

DPV-11

DPV-11 DCLT
CVCLHAO

AH-F584A-MC
FICHE 1 OF 1

NOV 1980
COPYRIGHT © 1980
MADE IN USA



The main body of the document is a large grid of approximately 20 columns and 20 rows of small, dense tables. Each cell in the grid contains a small table with multiple columns and rows of text, likely representing data points or a detailed index. The text is too small to be legible in this view.



1

.TITLE CVCLHA DPV-11 DATA COMM. LINK TEST

.REM 8

IDENTIFICATION

PRODUCT CODE: AC-F582A-MC
PRODUCT NAME: CVCLHA0 DPV-11 DATA COMM. LINK TEST
PRODUCT DATE: 20-AUG-80
MAINTAINER: MERRIMACK DIAGNOSTIC ENGINEERING
AUTHOR: BRUCE RIBOLINI-BRUCE LUHRS

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) 1980 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

REVISION HISTORY:

REV ---	DATE ---	AUTHOR -----	REASON -----
A	20-AUG-80	BRUCE RIBOLINI BRUCE LUHRS	ORIGINAL ISSUE, DCLT FOR THE DPV-11

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 RUN COMMAND
 - 2.5.3 DEFAULTS
 - 2.6 QUICK STARTUP PROCEDURE
- 3.0 ERROR INFORMATION
 - 3.1 TYPES OF ERROR MESSAGES
 - 3.2 SPECIFIC 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
 - 6.2 MESSAGE DESCRIPTIONS
 - 6.3 INTERFACING TO AN "ITEP" NODE
 - 6.4 TROUBLESHOOTING HINTS

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 DPV-11 TO DPV-11 COMMUNICATION LINKS. THIS DCLT PROGRAM WILL PROVIDE THE COVERAGE NECESSARY TO DETECT FAILURES TO 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 DPV DCLT PROGRAM, THE FOLLOWING MINIMUM HARDWARE IS REQUIRED:

- A LSI-11 CPU
- MINIMUM OF 24K WORDS OF MEMORY
- A WORKING CLOCK
- A CONSOLE TERMINAL
- ANY XXDP+ SUPPORTED LOAD MEDIA
- ONE OF THESE DPV-11 CONFIGURATIONS:

DPV11-DB
DPV11-DA

1.3 RELATED DOCUMENTS AND STANDARDS

- 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 DVP-11'S AT EACH END OF THE LINK HAVE ALREADY

BEEN TESTED.

IF A WORKING CLOCK IS NOT 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 DPV-11'S, BUT TO TEST THE COMMUNICATION LINK TO WHICH THEY ARE CONNECTED.

SOME OF THE DIAGNOSTICS THAT COULD BE RUN IF EITHER OF THE DPV-11'S LOOK BAD:

CVDPVAO DPV-11 FCTNL DIAG
CXDPVAO DPV-11 DECX MODULE

1.5 ASSUMPTIONS - RESTRICTIONS

IT IS ASSUMED THAT THE COMMUNICATIONS DEVICE (A DPV-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.

THIS DIAGNOSTIC DOES NOT RUN THE DPV IN BIT STUFF MODE
IT IS ASSUMED THAT IF THE LINK WORKS IN CHAR MODE THE LINK WILL WORK IN BIT STUFF MODE.

2.0 OPERATING INSTRUCTIONS

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

2.1 COMMANDS

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

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

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

2.2 SWITCHES

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

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

EXAMPLE OF SWITCH USAGE:

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

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

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

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

2.3 FLAGS

FLAGS ARE USED TO SET UP CERTAIN OPERATIONAL PARAMETERS SUCH AS LOOPING ON ERROR. ALL FLAGS ARE CLEARED AT STARTUP AND REMAIN CLEARED UNTIL EXPLICITLY SET USING THE FLAGS SWITCH. FLAGS ARE ALSO CLEARED AFTER A START COMMAND UNLESS SET USING THE FLAG SWITCH. THE ZFLAGS COMMAND MAY ALSO BE USED TO CLEAR ALL FLAGS. WITH THE EXCEPTION OF THE START AND ZFLAGS COMMANDS, NO COMMANDS AFFECT THE STATE OF THE FLAGS; THEY REMAIN SET OR CLEARED AS SPECIFIED BY THE LAST FLAG SWITCH.

