#01,@BSEL2	:CLEAR RDI AND DO COMMAND.
8304	065040	000753			BR	HLTIR1	:GO BACK AND GET ANOTHER
8305	065042	000207			HLTREX: RTS	PC	:RETURN TO CALLER
8306							

8307 .SBTTL TIME OUT OR INPUT INT. OR OUTPUT INT.
 8308
 8309
 8310

```

:++
: FUNCTIONAL DESCRIPTION:
: TOORIO - TIME OUT OR INPUT INTERRUPT OR OUTPUT INTERRUPT
: THIS ROUTINE SETS UP A TIMER FOR 100 (OCTAL) TICKS
: THEN CHECKS FOR TIME OUT, OR INPUT INTERRUPT, OR OUTPUT
: INTERRUPT. IF TIME OUT OCCURS IT REPORTS ERROR AND
: RESTARTS TIMER. IF INPUT INTERRUPT OCCURS RETURN TO CALLER
: IF OUTPUT INTERRUPT OCCURS LOG IT AND CONTINUE WAITING FOR
: INPUT INTERRUPT.
  
```

```

: USE OF FLAGS:
: 'OTINT' - SET BY OUTPUT INT ROUTINE
: 'ININT' - SET BY INPUT INT. ROUTINE
: CLEARED BY THIS ROUTINE.
  
```

```

: SUBORDINATE ROUTINES USED:
: 'OUTHDL' - OUTPUT INTERRUPT HANDLER
  
```

```

: CALLING SEQUENCE:
: JSR PC,TOORIO
:--
  
```

```

8333 065044 011637 017374 TOORIO: MOV (SP),PCADD ;SAVE ADDR. OF CALLING ROUTINE
8334 065050 012737 000100 017454 MOV #100,TIMER1 ;SET UP TIMER
8335 065056 032737 000003 023076 BIT #3,OPTYP ;IS THIS DMV
8336 065064 001403 BEQ TOOR3 ;BRANCH IF NOT
8337 065066 012737 000400 017454 MOV #400,TIMER1 ;MAKE TIME OUT GREATER IF DMV
8338 065074 005737 017454 TOOR3: TST TIMER1 ;IS TIME EXPIRED
8339 065100 001022 BNE TOOR1 ;IF NOT CONTINUE
8340 ;IF YES ERROR
8341 065102 012737 037352 017354 MOV #DVEM3,TEMP2
8342 065110 017737 135740 017360 MOV @SEL2,TEMP4
8343 065116 017737 135726 017356 MOV @SEL0,TEMP3
8344 065124 004737 042102 JSR PC,LGDVE
8345 065130 005237 017310 INC ERRCNT
8346 065134 ERRSOFT 7,DVEM3,ERR13
8347 065134 104457 TRAP C$ERSOFT
8348 065136 000007 .WORD 7
8349 065140 037352 .WORD DVEM3
8350 065142 041456 .WORD ERR13
8351 065144 000737 BR TOORIO
8352
8353 065146 TOOR1: BREAK
8354 065146 104422 TRAP C$BRK
8355 065150 032737 000002 017414 BIT #OTINT,FLAG ;IS THERE AN OUTPUT
8356 ;PENDING
8357 065156 001402 BEQ TOOR2 ;IF NOT GO TO 2
8358 ;ELSE GO HANDL IT
8359 065160 004737 065204 JSR PC,OUTHDL
8360 065164 032737 000001 017414 TOOR2: BIT #ININT,FLAG ;IS THERE AN INPUT PENDING
8361 065172 001740 BEQ TOOR3 ;IF NOT GO BACK TO TIMER CK.
8362 065174 042737 000001 017414 BIC #ININT,FLAG ;ELSE CLEAR THE INPUT PEND FLAG
  
```

CZCLMBO DMP/V-11 DCLT MACY11 30A(1052) 28-JUL-81 13:35 PAGE 206
CZCLMB.P11 28-JUL-81 13:34 TIME OUT OR INPUT INT. OR OUTPUT INT.

K 16

SEQ 0205

8363 065202 000207
8364

RTS PC

;AND RETURN TO CALLER

```

8365 .SBTTL OUTPUT INTERRUPT HANDLER
8366
8367
8368 : **
8369 : FUNCTIONAL DESCRIPTION:
8370 : OUTHDL - OUTPUT INTERRUPT HANDLER
8371 : THIS ROUTINE IS CALLED WHEN AN OUTPUT INTERRUPT HAS SET
8372 : THE 'OTINT' BIT IN THE 'FLAG' WORD. IT CHECKS FOR
8373 : AN RDO SIGNAL IF NO RDO THEN REPORT ILLEGAL INTERRUPT.
8374 : THEN IT CHECKS FOR BACC OUT IF NOT BACC OUT REPORT THE
8375 : TYPE OF OUTPUT ERROR. IF BACC OUT FIND IF RX OR TX
8376 : IF RX SET CRX BIT AND MOVE ADDR AND BYTE COUNT TO RSEL4
8377 : AND RSEL6. IF TX SET CTXV BIT AND MOVE ADDR AND BYTE COUNT
8378 : TO TSEL4 AND TSEL6. CLEAR OTINT FLAG AND RETURN TO CALLER.
8379
8380 : USE OF FLAGS:
8381 : 'OTINT' - SET BY OUPUT ROUTINE
8382 : CLEARED BY THIS ROUTINE.
8383 : 'DMRRUN' - SET BY DVINIT ROUTINE IF THIS IS DMR
8384 : CHECKED AND CLEARED BY THIS ROUTINE.
8385 : 'CTX' - SET IF TRANSMIT COMPLETED
8386 : 'CRX' - SET IF RECEIVE COMPLETED
8387
8388 : SUBORDINATE ROUTINES USED:
8389 :
8390 : 'LGDVE' -LOG DEVICE ERRORS TO EVENT LOG
8391
8392 : CALLING SEQUENCE
8393 : JSR PC,OUTHDL
8394
8395 : --
8396
8397 065204 011637 017374 OUTHDL: MOV (SP),PCADD ;SAVE ADDR. OF CALLING ROUTINE
8398 065210 042737 000002 017414 BIC #OTINT,FLAG
8399 065216 005737 017320 TST CLNSET
8400 065222 001404 BEQ OUTH1
8401 065224 142777 000200 135622 BICB #RDO,@BSEL2 ;CLEAR RDO
8402 065232 000207 RTS PC ;RETURN TO CALLER
8403 065234 OUTH1:
8404 065234 017703 135614 MOV @BSEL2,R3
8405 065240 042703 177770 BIC #^C<7>,R3 ;STRIP TO COMMAND CODE
8406 065244 022703 000001 CMP #1,R3 ;IS IT CONTROL OUT
8407 065250 001405 BEQ CONOHD ;IF SO GO TO CONTROL OUT HANDLER
8408 065252 022703 000002 CMP #2,R3 ;IS IN INFO OUT
8409 065256 001531 BEQ INFOHD ;IF SO GO TO INFORMATION OUT HANDLER
8410 065260 000137 066166 JMP BACC HD ;IF NOT JUMP TO BA CC HANDLER
8411
8412 :CONTROL OUT HANDLER
8413
8414 065264 CONOHD:
8415 065264 005003 CLR R3
8416 065266 157703 135572 BISB @BSEL6,R3 ;
8417
8418 065272 032703 000100 BIT #BIT6,R3 ;IS THIS ERROR IN THE 100-176 RANGE
8419 ;OR THE 300-376 RANGE
8420 065276 001047 BNE CONO1 ;BRANCH IF YES.
  