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

IDR	INHIBIT PROGRAM DROPPING OF UNITS
ADR	EXECUTE AUTODROP CODE
LOT	LOOP ON TEST
EVL	EXECUTE EVALUATION (ON DIAGNOSTICS WHICH HAVE EVALUATION SUPPORT)

*ERROR MESSAGES ARE DESCRIBED IN SECTION 3.1

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

/FLAGS:LOE:IER:BOE

2.4 HARDWARE QUESTIONS

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

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

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

UNIT 0
FULL DUPLEX OPERATION : (L) Y ?
DPV CSR ADDRESS : (0) 160170 ?
INTERRUPT VECTOR ADDRESS: (0) 300 ?
REMOTE NODE "ITEP" : (L) Y ?
```

THE FULL DUPLEX QUESTION SHOULD BE ANSWERED "Y" WHEN USING FULL DUPLEX MODEMS, OR NULL MODEM, OR MODEM ELIMINATORS. ANSWER "N" FOR HALF DUPLEX MODEMS.

REMOTE NODE ITEP SHOULD BE ANSWERED "Y" IF OTHER NODE IS RUNNING SOFTWARE THAT IS USING "ITEP" FORMATS (I.E. COMM TURNAROUND SYSTEM OR PDP-11 RUNNING ITEP).

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.

MESSAGE COMMANDS AVAILABLE:

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 (00'S) AND THEN PUTS DEFAULT ITEP MSG IN SO NOT REALLY EMPTY
CLEAR TRANSMITLIST	FILLS TRANSMITLIST (000'S) AND THEN PUTS DEFAULT ITEP MSG IN SO NOT REALLY EMPTY
HELP ?	TYPES HELP INFO FOR OPERATOR
SET EXPECTMSG=TYPE/QUAL	DEFINE A MESSAGE TO BE PUT ON THE EXPECTED LIST
WHERE: "TYPE" IS:	
=ONES	
=ZEROES	
=1ALT	
=0ALT	
=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 OPERATOR OR DEFAULTS.)
- /COPY=NN 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 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
- PRINT PRINTS THE EVENT LOG
- 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.
IS PRINTED IN WORD FORMAT.

WHERE '/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.

2.5.2 RUN COMMAND

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	(DOWN-LINE-LOADING IS NOT SUPPORTED FOR DPV-11 TO DPV-11 LINKS).

(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 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.

/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.
(IGNORED IN MODES OTHER THAN PASSIVE)

/MODEM/NOMODEM/ ENABLES/DISABLES THE REPORTING OF MODEM STATUS
INTERRUPT CHANGES.

/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	LOOPS DATA INTERNAL TO USYNRT
=CABLE	USE THIS FOR TESTING WITH H3260 OR H3259 TURNARROUND CONNECTOR

NOTE: THIS SKIPS OVER THE CHECK
FOR MODEM READY WHEN DTR IS SET.

=LOCALMODEM NOT USED BY DPV.,
=REMOTEMODEM ..

/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'

2.5.3 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:

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{N 1} 10:29 PAGE 14

SEQ 0013

^C
<CR>
"RUBOUT"

ALWAYS RETURNS YOU TO 'DR>' (THE SUPERVISOR)
IS SEEN AS A COMMAND TERMINATOR
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

COMMAND LINE INTERPRETER ERRORS:

ERROR MESSAGE:

?ILL CMD-BAD SYNTAX?

MEANING

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.

DCLT OR DEVICE 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 THE CLOCK FOUND ON THE SYSTEM 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.

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