```



```

8421 065300 022703 000024      CMP      #24,R3      ;IS THIS A RUN STATE
8422 065304 001006              BNE      CON01B     ;IF NOT GO TO 1B
8423 065306 032737 000400 017414  BIT      #RUNST,FLAG ;TEST THE RUN STATE
8424 065314 001434              BEQ      CON01A     ;IF NOT SET GO TO 1A
8425 065316 000137 066344      JMP      OUTHEX
8426 065322 022703 000002      CON01B: CMP      #2,R3      ;IS IT RX THRESH
8427 065326 001004              BNE      CON01C     ;BRANCH IF NOT
8428 065330 005237 017302      INC      OPVAR      ;BUMP OPVAR
8429 065334 000137 066344      JMP      OUTHEX     ;AND EXIT
8430 065340 022703 000004      CON01C: CMP      #4,R3      ;IS IT TX THRESH
8431 065344 001004              BNE      CON01D     ;BRANCH IF NOT
8432 065346 005237 017304      INC      OPVAR1     ;IN TX COUNT
8433 065352 000137 066344      JMP      OUTHEX     ;AND EXIT ROUTINE
8434 065356 022703 000006      CON01D: CMP      #6,R3      ;IS IT SELECT
8435 065362 001411              BEQ      CON01A     ;BRANCH IF SO
8436 065364 022703 000032      CMP      #32,R3     ;IS IT RING D
8437 065370 001406              BEQ      CON01A
8438 065372 022703 000022      CMP      #22,R3     ;IS IT DEAD TRIB
8439 065376 001403              BEQ      CON01A     ;BRANCH IF SO
8440 065400 012737 177777 017324  MOV      #-1,FTLFLG ;SET FATAL ERROR FLAG
8441 065406 016337 023104 017366  CON01A: MOV      CON0LS(R3),CONOTM
8442 065414 000427              BR       CON04      ;THEN GO TO 4
8443
8444 065416 012737 177777 017324  CON01:  MOV      #-1,FTLFLG ;SET FATAL ERROR FLAG
8445 065424 032703 000200              BIT      #BIT7,R3   ;IS THIS 300 RANGE
8446 065430 001006              BNE      CON03      ;IF SO GO TO 3
8447 065432 042703 000100              BIC      #BIT6,R3   ;CLEAR TOP BIT
8448 065436 016337 023140 017366  MOV      CON01S(R3),CONOTM
8449 065444 000413              BR       CON04      ;LOAD UP MSG AND GO TO 4
8450
8451 065446 022703 000306              CON03:  CMP      #306,R3 ;IS THIS QUE OVER FLOW
8452 065452 001003              BNE      CON03A
8453 065454 012737 177777 017330  MOV      #-1,OVRCNT
8454 065462 042703 000300              CON03A: BIC      #BIT7+#BIT6,R3 ;CLEAR THE TOP BITS
8455 065466 016337 023206 017366  MOV      CON03S(R3),CONOTM
8456
8457 065474 017737 135364 017360  CON04:  MOV      @SEL6,TEMP4
8458 065502 017737 135346 017356  MOV      @SEL2,TEMP3
8459 065510 012737 037426 017354  MOV      #DVEM4,TEMP2
8460 065516 004737 042102              JSR      PC,LGDVE   ;GO LOG ERROR
8461 065522 005237 017310              INC      ERRCNT
8462 065526              ERRSOFT 7,DVEM4,ERR14
8463 065526 104457
8464 065530 000007
8465 065532 037426
8466 065534 041510
8467 065536 000137 066344      JMP      OUTHEX     ;EXIT OUTPUT HANDLER
8468
8469              ;INFORMATION OUT HANDLER
8470              ;BA AND CC HANDLER
8471
8472
8473
8474 065542              INFOHD:
8475 065542 122777 000010 135314  CMPB     #10,@BSEL6 ;IS THIS A MODEM STATUS
8476 065550 001005              BNE     INFOH1     ;GO TO INFO 1 IF NOT MODEM STATUS
  
```

```

TRAP  CSERSOFT
.WORD 7
.WORD DVEM4
.WORD ERR14
  
```

```

8477 065552 017737 135302 020522      MOV      @SEL4,MODS      ;PUT IN NEW MOD STATUS
8478 065560 000137 066344      INFO1B: JMP      OUTHEX
8479 065564 032777 000040 135272      INFOH1: BIT      #BIT5,@BSEL6      ;
8480 065572 001011      BNE      INFOHA      ;BRANCH IF RD/TSS
8481 065574 022777 000020 135262      CMP      #20,@BSEL6
8482 065602 001766      BEQ      INFO1B      ;GET OUT IF BUFF RET CMP
8483 065604 012737 041210 017366      MOV      #INFOM,CONOTM      ;SET UP FOR INFO ERROR
8484 065612 000137 065474      JMP      CONO4      ;AND PRINT IT
8485 065616 005037 017326      INFOHA: CLR      TSSFLG      ;CLEAR FLAG
8486 065622 017704 135232      MOV      @SEL4,R4
8487 065626 017703 135232      MOV      @BSEL6,R3
8488 065632 042703 177740      BIC      #177740,R3      ;CLEAR ALL BUT LAST 5 BITS
8489 065636 105777 135214      TSTB    @BSEL3      ;IS THIS GSS
8490 065642 001007      BNE      INFOH8      ;BRANCH IF NOT
8491
8492
8493 065644 116302 021126      MOVB    GSSIND(R3),R2
8494 065650 006303      ASL     R3      ;USE WORD INDEX
8495 065652 016337 021026 017366      MOV     GSSLST(R3),CONOTM      ;USE GSS LIST
8496 065660 000406      BR     INFOH2
8497
8498 065662 116302 020760      INFOH8: MOVB    TSSIND(R3),R2
8499 065666 006303      ASL     R3
8500 065670 016337 020660 017366      MOV     TSSLST(R3),CONOTM      ;IF TSS USE THAT LIST
8501
8502 065676 000172 065702      INFOH2: JMP     @INFOH4(R2)
8503
8504 065702 065710      INFOH4: .WORD   INFOH5      ;WORD ROUTINE
8505 065704 065736      .WORD   INFOH6      ;BYTE ROUTINE
8506 065706 066002      .WORD   INFOH7      ;SPECIAL ROUTINE
8507 065710      INFOH5: PRINTS  CONOTM,R4
8508 065710 010446      MOV     R4,-(SP)
8509 065712 013746 017366      MOV     CONOTM,-(SP)
8510 065716 012746 000002      MOV     #2,-(SP)
8511 065722 010600      MOV     SP,R0
8512 065724 104416      TRAP   C$PNTS
8513 065726 062706 000006      ADD     #6,SP
8514 065732 000137 066344      INFOH6: JMP     OUTHEX
8515 065736 010437 017350      MOV     R4,TEMP
8516 065742      PRINTS  CONOTM,<B,TEMP>,<B,TEMP+1>
8517 065742 005046      CLR     -(SP)
8518 065744 153716 017351      BISB   TEMP+1,(SP)
8519 065750 005046      CLR     -(SP)
8520 065752 153716 017350      BISB   TEMP,(SP)
8521 065756 013746 017366      MOV     CONOTM,-(SP)
8522 065762 012746 000003      MOV     #3,-(SP)
8523 065766 010600      MOV     SP,R0
8524 065770 104416      TRAP   C$PNTS
8525 065772 062706 000010      ADD     #10,SP
8526 065776 000137 066344      INFOH7: JMP     OUTHEX
8527 066002 010437 017350      MOV     R4,TEMP
8528 066006 005037 017352      CLR     TEMP1
8529 066012 005037 017354      CLR     TEMP2
8530 066016 005037 017356      CLR     TEMP3
8531 066022 005037 017360      CLR     TEMP4
8532 066026 032737 000400 017350      BIT     #BIT8,TEMP
  
```

```

8533 066034 001402          BEQ      1$
8534 066036 005237 017352    INC      TEMP1
8535 066042 032737 001000 017350 1$:    BIT      #BIT9,TEMP
8536 066050 001402          BEQ      2$
8537 066052 005237 017354    INC      TEMP2
8538 066056 032737 002000 017350 2$:    BIT      #BIT10,TEMP
8539 066064 001402          BEQ      4$
8540 066066 005237 017356    INC      TEMP3
8541 066072 032737 004000 017350 4$:    BIT      #BIT11,TEMP
8542 066100 001402          BEQ      3$
8543 066102 005237 017360    INC      TEMP4
8544 066106          3$:    PRINTS  CONOTM,<B,TEMP>,<B,TEMP1>,<B,TEMP2>,<B,TEMP3>,<B,TEMP4>
8545 066106 005046          CLR      -(SP)
8546 066110 153716 017360    BISB    TEMP4,(SP)
8547 066114 005046          CLR      -(SP)
8548 066116 153716 017356    BISB    TEMP3,(SP)
8549 066122 005046          CLR      -(SP)
8550 066124 153716 017354    BISB    TEMP2,(SP)
8551 066130 005046          CLR      -(SP)
8552 066132 153716 017352    BISB    TEMP1,(SP)
8553 066136 005046          CLR      -(SP)
8554 066140 153716 017350    BISB    TEMP,(SP)
8555 066144 013746 017366    MOV     CONOTM,-(SP)
8556 066150 012746 000006    MOV     #6,-(SP)
8557 066154 010600          MOV     SP,R0
8558 066156 104416          TRAP   C$PNTS
8559 066160 062706 000016    ADD     #16,SP
8560 066164 000467          BR      OUTHEX
8561 066166          BACCHD:
8562 066166 032703 000004    BIT      #BIT2,R3
8563 066172 001035          BNE     BACCTX
8564 066174 022737 000022 015766    CMP     #18.,CRX
8565 066202 001001          BNE     1$
8566 066204          DOCLN
8567 066204 104444          TRAP   C$DCLN
8568 066206 005237 015766    1$:    INC     CRX
8569
8570 066212 013702 015772    MOV     RSPTRE,R2
8571 066216 017722 134642    MOV     @SEL6,(R2)+
8572 066222 017722 134632    MOV     @SEL4,(R2)+
8573 066226 017722 134622    MOV     @SEL2,(R2)+
8574 066232 022702 016166    CMP     #RXSKEN,R2
8575 066236 001002          BNE     2$
8576 066240 012702 016012    MOV     #RXSTAK,R2
8577 066244 010237 015772    2$:    MOV     R2,RSPTRE
8578 066250 005737 017330    3$:    TST     OVRCNT
8579 066254 001433          BEQ     OUTHEX
8580 066256          ERRSOFT 12,DVEM7,ERR15
8581 066256 104457          TRAP   C$ERSOFT
8582 066260 000014          .WORD  12
8583 066262 037643          .WORD  DVEM7
8584 066264 041546          .WORD  ERR15
8585
8586
8587 066266 005237 015764    -BACCTX: INC     CTX
8588 066272 013702 015774    MOV     TSPTR,R2
;INC TX COMPLETE COUNT
;LOAD R2 WITH POINTER

```

```

8589 066276 017722 134562      MOV    @SEL6,(R2)+
8590 066302 017722 134552      MOV    @SEL4,(R2)+
8591 066306 017722 134542      MOV    @SEL2,(R2)+
8592 066312 010237 015774      MOV    R2,TSPTR
8593 066316 022702 016012      CMP    #RXSTAK,R2
8594 066322 001001                BNE    1$
8595 066324                DOCLN                ;BAD NEWS
8596 066324 104444                TRAP   C$DCLN
8597
8598 066326 005737 017330      1$:   TST    OVRCNT      ;CHECK IF HERE FROM QUE OVER
8599 066332 001404                BEQ    OUTHEX
8600 066334                ERRSOFT 13,DVEM7,ERR16
8601 066334 104457                TRAP   C$ERSOFT
8602 066336 000015                .WORD 13
8603 066340 037643                .WORD DVEM7
8604 066342 041606                .WORD ERR16
8605
8606 066344 142777 000200 134502  OUTHEX: BICB   #RDO,@BSEL2  ;CLEAR RDO
8607 066352 005737 017330                TST    OVRCNT      ;TEST THE OVER FLOW COUNT
8608 066356 001427                BEQ    OUTHE3      ;BRANCH IF ZERO
8609 066360                OUTHE4: BREAK
8610 066360 104422                TRAP   C$BRK
8611 066362 032737 000002 017414      BIT    #OTINT,FLAG  ;IS OUTPUT INTERRUPT SET
8612 066370 001402                BEQ    OUTHE5      ;BRANCH IF NOT
8613 066372 000137 065204      JMP    OUTHDL      ;WHEN SET GO BACK FOR NEXT ON QUE
8614 066376 032737 000001 017414  OUTHE5: BIT    #ININT,FLAG  ;TEST FOR INPUT INT
8615 066404 001414                BEQ    OUTHE3      ;BRANCH IF NOT INPUT
8616 066406 002737 000001 017414      BIC    #ININT,FLAG
8617 066414 105077 134444      CLRB   @BSEL6
8618 066420 112777 000001 134426      MOVB  #01,@BSEL2  ;DO NO REQUEST
8619 066426 012737 177777 017322      MOV    #-1,RQIFLG ;SET RQI FLAG
8620 066434 000751                BR     OUTHE4
8621 066436 005737 017324      OUTHE3: TST    FTLFLG
8622 066442 001406                BEQ    OUTHE6      ;BRANCH IF NOT FATAL
8623 066444 005037 017324      CLR    FTLFLG      ;CLEAR FATAL FLAG
8624 066450 013706 017364      MOV    SAVSP,SP   ;RESET STACK
8625 066454 000137 052354      JMP    GTRAS      ;RETRACT NOW
8626 066460 005737 017322      OUTHE6: TST    RQIFLG
8627 066464 001405                BEQ    OUTHE2
8628 066466 005037 017322      CLR    RQIFLG      ;CLEAR THE RQI FLAG.
8629 066472 152777 000200 134350  OUTHE2: BISB   #RQI,@BSEL0
8630 066500 005737 017326      TST    TSSFLG      ;TEST THE TSSFLG
8631 066504 001325                BNE    OUTHE4      ;IF NOT ZERO WAIT TIL IT IS.
8632 066506 005037 017330      CLR    OVRCNT      ;CLEAR THE OVERFLOW FLAG
8633 066512 000207                RTS    PC          ;RETURN TO CALLER
8634
8635 066514 000000      TSEL6: .WORD 0
8636 066516 000000      TSEL4: .WORD 0
8637 066520 000000      TSEL3: .WORD 0
8638 066522 000000      RSEL4: .WORD 0
8639 066524 000000      RSEL6: .WORD 0
8640 066526 000000      RSEL3: .WORD 0
8641

```

;TEMP STORAGE LOCS.

8642
8643
8644
8645
8646
8647
8648
8649
8650
8651
8652
8653
8654
8655
8656
8657
8658
8659
8660
8661
8662
8663
8664

.SBTTL

DEVICE INTERRUPT SERVICE ROUTINES

BGNSRV DVINS

DVINS::

BIS #ININT,FLAG
BIC #BIT7,@BSELO ;CLEAR ROI

ENDSRV

L10021:
RTI

BGNSRV DVOUTS

DVOUTS::

BIS #OTINT,FLAG
ENDSRV

L10022:
RTI

8665
8666
8667
8668 066556
8669 066556
8670 066556 104401
8671

.EVEN
ENDTST

L10020: TRAP CSETST

8672
8673

8674
8675
8676
8677
8678
8679
8680
8681
8682
8683
8684
8685
8686
8687
8688
8689
8690
8691
8692
8693
8694
8695
8696
8697
8698
8699
8700
8701
8702
8703
8704
8705
8706
8707
8708
8709
8710
8711
8712
8713
8714
8715
8716
8717
8718
8719
8720
8721
8722
8723
8724
8725
8726
8727
8728
8729