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

RECV=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 DIFFENT LENGTH THEN THE MESSAGE THAT WAS EXPECTED.

MODEM STATUS CHANGES FOR THIS PASS WERE..

HARD CHANGES=XXXXX GLITCHES=XXXXX

WHERE XXXXX IS A 5 DIGIT DECIMAL NUMBER
THIS MSG IS ONLY PRINTED IF NUMBER OF
EITHER HARD CHANGES OR GLITCHES IS
GREATER THAN 0. A HARD CHANGE IS ONE
WHERE THE DPV WAS ABLE TO LATCH UP A
DIFFERENCE IN THE MODEM STATUS. A
GLITCH IS WHEN A MODEM STATUS INTERRUPT
OCCURS BUT THE DPV CANNOT FIND A
DIFFERENCE IN STATUS BIT.

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

.ENDC

MASTER RESET DID NOT WORK
RXCSR TXCSR
XXXXXX XXXXXXXX

THIS MEANS THAT AFTER A MASTER
RESET WAS ISSUED TO DPV THE
RXCSR REGSITER WAS NON ZERO.

NO CLEAR TO SEND FROM MODEM
RXCSR TXCSR
XXXXXXX XXXXXXXX

WHEN REQUEST TO SEND SIGNAL
IS SET MODEM DOES NOT RESPOND
WITH CLEAR TO SEND

TIME OUT WAITING FOR RX OR TX TO COMPLETE
RXCSR TXCSR
XXXXXXX XXXXXXXX

DEVICE TIMED OUT AFTER
WAITING FOR A RECIEVER OR
TRANSMITTER TO COMPLETE

MODEM DID NOT RETURN MODEM READY
RXCSR TXCSR
XXXXXXX XXXXXXXX

MODEM DID NOT RESPOND WITH MR.

CRC IN ERROR
RDSR RXCSR
XXXXXX XXXXXX

THE CRC ERROR BIT IS CLEAR
AT TIME THAT IT SHOULD BE SET

RECEIVER OVERRUN
RDSR RXCSR
XXXXXXX XXXXXXXX

RECIEVER DETECTED AN OVERRUN

TIMED OUT IN START,STACK ACK SEQ
RDATA SDATA
XXXXXXX XXXXXXXX

THIS ERROR OCCURS WHEN IN THE
DEVICE INIT ROUTINE A TIME OUT
OCCURS WHILE TRYING TO DO START
STACK ACK START UP SEQUENCE.
THE VALUES IN RDATA AND SDATA
INDICATE THE LAST CONTROL CHAR
RECIEVED(RDATA) AND TRANSMITTED
(SDATA).

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" IS REPORTED WHEN " X NUMBER OF DCLT PASSES" ARE 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 "N" 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 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 AVAILIABLE 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" 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 1ST BYTE OF MESSAGE,
TOTAL NO. OF BYTES, MODEM STATUS AT THAT TIME.

TRANSMIT MESSAGE COMPLETED:

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

RECEIVE SPACE QUEUED:

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

RECEIVE MESSAGE COMPLETED:

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

DATA COMPARISON STARTED:

EVENT TIME, 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 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 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,
 AND THE 'STRT-TO'(COUNT OF START TIME OUTS).

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 TRANSMITING 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
MSC	ABNORMAL MODEM STATUS CHANGE
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] SPARE
.WORD	0	:[10] SPARE
.WORD	0	:[12] SPARE
.WORD	0	:[14] REMOTE NODE "ITEP"

6.0 MODE AND MESSAGE DESCRIPTIONS

6.1 MODE DESCRIPTIONS

THE FOLLOWING MODE DESCRIPTIONS REFER TO MESSAGE LISTS BEING TRANSMITTED AND RECEIVED BE AWARE THAT OTHER DATA IS ALSO SENT WITH THE MESSAGE. MESSAGE FORMATS ARE..... 177(OCTAL)SYNC CHARS;SOH CHAR(201),BYTE COUNT OF MSG.(2 BYTES);THREE BYTES OF OCTAL 001;2 BYTES OF CRC;THE MESSAGE BODY;2BYTES OF CRC. IF REMOTE NODE ITEP QUESTION IS ANSWERED Y THEN SYNC CHAR WILL BE OCTAL 26 INSTEAD OF 226;NO SOH,BYTE COUNT OR HEADER DATA WILL BE SENT. ON FULL DUPLEX LINKS A START STACK ACK SEQ WILL BE SENT ONCE AT INIT TIME FOR EACH MODE. THIS SEQUENCE CONTAINS CONTROL MESSAGES WHOSE FORMAT IS ..8 SYNC CHAR, 1BYTE ENQ(005),START STACK OR ACK,3 BYTES OF 001, AND 2BYTES OF CRC.