066560
066560 000034
066562

066562
066562 000130
066564 066652
066566 000001

066570
066570 001031
066572 066703
066574 160000
066576 177776

066600
066600 002031
066602 066730
066604 000300
066606 000776

066610
066610 003032
066612 066763
066614 000340
066616 000004
066620 000007

066622
066622 005032
066624 067100
066626 000007
066630 000000
066632 000004

066634
066634 004130
066636 067011
066640 000001
066642

.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.
:--
  
```

BGNHRD

.WORD L10023-L\$HARD/2
L\$HARD::

.SBTTL DEVICE INDEPENDENT SECTION

GPRML DPLX,0,1,YES

.WORD T\$CODE
.WORD DPLX
.WORD 1

.SBTTL DEVICE DEPENDENT SECTION

GPRMA CSRADR,2,0,160000,177776,YES

.WORD T\$CODE
.WORD CSRADR
.WORD T\$LLOLIM
.WORD T\$HILIM

GPRMA VECTOR,4,0,300,776,YES

.WORD T\$CODE
.WORD VECTOR
.WORD T\$LLOLIM
.WORD T\$HILIM

GPRMD PRIOR,6,0,340,4,7,YES

.WORD T\$CODE
.WORD PRIOR
.WORD 340
.WORD T\$LLOLIM
.WORD T\$HILIM

GPRMD OPTYPM,12,0,7,0,4,YES

.WORD T\$CODE
.WORD OPTYPM
.WORD 7
.WORD T\$LLOLIM
.WORD T\$HILIM

GPRML PTPMLP,10,1,YES

.WORD T\$CODE
.WORD PTPMLP
.WORD 1

XFERF ENDHWL

8730 066642 004044 .WORD T\$CODE
8731 066644 GPRML TRIBCO,10,2,YES .WORD T\$CODE
8732 066644 004130 .WORD TRIBCO
8733 066646 067045 .WORD 2
8734 066650 000002 .WORD
8735 066652 ENDPHWL: .EVEN
8736 066652 ENDHRD L10023:
8737
8738 066652
8739
8740

.NLIST BEX

;DEVICE INDEPENDENT QUESTIONS

066652 052506 046114 042040 DPLX: .ASCIZ /FULL DUPLEX OPERATION : /

;DEVICE DEPENDENT QUESTION

066703 104 053105 041511 CSRADR: .ASCIZ /DEVICE CSR ADDRESS: /
066730 047111 042524 051122 VECTOR: .ASCIZ /INTERRUPT VECTOR ADDRESS: /
066763 111 052116 051105 PRIOR: .ASCIZ /INTERRUPT PRIORITY : /
067011 111 020123 044124 PTPMLP: .ASCIZ /IS THIS MULTIPOINT NETWORK: /
067045 111 020123 044124 TRIBCO: .ASCIZ /IS THIS A CONTROL STATION: /
067100 050117 044524 047117 OPTYPM: .ASCII /OPTION TYPE /<15><12>
067116 030040 042075 050115 .ASCII / 0=DMP/<15><12>/ 1=DMV: /

.LIST BEX
.EVEN

8741 067136
8742
8743

8744
8745
8746
8747
8748
8749
8750
8751
8752
8753
8754
8755
8756
8757
8758
8759
8760
8761
8762
8763
8764
8765
8766
8767
8768
8769
8770
8771
8772
8773
8774
8775
8776
8777
8778

;.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.
:--

: BGNSFT

: ENDSFT

.....
: TEMPORARY PATCH AREA - FOR DEBUG PURPOSES
:

\$PATCH:
 .BLKW 30

 LASTAD

L\$LAST::
 ENDMOD

.END

.EVEN
.WORD 0
.WORD 0

067136
067136 000030

067216

067216 000000
067220 000000
067222
067222

000001

ACT =	000003	2628#	6181	6636	6716					
ACTATV	055000	6402	6636#							
ACTBCR	054604	6420	6594#							
ACTCHK	055214	6382	6682#							
ACTCKT	056762	6431	6997#							
ACTCLB	054126	6483	6497#							
ACTCLP	055326	6416	6710#							
ACTCLR	053616	6380	6440#							
ACTCOP	054424	6390	6557#							
ACTCRC	055230	6411	6688#							
ACTCSE	053752	6385	6469#							
ACTCST	054044	6386	6485#							
ACTDLL	055046	6406	6650#							
ACTDME	054352	6422	6538	6541#						
ACTDMQ	054344	6423	6540#							
ACTDMS	054322	6421	6535#							
ACTDMX	054360	6542#								
ACTECH	055124	6410	6666#							
ACTEKE	056574	6924	6953#							
ACTEKT	056356	6430	6904#							
ACTEQO	054546	6394	6583#							
ACTETB	055424	6427	6729#							
ACTEWS	055526	6432	6751#							
ACTEXT	053702	6433	6458#							
ACTEXX	056760	6909	6922	6938	6951	6967	6985	6994#		
ACTESA	056562	6948#	6950							
ACTESB	056556	6947#								
ACTHLP	053636	6384	6446#							
ACTKAL	055444	6187	6429	6735#						
ACTKTB	055434	6428	6732#							
ACTLIS	055036	6405	6647#							
ACTLLP	055336	6417	6712#							
ACTLPX	055354	6707	6709	6711	6713	6716#				
ACTLXX	055416	6680	6701	6704	6717	6726#				
ACTMEX	054772	6576	6592	6614	6619	6626	6629	6633#		
ACTME1	054726	6603	6605	6607	6609	6611	6618#			
ACTMOP	055306	6414	6706#							
ACTMOS	055236	6425	6691#							
ACTMSO	054626	6395	6602#							
ACTMS1	054634	6396	6604#							
ACTMS2	054644	6397	6606#							
ACTMS3	054654	6398	6608#							
ACTMS4	054664	6399	6610#							
ACTMS5	054674	6400	6612#							
ACTMS6	054712	6401	6615#							
ACTM2X	055074	6637	6645	6648	6651	6654	6658#			
ACTNO	055114	6409	6663#							
ACTNUF	053606	6419	6437#							
ACTNUL	053614	6379	6438#							
ACTNUM	054434	6391	6560#							
ACTOPM	054526	6392	6578#							
ACTPAS	055010	6403	6639#							
ACTPRO	055244	6412	6694#							
ACTPRT	053712	6424	6460#							
ACTQFG	055250	6683	6686	6689	6692	6696#				
ACTREC	055030	6404	6644#							

ACTREX	044476	4496	4520#		
ACTRGS	044522	4498	4525#		
ACTRHL	044432	4495	4508#		
ACTRLG	044506	4497	4522#		
ACTRLP	055346	6418	6714#		
ACTRNF	044422	4504	4506#		
ACTRNL	044430	4494	4507#		
ACTRPS	055276	6413	6703#		
ACTRSE	044544	4501	4529#		
ACTRSF	044670	4502	4553#		
ACTRSO	044704	4503	4556#		
ACTRTN	044576	4500	4535#		
ACTRTS	044552	4499	4531#		
ACTRUN	053726	6383	6464#		
ACTSEX	054736	6434	6623#		
ACTSHO	053626	6381	6443#		
ACTSHW	054166	6475	6489	6508#	6527
ACTSLS	056122	6426	6849#		
ACTSTE	054366	6387	6547#		
ACTSTS	055222	6393	6685#		
ACTSTT	054376	6388	6550#		
ACTSTX	054404	6548	6551#		
ACTSZE	054414	6389	6554#		
ACTTAL	055066	6408	6656#		
ACTTLP	055316	6415	6708#		
ACTTRA	055056	6407	6653#		
ACTWS1	055704	6782	6783#		
ACTWS2	055710	6783	6785#		
ACTWS3	056010	6784	6815#		
ACTWS5	055640	6771	6773#	6807	
ACTWS7	055754	6802#	6847		
ACTWS9	055542	6756#			
ACTW7A	055772	6806#			
ACTW7B	056000	6755	6804	6808#	
ADDCC	045346	4687#	6310	6353	
ADDCC1	045442	4689	4704#		
ADR =	000020	2598#			G
ALCK	060122	3192	7246#		
ALCK1	060206	7301	7305#	7356	
ALCK2	060262	7306	7317#	7400	
ALCK2A	060522	7352	7358#		
ALCK3	060562	7320	7362	7366#	
ALCK3A	060704	7376	7386#		
ALCK3B	060726	7388	7391#		
ALCK3C	060716	7384	7389#	7397	
ALCK3D	060666	7379	7382#		
ALCK4	060750	7367	7395#		
ALCK4A	060762	7394	7398#		
ALCK5	060162	7300#	7390		
ALCK5B	060172	7302#			
ALLTR	060162	7178	7201	7225	7299#
ASSEMB=	000010	2367			
ATVMOD=	000027	2733#	3484		
BABTM	040273	3665	3764#		
BACCHD	066166	8410	8561#		
BACCTX	066266	8563	8587#		

LOGEOP	042246	4095#	7530					
LOGEX	042550	4111	4159#					
LOGMSC	042264	4100#						
LOGRXC	042072	4064#	7324	7666	7809			
LOGRXQ	042054	4059#	5464	7631	7803			
LOGS1	042302	4052	4057	4062	4066	4105#		
LOGS2	042542	4153	4157#					
LOGS3	042342	4070	4079	4093	4098	4103	4115#	
LOGS3A	042236	4084	4088	4092#				
LOGS4	042416	4123	4132#					
LOGS5	042442	4117	4119	4140#				
LOGTXC	042036	4054#	7371	7654	7758			
LOGTXQ	042020	4049#	7315	7637	7750			
LOGUNT	017372	3166#	5938*	5940*	5941	5945		
LOOPS	003362	3016#	5497					
LOT =	000010	2597#						
LPO	026763	3016	3253	3764#	5496			
LPO0	026764	3764#	5493					
LP1	026773	3017	3764#					
LP2	027004	3018	3764#					
LP3	027012	3019	3764#					
LP4	027025	3020	3764#					
LSACP	002110	2461#						
LSAPT	002036	2419#						
LSAU	052030	2446	6104#					
LSAUT	002070	2445#						
LSAUTO	051762	2462	6041#					
LSCCP	002106	2459#						
LSCLEA	051764	2460	6055#					
LSCO	002032	2415#						
LSDEPO	002011	2397#						
LSDESC	023302	2452	3748#					
LSDESP	002076	2451#						
LSDEVP	002060	2437#						
SDISP	002124	2422	2481#					
SDLY	002116	2467#						
SDTP	002040	2421#						
SDTYP	002034	2417#						
SDU	052022	2448	6082#					
SDUT	002072	2447#						
SDVTY	023264	2438	3737#					
SEF	002052	2432#						
SENV1	002044	2425#						
SETP	002102	2455#						
SEXP1	002046	2427#						
SEXP4	002064	2441#						
SEXP5	002066	2443#						
SHARD	066562	2404	8687	8688#				
SHIME	002120	2469#						
SHPCP	002016	2403#						
SHPTP	002022	2407#						
SHW	002130	2408	2494	2495#				
SICP	002104	2457#						
SINIT	050750	2458	5842#					
SLADP	002026	2411#						
SLAST	067222	2412	8775#					

NOD12	021264	3465#
NOD120	022274	3558#
NOD121	022310	3559#
NOD122	022314	3560#
NOD123	022330	3561#
NOD124	022334	3562#
NOD125	022350	3563#
NOD126	022354	3564#
NOD127	022370	3565#
NOD13	021270	3466#
NOD130	022374	3566#
NOD131	022410	3567#
NOD132	022414	3568#
NOD133	022434	3569#
NOD134	022440	3572#
NOD135	022444	3573#
NOD136	022450	3574#
NOD137	022454	3575#
NOD14	021304	3467#
NOD140	022460	3576#
NOD141	022464	3577#
NOD142	022470	3578#
NOD143	022472	3581#
NOD144	022476	3582#
NOD145	022502	3583#
NOD146	022516	3584#
NOD147	022522	3585#
NOD15	021310	3468#
NOD150	022536	3586#
NOD151	022542	3589#
NOD152	022546	3590#
NOD153	022552	3591#
NOD154	022556	3594#
NOD155	022562	3597#
NOD156	022604	3598#
NOD157	022610	3599#
NOD16	021324	3469#
NOD160	022624	3600#
NOD161	022630	3601#
NOD162	022652	3602#
NOD163	022656	3603#
NOD164	022700	3604#
NOD165	022704	3607#
NOD166	022710	3608#
NOD167	022714	3609#
NOD17	021330	3470#
NOD170	022720	3611#
NOD171	022724	3612#
NOD172	022740	3613#
NOD173	022742	3614#
NOD174	022762	3615#
NOD175	022766	3616#
NOD176	023002	3617#
NOD177	023006	3618#
NOD2	021176	3457#
NOD20	021344	3471#

NOD200	023020	3619#
NOD201	023024	3620#
NOD202	023036	3621#
NOD203	023042	3622#
NOD204	023046	3625#
NOD205	044772	4574#
NOD206	044776	4575#
NOD207	045002	4576#
NOD21	021350	3472#
NOD210	045004	4577#
NOD211	045020	4578#
NOD212	045022	4579#
NOD213	045036	4580#
NOD214	045040	4581#
NOD215	045052	4582#
NOD216	045056	4583#
NOD217	045070	4584#
NOD22	021354	3473#
NOD220	045072	4585#
NOD221	045104	4586#
NOD222	045110	4587#
NOD223	045114	4588#
NOD224	045120	4589#
NOD225	045134	4590#
NOD226	045136	4591#
NOD227	045152	4592#
NOD23	021366	3474#
NOD230	045154	4593#
NOD231	045172	4594#
NOD232	045176	4595#
NOD233	045202	4596#
NOD234	045204	4597#
NOD235	045206	4598#
NOD24	021372	3475#
NOD25	021404	3476#
NOD26	021410	3477#
NOD27	021412	3481#
NOD3	021200	3458#
NOD30	021416	3482#
NOD31	021432	3483#
NOD32	021436	3484#
NOD33	021454	3485#
NOD34	021460	3486#
NOD35	021476	3487#
NOD36	021502	3488#
NOD37	021520	3489#
NOD4	021214	3459#
NOD40	021524	3490#
NOD41	021542	3491#
NOD42	021546	3492#
NOD43	021572	3493#
NOD44	021576	3494#
NOD45	021602	3495#
NOD46	021620	3496#
NOD47	021624	3497#
NOD5	021216	3460#

SHTFL	026025	3764#	6962											
SHTIV	026541	3764#	6917											
SHTLP	026227	3764#	7045											
SHTLPA	026314	3764#	7035											
SHTLPB	026367	3764#	7056											
SHTLPC	026431	3764#	7069											
SHTLPD	026502	3764#	7075											
SHTNF	026163	3764#	4544	6933										
SHTRE	025725	3764#	6858											
SHTRH	025764	3764#	6867											
SHTUN	026113	3764#	6988											
SHTYP0	025645	2995	3764#											
SHTYP1	025654	2995	3764#											
SHTYP2	025661	2995	3764#											
SHTYP3	025666	2995	3764#											
SHTYP4	025673	2995	3764#											
SHTYP5	025701	2995	3764#											
SHTYP6	025706	2995	3764#											
SHTYP7	025714	2995	3764#											
SHTYTB	003314	2995#	6519											
SHWOP	047342	4358	5488#	6202	6532									
SIZE =	000012	2720#	3583	6554	6560									
SLST =	000057	2757#	3612											
SLTHEM	040074	3657	3764#											
SMSC	031710	3764#	4101											
SQD =	040000	2789#	3229											
SRXQ	031611	3764#	4060											
STADD	017312	3133#	4627	6535*										
START	051040	5855	5874#											
STATB =	000001	2651#	4118	5512	6685									
STATUS =	000016	2724#	3536											
STATYP	023102	3652#	5985*	5988*	5991*	7933								
STRCM	040115	3658	3764#											
STREAM	040310	3666	3764#											
STRMM	040201	3661	3764#											
STXC	031600	3764#	4055											
STXQ	031567	3764#	4050											
SVCGBL =	000000	2367#	2386	2395	2397	2399	2401	2403	2405	2407	2409	2411	2413	2415
		2417	2419	2421	2423	2425	2427	2429	2432	2435	2437	2439	2441	2443