TRANSMIT MODE

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

RECEIVE MODE

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.

PASSIVE MODE

THEN 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.

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.

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

DOWN-LINE-LOAD

DOWN-LINE-LOADING IS NOT SUPPORTED FOR DPV-11 TO DPV-11 LINKS.

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

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".

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 !

INTERNALTTL LOOPS DATA INTERNAL TO THE USYNRT

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):

HALF DUPLEX START	STATION A "HOST" NODE	"/LOOP" ALLOWED?	STATION B "REMOTE" NODE	DUPLEX
B	TALK	NO	LISTEN*, RECEIVE	HALF OR FULL
A	LISTEN	NO	TALK*, TRANSMIT	HALF OR FULL
B	TRANSMIT	NO	RECEIVE*, LISTEN	HALF OR FULL
A	RECEIVE	NO	TRANSMIT*, TALK	HALF OR FULL
A	PASSIVE	NO	ACTIVE*	HALF OR FULL
-NA-	ACTIVE	YES	ACTIVE*	FULL
B	ACTIVE	YES	PASSIVE*	HALF OR FULL
-NA-	DOWNLINELOAD	** DOWN-LINE-LOADING IS NOT SUPPORTED FOR DPV-11 TO DPV-11 LINKS.		

*= MOST LIKELY TO BE IN THAT MODE

NOTE: H/D START COLUMN INDICATES WHICH NODE TO START FIRST ON A HALF DUPLEX LINK

6.2 MESSAGE DESCRIPTIONS

NAME	DESCRIPTION
ZERES	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.

6.3 INTERFACING TO AN "ITEP" NODE

THESE ARE THE RULES WHEN USING ITEP/WITH A DUP TO TALK TO A DPV USING DCLT.

ITEP NODE	DCLT NODE
-----------	-----------

ANSWER ALL QUESTION TO THE SET SWITCHES PROMPT.	ANSWER ALL QUESTIONS TO THE DCLT> PROMPT.
---	---

FOR ONE WAY OUT... SET SWITCHES TO 1221	CLEAR EXPECTED SET E=ITEP/S=56 RUN MODE=REC/STATUS/CHECK
--	--

NOTE: DUP ITEP SENDS ONLY 56 CHARS

FOR ONE WAY IN..... SET SWITCHES TO1222	RUN MODE=TRA/STATUS
---	---------------------

FOR EXTERNAL LOOPBACK.... SET SWITCHES.....1224	CLEAR EXPECTED SET EXP=ITEP/S=56 RUN MODE=ACTIVE/STATUS/CHECK
--	---

FOR INTERNAL LOOPBACK..... SET SWITCHES.....1260	CLEAR EXPECTED SET EXP=ITEP/S=56 RUN MODE=ACTIVE/STATUS/CHECK
---	---

NOTE: DO NOT USE SWITCH 8 WITH ITEP GOING TO DCLT
THE ONLY MESSG. DCLT SUPPORTS IS MSG 1.
DCLT IGNORES CRC ERRORS WHEN REC DATA FROM ITEP
BECAUSE ITPE SENDS NO CRC.

6.4 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.

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.

A POSSIBLE COMMAND SEQUENCE IS:

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

THIS GIVES YOU A IDEA IF THE COMM. DEVICE CAN EVEN 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 PREREQUISTE DIAGNOSTICS FOR THE COMM.
DEVICE.

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

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

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.
POSSIBLE COMMAND SEQUENCE IS:

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

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:

```
NODE A
-----
C E
```

```
NODE B
-----
C E
```

C T
SE T=1ALT/S=250
R M=TR/PAS=3

C T
SE E=1ALT/S=250
R M=R/NOCH/PAS=3

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

R M=TR/PAS=3

R M=R/CH/PAS=3

NOW RUN THRU THE SEQUENCE AGAIN WITH NODE A RECEIVING AND NODE B RECEIVING.

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:

NODE A

C E
C T
SE T=CCITT/S=10/C=2
R M=ACT/NOCH/PAS=3

NODE B

C E
C T
SE T=1ALT/S=20/C=2
R M=P/NOCH/PAS=3

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.

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.

NODE A

C E
C T
SE T=0ALT/S=10
SE T=CCITT/S=20
SE T=ALPHA/S=30
SE E=ZERO/S=11
SE E=ONES/S=21
SE E=ITEP/S=31
R M=A/CH/NOST/PAS=3

NODE B

C E
C T
SE E=0ALT/S=10
SE E=CCITT/S=20
SE E=ALPHA/S=30
SE T=ZERO/S=11
SE T=ONES/S=21
SE T=ITEP/S=31
R M=A/CH/NOST/PAS=3

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.

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.

R M=LIS/NOST
LIS>

R M=TA/NOST
TLK>

&

1286
1287
1288
1289
1290
1291
1292 002000
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303 002000
1304
1305
1306
1307
1308
1309 002000
1310 002000
1311 002000 103
1312 002001 126
1313 002002 103
1314 002003 114
1315 002004 110
1316 002005 000
1317 002006 000
1318 002007 000
1319 002010
1320 002010 101
1321 002011
1322 002011 060
1323 002012
1324 002012 000000
1325 002014
1326 002014 003410
1327 002016
1328 002016 033706
1329 002020
1330 002020 000000
1331 002022
1332 002022 002130
1333 002024
1334 002024 000000
1335 002026
1336 002026 034164
1337 002030
1338 002030 000000
1339 002032
1340 002032 000000
1341 002034

.SBTTL PROGRAM HEADER

BGNMOD

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

POINTER BGNRPT,BGNAU,BGNDU

HEADER CVCLH,A,0,1800.,0,#PRI07

LSNAME::
 .ASCII /C/
 .ASCII /V/
 .ASCII /C/
 .ASCII /L/
 .ASCII /M/
 .BYTE 0
 .BYTE 0
 .BYTE 0
LSREV::
 .ASCII /A/
LSDEPO::
 .ASCII /O/
LSUNIT::
 .WORD 0
LSTIML::
 .WORD 1800.
LSHPCP::
 .WORD LSHARD
LSSPCP::
 .WORD 0
LSHPTP::
 .WORD LSHW
LSSPTP::
 .WORD 0
LSLADP::
 .WORD LSLAST
LSSTA::
 .WORD 0
LSCO::
 .WORD 0
LSDTYP::

1342 002034 000000
1343 002036
1344 002036 000000
1345 002040
1346 002040 002124
1347 002042
1348 002042 000340
1349 002044
1350 002044 000000
1351 002046
1352 002046 000000
1353 002050
1354 002050 003
1355 002051 003
1356 002052
1357 002052 000000
1358 002054 000000
1359 002056
1360 002056 000000
1361 002060
1362 002060 011422
1363 002062
1364 002062 022720
1365 002064
1366 002064 000000
1367 002066
1368 002066 000000
1369 002070
1370 002070 023612
1371 002072
1372 002072 023604
1373 002074
1374 002074 000000
1375 002076
1376 002076 011432
1377 002100
1378 002100 104035
1379 002102
1380 002102 000000
1381 002104
1382 002104 022734
1383 002106
1384 002106 023562
1385 002110
1386 002110 023560