		2445	2447	2449	2451	2453	2455	2457	2459	2461	2463	2465	2467	2469
		2481	2495	2496	3737	3748	3786	3803	3816	3835	3854	3870	3886	3985
		5806	5826	5842	6041	6055	6082	6104	8647	8656	8688	8775#	8776	
SVCINS =	000001	2367#	2387	2388	2389	2390	2391	2392	2393	2394	2396	2398	2400	2402
		2404	2406	2408	2410	2412	2414	2416	2418	2420	2422	2424	2426	2428
		2430	2431	2433	2434	2436	2438	2440	2442	2444	2446	2448	2450	2452
		2454	2456	2458	2460	2462	2464	2466	2468	2470	2480	2482	2494	3738
		3741	3749	3756	3788	3789	3790	3791	3792	3793	3794	3795	3796	3797
		3800	3805	3806	3807	3808	3809	3810	3813	3818	3819	3820	3821	3822
		3823	3824	3827	3837	3838	3839	3840	3841	3842	3843	3846	3856	3857
		3858	3859	3860	3861	3862	3863	3866	3872	3873	3874	3875	3876	3877
		3878	3879	3880	3883	3888	3889	3890	3891	3892	3893	3894	3895	3896
		3899	3902	3903	4011	4127	4128	4129	4130	4131	4135	4136	4137	4138
		4139	4172	4173	4174	4175	4176	4183	4184	4185	4186	4187	4188	4189
		4190	4200	4201	4202	4203	4204	4209	4210	4211	4212	4213	4271	4272
		4273	4274	4275	4294	4295	4296	4297	4298	4304	4305	4306	4307	4308
		4309	4310	4311	4312	4320	4321	4322	4323	4324	4325	4326	4335	4336

4337	4338	4339	4340	4342	4343	4344	4345	4346	4347	4348	4370	4371	
4372	4373	4374	4375	4376	4378	4379	4380	4381	4382	4383	4384	4393	
4394	4395	4396	4397	4398	4399	4400	4409	4410	4411	4412	4413	4414	
4415	4416	4426	4427	4428	4429	4430	4448	4449	4450	4451	4452	4468	
4469	4470	4471	4472	4479	4480	4481	4482	4483	4484	4510	4511	4512	
4513	4514	4515	4543	4544	4545	4546	4547	4548	4560	4561	4562	4563	
4564	4565	4630	4631	4632	4633	4634	4635	4640	4641	4642	4643	4644	
4645	4646	4649	4650	4651	4652	4653	4654	4694	4695	4696	4697	4698	
4875	4876	4877	4878	4879	4888	4889	4890	4891	4892	4893	4895	4896	
4897	4898	4899	4900	4901	4902	5500	5501	5502	5503	5504	5505	5506	
5507	5508	5531	5532	5533	5534	5535	5536	5537	5538	5539	5681	5682	
5683	5684	5685	5724	5725	5726	5727	5728	5817	5849	5852	5853	5855	
5857	5858	5860	5862	5863	5865	5868	5869	5871	5876	5881	5882	5883	
5885	5891	5892	5893	5895	5904	5906	5911	5912	5913	5914	5915	5916	
5917	5918	5924	5925	5926	5927	5928	5945	5946	5947	5949	5994	5995	
5996	5997	5998	5999	6004	6005	6006	6007	6008	6009	6011	6012	6013	
6014	6015	6016	6019	6020	6022	6023	6030	6046	6062	6063	6065	6066	
6073	6086	6087	6094	6108	6109	6116	6146	6147	6148	6149	6150	6192	
6193	6194	6195	6196	6205	6207	6211	6212	6221	6222	6223	6224	6225	
6226	6227	6228	6238	6239	6240	6241	6242	6247	6248	6249	6250	6251	
6267	6268	6283	6284	6285	6286	6287	6288	6298	6299	6300	6301	6302	
6303	6324	6325	6326	6327	6328	6329	6341	6342	6343	6344	6345	6346	
6448	6449	6450	6451	6452	6453	6518	6519	6520	6521	6522	6523	6524	
6565	6566	6567	6568	6569	6595	6596	6597	6598	6599	6672	6673	6674	
6675	6676	6721	6722	6723	6724	6725	6758	6759	6760	6761	6762	6763	
6786	6787	6788	6789	6790	6791	6793	6794	6795	6796	6797	6798	6799	
6800	6816	6817	6818	6819	6820	6821	6822	6823	6824	6826	6827	6828	
6829	6830	6831	6832	6833	6837	6838	6839	6840	6841	6842	6843	6844	
6858	6859	6860	6861	6862	6867	6868	6869	6870	6871	6876	6877	6878	
6879	6880	6881	6882	6887	6888	6889	6890	6891	6897	6898	6899	6900	
6901	6916	6917	6918	6919	6920	6921	6932	6933	6934	6935	6936	6937	
6961	6962	6963	6964	6965	6966	6987	6988	6989	6990	6991	6992	7002	
7003	7004	7005	7006	7022	7023	7024	7025	7026	7035	7036	7037	7038	
7039	7045	7046	7047	7048	7049	7056	7057	7058	7059	7060	7069	7070	
7071	7072	7073	7075	7076	7077	7078	7079	7097	7098	7099	7100	7101	
7466	7467	7468	7469	7484	7485	7486	7487	7497	7498	7499	7500	7575	
7576	7577	7578	7579	7580	7581	7582	7606	7607	7608	7609	7610	7615	
7616	7617	7618	7690	7691	7692	7693	7694	7695	7696	7730	7731	7732	
7733	7734	7735	7736	7737	7790	7791	7792	7793	7794	7813	7814	7815	
7816	7817	7868	7869	7870	7871	7882	7892	7893	7894	7895	7908	7909	
7910	7911	7921	8137	8138	8139	8140	8145	8164	8165	8166	8167	8190	
8191	8192	8193	8347	8348	8349	8350	8354	8463	8464	8465	8466	8508	
8509	8510	8511	8512	8513	8517	8518	8519	8520	8521	8522	8523	8524	
8525	8545	8546	8547	8548	8549	8550	8551	8552	8553	8554	8555	8556	
8557	8558	8559	8567	8581	8582	8583	8584	8596	8601	8602	8603	8604	
8610	8653	8661	8670	8687	8694	8695	8696	8704	8705	8706	8707	8709	
8710	8711	8712	8714	8715	8716	8717	8718	8720	8721	8722	8723	8724	
8726	8727	8728	8730	8732	8733	8734	8737	8772	8773	8774			
2367#													
SVCSUB= 000001													
SVCTAG= 000001	2367#	2525	3799	3812	3826	3845	3865	3882	3898	4010	4191	4903	5816
	5919	6029	6045	6072	6093	6115	6229	6801	6834	6845	7583	7738	8652
	8660	8669	8738										
	2367#	6131											
SVCTST= 000001	2367#	2526#	3800#	3813#	3827#	3846#	3866#	3883#	3899#	4011#	4184	4191	4192#
S&LSYM= 010000	4896	4903	4904#	5817#	5912	5919	5920#	6030#	6046#	6073#	6094#	6116#	6222
	6229	6230#	6794	6801	6802#	6827	6834	6835#	6838	6845	6846#	7576	7583

ENDINI	1#	2367#	6028															
ENDMOD	1#	2367#	8776															
ENDMSG	1#	2367#	3798	3811	3825	3844	3864	3881	3897									
ENDPRO	1#	2367#	5832															
ENDPTA	1#	2367#																
ENDRPT	1#	2367#	5815															
ENDSEG	1#	2367#																
ENDSET	1#	2367#																
ENDSFT	1#	2367#																
ENDSRV	1#	2367#	4009	8651	8659													
ENDSUB	1#	2367#																
ENDSW	1#	2367#																
ENDTST	1#	2367#	8668															
EQUALS	1#	2367#	2541															
ERRDF	1#	2367#																
ERRHRD	1#	2367#																
ERROR	1#	2367#																
ERRSF	1#	2367#																
ERRSOF	1#	2367#	7465	7483	7496	7614	7867	7891	7907	8136	8163	8189	8346	8462	8580			
	8600																	
ERRTBL	1#	2367#																
ESCAPE	1#	2367#																
EXIT	1#	2367#	3901	6021	6064	6085	6107	6210	6266									
FEQUAL	1#	2367#																
GETBYT	1#	2367#																
GETPRI	1#	2367#																
GETWOR	1#	2367#																
GMANIA	1#	2367#																
GMANID	1#	2367#	4182	4894	5910	6220	6792	6825	6836	7574	7729							
GMANIL	1#	2367#																
GPHARD	1#	2367#	5944															
GPRMA	1#	2367#	8703	8708														
GPRMD	1#	2367#	4183#	4186	4895#	4898	5911#	5914	6221#	6224	6793#	6796	6826#	6829	6837#			
	6840		7578	7730#	7733	8713	8719											
GPRML	1#	2367#	8693	8725	8731													
HEADER	1#	2367#	2385															
INLOOP	1#	2367#																
IOSETU	1#	2367#																
IOSTAR	1#	2367#																
KT11	1#	2367#																
LASTAD	1#	2367#	8771															
MANUAL	1#	2367#	6204															
MEMORY	1#	2367#																
MSBYTE	1#	2367#	2386#	2392	2393	2394												
MSCHEC	1#	2367#	3902#	6022#	6065#	6086#	6108#	6211#	6267#									
MSCNT0	1#	2367#	4186#	4898#	5914#	6224#	6796#	6829#	6840#	7578#	7733#	8694#	8704#	8709#	8714#			
	8720#		8726#	8732#														
MSCOUN	1#	2367#	3788#	3805#	3818#	3837#	3856#	3872#	3888#	4127#	4135#	4172#	4200#	4209#	4271#			
	4294#		4304#	4320#	4335#	4342#	4370#	4378#	4393#	4409#	4426#	4448#	4468#	4479#	4510#	4543#		
	4560#		4630#	4640#	4649#	4694#	4875#	4888#	5500#	5531#	5681#	5724#	5924#	6146#	6192#	6238#		
	6247#		6283#	6298#	6324#	6341#	6448#	6518#	6565#	6595#	6672#	6721#	6758#	6786#	6816#	6858#		
	6867#		6876#	6887#	6897#	6916#	6932#	6961#	6987#	7002#	7022#	7035#	7045#	7056#	7069#	7075#		
	7097#		7606#	7690#	7790#	7813#	8508#	8517#	8545#									
MSDATA	1#	2367#	2386#	2395	2397	2399	2401	2403	2405	2407	2409	2411	2413	2415	2417			
	2419		2421	2423	2425#	2427	2429	2432	2435	2437	2439	2441	2443	2445	2447	2449		
	2451		2453	2455	2457	2459	2461	2463	2465	2467	2469	3737#	3748#					

MSDECR	1#	2367#	2525#	3799#	3812#	3826#	3845#	3865#	3882#	3898#	4010#	5816#	5833#	6029#	6045#
	6072#	6093#	6115#	8652#	8660#	8669#	8737#	8777#							
MSDEFA	1#	2367#	4186#	4898#	5914#	6224#	6796#	6829#	6840#	7578#	7733#	8694#	8704#	8709#	8714#
	8720#	8726#	8732#												
MSEENDE	1#	2367#	2525#	3799#	3812#	3826#	3845#	3865#	3882#	3898#	4010#	5816#	6029#	6045#	6072#
	6093#	6115#	8652#	8660#	8669#	8737#	8777#								
MSERRI	1#	2367#	7466#	7484#	7497#	7615#	7868#	7892#	7908#	8137#	8164#	8190#	8347#	8463#	8581#
	8601#														
MSESCA	1#	2367#													
MSESCS	1#	2367#													
MSEXCP	1#	2367#	4186#	4898#	5914#	6224#	6796#	6829#	6840#	7578#	7733#	8704#	8709#	8714#	8720#
MSEXIT	1#	2367#	3902#	6022#	6023	6065#	6066	6086#	6108#	6211#	6212	6267#	6268		
MSEXSE	1#	2367#	3902#	6022#	6065#	6086#	6108#	6211#	6267#						
MSEXTJ	1#	2367#	3902#	3903	6022#	6065#	6086#	6087	6108#	6109	6211#	6267#			
MSGEN	1#	2367#	2386#	2395#	2397#	2399#	2401#	2403#	2405#	2407#	2409#	2411#	2413#	2415#	2417#
	2419#	2421#	2423#	2425#	2427#	2429#	2432#	2435#	2437#	2439#	2441#	2443#	2445#	2447#	2449#
	2451#	2453#	2455#	2457#	2459#	2461#	2463#	2465#	2467#	2469#	2481#	2495#	2496#	2525#	3737#
	3748#	3786#	3799#	3803#	3812#	3816#	3826#	3835#	3845#	3854#	3865#	3870#	3882#	3886#	3898#
	3985#	4010#	4191#	4903#	5806#	5816#	5826#	5842#	5919#	6029#	6041#	6045#	6055#	6072#	6082#
	6093#	6104#	6115#	6131#	6229#	6801#	6834#	6845#	7583#	7738#	8647#	8652#	8656#	8660#	8669#
	8688#	8738#	8775#												
MSGENB	1#	2367#	4183#	4184	4895#	4896	5911#	5912	6221#	6222	6793#	6794	6826#	6827	6837#
	6838	7575#	7576	7730#	7731										
MSGETS	1#	2367#	2525#	3799#	3812#	3826#	3845#	3865#	3882#	3898#	4010#	5816#	5833#	6029#	6045#
	6072#	6093#	6115#	8652#	8660#	8669#	8730#	8737#	8777#						
MSGETT	1#	2367#	3902#	6022#	6065#	6086#	6108#	6211#	6267#	8730#					
MSGNGB	1#	2367#	2369#	2386#	2395#	2397#	2399#	2401#	2403#	2405#	2407#	2409#	2411#	2413#	2415#
	2417#	2419#	2421#	2423#	2425#	2427#	2429#	2432#	2435#	2437#	2439#	2441#	2443#	2445#	2447#
	2449#	2451#	2453#	2455#	2457#	2459#	2461#	2463#	2465#	2467#	2469#	2480#	2481	2494#	2495
	2496	3737#	3748#	3786#	3803#	3816#	3835#	3854#	3870#	3886#	3985#	5806#	5826#	5842#	6041#
	6055#	6082#	6104#	8647#	8656#	8687#	8688	8772#	8775						
MSGNIN	1#	2367#	2386#	2387	2388	2389	2390	2391	2392#	2393#	2394#	2395#	2396	2397#	2398
	2399#	2400	2401#	2402	2403#	2404	2405#	2406	2407#	2408	2409#	2410	2411#	2412	2413#
	2414	2415#	2416	2417#	2418	2419#	2420	2421#	2422	2423#	2424	2425#	2426	2427#	2428
	2429#	2430	2431	2432#	2433	2434#	2435#	2436	2437#	2438	2439#	2440	2441#	2442	2443#
	2444	2445#	2446	2447#	2448	2449#	2450	2451#	2452	2453#	2454	2455#	2456	2457#	2458
	2459#	2460	2461#	2462	2463#	2464	2465#	2466	2467#	2468	2469#	2470	2480#	2482#	2494#
	3737#	3738	3741	3748#	3749	3756	3788#	3789	3790#	3791	3792#	3793#	3794#	3795	3796#
	3797	3800#	3805#	3806#	3807#	3808	3809#	3810	3813#	3818#	3819#	3820#	3821#	3822	3823#
	3824	3827#	3837#	3838#	3839#	3840#	3841	3842#	3843	3846#	3856#	3857#	3858#	3859#	3860#
	3861	3862#	3863	3866#	3872#	3873	3874#	3875#	3876#	3877#	3878	3879#	3880	3883#	3888#
	3889	3890#	3891#	3892#	3893#	3894	3895#	3896	3899#	3902#	3903#	4010#	4011	4127#	4128#
	4129	4130#	4131	4135#	4136#	4137	4138#	4139	4172#	4173#	4174	4175#	4176	4183#	4184#
	4185#	4186#	4187	4188	4189	4190	4200#	4201#	4202	4203#	4204	4209#	4210#	4211	4212#
	4213	4271#	4272#	4273	4274#	4275	4294#	4295#	4296	4297#	4298	4304#	4305#	4306#	4307#
	4308#	4309#	4310	4311#	4312	4320#	4321#	4322#	4323#	4324	4325#	4326	4335#	4336#	4337#
	4338	4339#	4340	4342#	4343#	4344#	4345#	4346	4347#	4348	4370#	4371#	4372#	4373#	4374
	4375#	4376	4378#	4379#	4380#	4381#	4382	4383#	4384	4393#	4394#	4395#	4396#	4397#	4398