1387 002112
1388 002112 022726
1389 002114
1390 002114 000000
1391 002116
1392 002116 000000
1393 002120
1394 002120 000000
1395

LSAPT:: .WORD 0
LSDTP:: .WORD 0
LSPRIO:: .WORD LSDISPATCH
LSENV1:: .WORD #PRI07
LSEXP1:: .WORD 0
LSMREV:: .WORD 0
LSEF:: .BYTE CSREVISION
 .BYTE CSREDIT
LSSPC:: .WORD 0
LSEVP:: .WORD 0
LSREPP:: .WORD LSDVTYP
LSEXP4:: .WORD LSRPT
LSEXP5:: .WORD 0
LSAUT:: .WORD 0
LSDUT:: .WORD LSAU
LSLUN:: .WORD LSDU
LSDESP:: .WORD 0
LSLOAD:: .WORD LDESC
LSETP:: EMT ESLOAD
LSICP:: .WORD 0
LSCCP:: .WORD LSINIT
LSACP:: .WORD LSCLEAN
LSPRT:: .WORD LSAUTO
LSTEST:: .WORD LSPROT
LSDLY:: .WORD 0
LSHIME:: .WORD 0

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{F 3} 10:29 PAGE 32
DISPATCH TABLE

SEQ 0031

1396
1397
1398
1399
1400
1401
1402
1403 002122
1404 002122 000001
1405 002124
1406 002124 023620
1407

.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

.WORD 1
LSDISPATCH::
.WORD T1

1408
1409
1410
1411
1412
1413
1414
1415
1416
1417 002126
1418 002126 000010
1419 002130
1420 002130
1421
1422
1423
1424
1425
1426
1427
1428 002130 000001
1429
1430
1431
1432
1433
1434
1435
1436 002132 160170
1437 002134 000300
1438 002136 000240
1439 002140 000000
1440 002142 000000
1441 002144 000000
1442 002146 000000
1443
1444
1445 002150
1446 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-LSHW/2
LSHW::
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 PARAMETER CODING SECTION BY CHANGING "OFFSETS"

.WORD 160170 ;[2] CSR ADDRESS
.WORD 300 ;[4] INTERRUPT VECTOR
.WORD 240 ;[6] INTERRUPT PRIORITY (5)
.WORD 0 ;[10] SPARE
.WORD 0 ;[12] SPARE
.WORD 0 ;[14] OTHER NODE "ITEP"
.WORD 0 ;[16] SPARE

ENDHW

L10000:

1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502

002150

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

000040
000037
000036
000035
000034

.SBTTL GLOBAL EQUATES SECTION

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

EQUALS

: BIT DEFINITIONS

:
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.
EF.RESTART== 31.
EF.CONTINUE== 30.
EF.NEW== 29.
EF.PWR== 28.
:

: START COMMAND WAS ISSUED
: RESTART COMMAND WAS ISSUED
: CONTINUE COMMAND WAS ISSUED
: A NEW PASS HAS BEEN STARTED
: A POWER-FAIL/POWER-UP OCCURRED

1503		:
1504		: PRIORITY LEVEL DEFINITIONS
1505		:
1506	000340	PRI07== 340
1507	000300	PRI06== 300
1508	000240	PRI05== 240
1509	000200	PRI04== 200
1510	000140	PRI03== 140
1511	000100	PRI02== 100
1512	000040	PRI01== 40
1513	000000	PRI00== 0
1514		:
1515		: OPERATOR FLAG BITS
1516		:
1517	000004	EVL== 4
1518	000010	LOT== 10
1519	000020	ADR== 20
1520	000040	IDU== 40
1521	000100	ISR== 100
1522	000200	UAM== 200
1523	000400	BOE== 400
1524	001000	PNT== 1000
1525	002000	PRI== 2000
1526	004000	IXE== 4000
1527	010000	IBE== 10000
1528	020000	IER== 20000
1529	040000	LOE== 40000
1530	100000	HOE== 100000
1531		