	4399#	4400	4409#	4410#	4411#	4412#	4413#	4414	4415#	4416	4426#	4427#	4428	4429#	4430
	4448#	4449#	4450	4451#	4452	4468#	4469#	4470	4471#	4472	4479#	4480#	4481#	4482	4483#
	4484	4510#	4511#	4512#	4513	4514#	4515	4543#	4544#	4545#	4546	4547#	4548	4560#	4561#
	4562#	4563	4564#	4565	4630#	4631#	4632#	4633	4634#	4635	4640#	4641	4642#	4643#	4644
	4645#	4646	4649#	4650#	4651#	4652	4653#	4654	4694#	4695#	4696	4697#	4698	4875#	4876#
	4877	4878#	4879	4888#	4889#	4890#	4891	4892#	4893	4895#	4896#	4897#	4898#	4899	4900
	4901	4902	5500#	5501#	5502#	5503#	5504#	5505#	5506	5507#	5508	5531#	5532#	5533#	5534#
	5535#	5536#	5537	5538#	5539	5681#	5682#	5683	5684#	5685	5724#	5725#	5726	5727#	5728

M\$MCL0	1#	2367#															
M\$MSK1	1#	2367#															
M\$POP	1#	2367#	2525#	3799#	3812#	3826#	3845#	3865#	3882#	3898#	4010#	5816#	5833#	6029#	6045#		
6072#		6093#	6115#	8652#	8660#	8669#	8737#	8777#									
M\$PRIN	1#	2367#	3788#	3805#	3818#	3837#	3856#	3872#	3888#	4127#	4135#	4172#	4200#	4209#	4271#		
4294#		4304#	4320#	4335#	4342#	4370#	4378#	4393#	4409#	4426#	4448#	4468#	4479#	4510#	4543#		
4560#		4630#	4640#	4649#	4694#	4875#	4888#	5500#	5531#	5681#	5724#	5924#	6146#	6192#	6238#		
6247#		6283#	6298#	6324#	6341#	6448#	6518#	6565#	6595#	6672#	6721#	6758#	6786#	6816#	6858#		
6867#		6876#	6887#	6897#	6916#	6932#	6961#	6987#	7002#	7022#	7035#	7045#	7056#	7069#	7075#		
7097#		7606#	7690#	7790#	7813#	8508#	8517#	8545#									
M\$PUSH	1#	2367#	2369#	2494#	3786#	3803#	3816#	3835#	3854#	3870#	3886#	3985#	5806#	5826#	5842#		
6041#		6055#	6082#	6104#	6131#	6132	8647#	8656#	8687#								
M\$PUT	1#	2367#	3788#	3805#	3818#	3837#	3856#	3872#	3888#	4127#	4135#	4172#	4200#	4209#	4271#		
4294#		4304#	4320#	4335#	4342#	4370#	4378#	4393#	4409#	4426#	4448#	4468#	4479#	4510#	4543#		
4560#		4630#	4640#	4649#	4694#	4875#	4888#	5500#	5531#	5681#	5724#	5924#	5994#	6004#	6011#		
6146#		6192#	6238#	6247#	6283#	6298#	6324#	6341#	6448#	6518#	6565#	6595#	6672#	6721#	6758#		
6786#		6816#	6858#	6867#	6876#	6887#	6897#	6916#	6932#	6961#	6987#	7002#	7022#	7035#	7045#		
7056#		7069#	7075#	7097#	7606#	7690#	7790#	7813#	8508#	8517#	8545#						
M\$PUT1	1#	2367#	3788#	3790	3792	3793	3794	3805#	3806	3807	3818#	3819	3820	3821	3837#		
3838		3839	3840	3856#	3857	3858	3859	3860	3872#	3874	3875	3876	3877	3888#	3890		
3891		3892	3893	4127#	4128	4135#	4136	4172#	4173	4200#	4201	4209#	4210	4271#	4272		
4294#		4295	4304#	4305	4306	4307	4308	4309	4320#	4321	4322	4323	4335#	4336	4337		
4342#		4343	4344	4345	4370#	4371	4372	4373	4378#	4379	4380	4381	4393#	4394	4395		
4396		4397	4409#	4410	4411	4412	4413	4426#	4427	4448#	4449	4468#	4469	4479#	4480		
4481		4510#	4511	4512	4543#	4544	4545	4560#	4561	4562	4630#	4631	4632	4640#	4642		
4643		4649#	4650	4651	4694#	4695	4875#	4876	4888#	4889	4890	5500#	5501	5502	5503		
5504		5505	5531#	5532	5533	5534	5535	5536	5681#	5682	5724#	5725	5924#	5925	5994#		
5995		5996	5997	6004#	6005	6006	6007	6011#	6012	6013	6014	6146#	6147	6192#	6193		
6238#		6239	6247#	6248	6283#	6284	6285	6298#	6299	6300	6324#	6325	6326	6341#	6342		
6343		6448#	6449	6450	6518#	6519	6520	6521	6565#	6566	6595#	6596	6672#	6673	6721#		
6722		6758#	6759	6760	6786#	6787	6788	6816#	6818	6820	6821	6858#	6859	6867#	6868		
6876#		6878	6879	6887#	6888	6897#	6898	6916#	6917	6918	6932#	6933	6934	6961#	6962		
6963		6987#	6988	6989	7002#	7003	7022#	7023	7035#	7036	7045#	7046	7056#	7057	7069#		
7070		7075#	7076	7097#	7098	7606#	7607	7690#	7691	7692	7693	7790#	7791	7813#	7814		
8508#		8509	8510	8517#	8519	8521	8522	8545#	8547	8549	8551	8553	8555	8556			
M\$RADI	1#	2367#	4186#	4898#	5914#	6224#	6796#	6829#	6840#	7578#	7733#	8694#	8704#	8709#	8714#		
8720#		8726#	8732#														
M\$RBRO	1#	2367#															
M\$RNRO	1#	2367#	5881#	5883	5891#	5893	5945#	5947									
M\$SETS	1#	2367#	2369#	2494#	3786#	3803#	3816#	3835#	3854#	3870#	3886#	3985#	5806#	5826#	5842#		
6041#		6055#	6082#	6104#	6132#	8647#	8656#	8687#									
M\$STAR	1#	2367#															
M\$SVC	1#	2367#	3788#	3796	3799#	3800	3805#	3809	3812#	3813	3818#	3823	3826#	3827	3837#		
3842		3845#	3846	3856#	3862	3865#	3866	3872#	3879	3882#	3883	3888#	3895	3898#	3899		
3902#		4127#	4130	4135#	4138	4172#	4175	4183#	4200#	4203	4209#	4212	4271#	4274	4294#		
4297		4304#	4311	4320#	4325	4335#	4339	4342#	4347	4370#	4375	4378#	4383	4393#	4399		
4409#		4415	4426#	4429	4448#	4451	4468#	4471	4479#	4483	4510#	4514	4543#	4547	4560#		
4564		4630#	4634	4640#	4645	4649#	4653	4694#	4697	4875#	4878	4888#	4892	4895#	5500#		
5507		5531#	5538	5681#	5684	5724#	5727	5816#	5817	5849#	5852#	5853	5857#	5858	5862#		
5863		5868#	5869	5876#	5881#	5882	5891#	5892	5904#	5911#	5924#	5927	5945#	5946	5994#		
5998		6004#	6008	6011#	6015	6019#	6020	6022#	6029#	6030	6045#	6046	6062#	6063	6065#		
6072#		6073	6086#	6093#	6094	6108#	6115#	6116	6146#	6149	6192#	6195	6205#	6211#	6221#		
6238#		6241	6247#	6250	6267#	6283#	6287	6298#	6302	6324#	6328	6341#	6345	6448#	6452		
6518#		6523	6565#	6568	6595#	6598	6672#	6675	6721#	6724	6758#	6762	6786#	6790	6793#		
6816#		6823	6826#	6837#	6858#	6861	6867#	6870	6876#	6881	6887#	6890	6897#	6900	6916#		
6920		6932#	6936	6961#	6965	6987#	6991	7002#	7005	7022#	7025	7035#	7038	7045#	7048		

RFLAGS	1#	2367#							
SETPRI	1#	2367#	6018	6061					
SETVEC	1#	2367#	5993	6003	6010				
SLASH	1#	2367#							
STARS	1#	2367#							
SVC	1#	2367#							
XFER	1#	2367#	3902#	6022#	6065#	6086#	6108#	6211#	6267#
XFERF	1#	2367#	8729						
XFERT	1#	2367#							

. ABS. 067222 000

ERRORS DETECTED: 0

CZCLMB/I,CZCLMB.SEQ/CRF/DOC/SOL=SVC34R.MLB,CZCLMB.P11
RUN-TIME: 30 38 5 SECONDS
RUN-TIME RATIO: 88/73=1.2
CORE USED: 24K (47 PAGES)

DOCUMENT PAGES: 255