```
1532 ;***** INDEPENDENT EQUATES
1533
1534 001000 BUFLIM=512. ;MAX BUFFER SIZE IN BYTES
1535 ; APPLIES TO TX,RX AND CMP BUFFS
1536 000017 MSGLIM=15. ;MAX NO. OF MESSAGES PER BUFFER
1537 ; (FOR EACH INCREMENT (+1) TO MSGLIM,
1538 ; ADD 6 WORDS TO THE POINTER TABLE
1539 ; (PTRTAB:) SINCE THIS MEANS 2 MORE
1540 ; 'POINTER' WORDS PER BUFFER.
1541
1542 ;MODE OF OPERATION EQUATES
1543 000000 REC=0 ;RECEIVE MODE
1544 000001 TRA=1 ;TRANSMIT MODE
1545 000002 PAS=2 ;PASSIVE MODE
1546 000003 ACT=3 ;ACTIVE MODE
1547 000004 DOW=4 ;DOWN-LINE-LOAD MODE
1548 000005 TAL=5 ;TALK MODE
1549 000006 LIS=6 ;LISTEN MODE
1550
1551 ;MAINT LOOP TYPE EQUATES
1552 000000 NONE= 0 ;NO LOOP
1553 000001 TTL= 1 ;INTERNAL TTL
1554 000002 CABLE= 2 ;CABLE LOOP
1555 000003 MODLOC= 3 ;MODEM LOCAL
1556 000004 MODREM= 4 ;MODEM REMOTE
1557 000005 MOP= 5 ;MOP
1558
1559 ;CLOCK ENABLE VALUES TO BE LOADED IN CLK'S CSR
1560 000100 LCLKEN= 100 ;L-CLOCK CSR VALUE TO ENABLE THE CLOCK
1561 000111 PCLKEN= 111 ;P-CLOCK CSR VALUE TO ENABLE THE CLOCK
1562 001600 PCLKCT= 1600 ;P-CLOCK COUNT SET REGISTER FOR COUNTER
1563
1564 ;PARAM WORD EQUATES
1565
1566 000001 STATB= BIT0 ;OPERATOR AWAKE ASKED FOR
1567 000002 DATCKB= BIT1 ;DATA CHECK BIT
1568 000004 ECHOB= BIT2 ;ECHO BIT
1569 000010 MOCHK= BIT3 ;MODEM STATUS CHECK BIT
1570 000020 CRCB= BIT4 ;CRC CALCUALTE ASKED FOR
1571 000040 PROTOB= BIT5 ;PROTOCOL PROCESSING ASKED FOR
1572
1573 ;OPTION TYPE EQUATES
1574
1575
1576 000000 DPV= 0 ;CODE FOR DPV CHAR MODE
1577
1578 ;EVENT LOG MESSAGE TYPES (USED TO LOCATE EVENT DESCRIPTION IN EVENT TABLE
1579 ; AND DISPATCHING TO SEPERATE SECTIONS OF THE EVENT REPORTING SECTION)
1580 000000 TXQ= 0 ;TRANSMIT MESSAGE QUEUED
1581 000002 TXC= 2 ;TRANSMIT COMPLETE
1582 000004 RXQ= 4 ;RECEIVE BUFFER QUEUED
1583 000006 RXC= 6 ;RECEIVE COMPLETE
1584 000010 DER= 10 ;DEVICE INFORMATION
1585 000012 DVI= 12 ;DEVICE ABOUT TO INIT
1586 000014 DCK= 14 ;DATA COMPARISON RESULTS
1587
```



```
1588      000016      MSC= 16      ;MODEM STATUS CHANGE
1589
1590      000020      DLE= 20      ;DATA COMPARISON LENGH ERROR
1591      000022      DDE= 22      ;DATA COMPARISON DATA ERROR
1592      000024      FOP= 24      ;END OF PASS
1593
1594      ;EQUATES FOR FLAG WORD
1595
1596      000001      ININT= BIT0      ;INPUT INT. REC.
1597      000002      OTINT= BIT1      ;OUTPUT INT REC
1598      000004      QRX= BIT2      ;RX QUED /COMPL
1599      000010      QTX= BIT3      ;TX QUED/COMPL
1600      000100      ERX= BIT6      ;EXPECT TO GET A RX COMPLETED
1601      000200      ETX= BIT7      ;EXPECT TO GET A TX COMPLETED
1602
1603
1604      000020      TXM= BIT4      ;INDICATES TO TX INTERRUPT ROUTINE
1605      ;THAT IT IS TIME TO TRANSMIT BODY OF MSG.
1606      000040      RXM= BIT5      ;INDICATES TO RX INTERUPPT ROUTINE
1607      ;THAT IT IS TIME TO REC MSG BODY
1608      000400      BCC= BIT8      ;TIME FOR CRC CHECK.
1609
1610      001000      PAD= BIT9      ;INDICATES THAT PAD MUST BE SENT
1611
1612      002000      INOVR= BIT10      ;INIT OVER
1613
1614      004000      FIRST= BIT11      ;FIRST TIME FOR CTS
1615
1616      ; SPECIAL CLI CODES FOR "CHAR" ARGUMENT IN CLI CALLS
1617      ; (COMMAND LINE INTERPRETER DEFINITIONS)
1618      000000      CLIERR= 0
1619      000001      CLIEXI= 1
1620      000002      CLIBR= 2
1621      000003      CLIBIF= 3
1622      000004      CLISPA= 4
1623      000005      CLINUM= 5
1624      000006      CLIALP= 6
1625      000007      CLIALN= 7
1626      000010      CLIOCT= 8.
1627      000011      CLIDEC= 9.
1628      000012      CLISTR= 10.
1629
1630      ; DEFS FOR COMMAND LINE INTERPRETATION ACTION VALUES
1631      000000      NULL=0
1632      000001      CLEAR=1
1633      000002      SHOW=2
1634      000003      CHECK=3
1635      000004      RUN=4
1636      000005      HLP=5
1637      000006      CSHEXP=6
1638      000007      CSHTRN=7
1639      000010      SETEXP=10
1640      000011      SETTRN=11
1641      000012      SIZE=12
1642      000013      QCOPY=13
1643      000014      NUM=14
```

1644	000015	OPRMSG=15
1645	000016	STATUS=16
1646	000017	ENDQO=17
1647	000020	CMSG0=20
1648	000021	CMSG1=21
1649	000022	CMSG2=22
1650	000023	CMSG3=23
1651	000024	CMSG4=24
1652	000025	CMSG5=25
1653	000026	CMSG6=26
1654	000027	ATVMOD=27
1655	000030	PASMOD=30
1656	000031	RECMOD=31
1657	000032	LISMOD=32
1658	000033	DLLMOD=33
1659	000034	TRAMOD=34
1660	000035	TALMOD=35
1661	000036	NO=36
1662	000037	ECHO=37
1663	000040	CRC=40
1664	000041	PROTO=41
1665	000042	PASC=42
1666	000043	MOP=43
1667	000044	TTLLOP=44
1668	000045	CBLLOP=45
1669	000046	LMDLOP=46
1670	000047	RMDLOP=47
1671	000050	NOTNUF=50
1672	000051	BADCHR=51
1673	000052	DMPS=52
1674	000053	DMPE=53
1675	000054	DMPQ=54
1676	000055	PRNT=55
1677	000056	MOSC=56

:***** DEVICE DEPENDENT EQUATES

: MODEM SIGNAL BIT DEFINITIONS

: IF SIGNAL AVAILABLE IN DEVICE, EQUATE NAME TO BIT POSITION,
: ELSE EQUATE IT TO = 0

1684			
1685	020000	CTS= BIT13	:CLEAR TO SEND (CIRCUIT CB)
1686	001000	DSR= BIT9	:DATA SET READY (CIRCUIT CC)
1687	010000	DCD= BIT12	:DATA CARRIER DETECT (CIRCUIT CF)
1688	000004	RTS= BIT2	:REQUEST TO SEND (CIRCUIT CA)
1689	040000	RI= BIT14	:RING INDICATOR (CIRCUIT CE)
1690	000040	SQD= BIT5	:SIGNAL QUALITY DETECT (CIRCUIT CG)
1691	000040	TM= BIT5	:MODEM IN TEST MODE (RS 449 ONLY CIRCUIT TM)

: DEVICE SIGNALS

1695			
1696	000002	DTR= BIT1	:DATA TERMINAL READY
1697	000020	RXENA= BIT4	:RECEIVER ENABLE
1698	000040	DSITEN= BIT5	:DATA SET CHANGE ENABLE
1699	000100	RINTEN= BIT6	:REC INT. ENABLE

1700 000200
1701 002000
1702 004000
1703 000001
1704 000002
1705 000004
1706 000010
1707 000020
1708 000100
1709 000400
1710 001000
1711 100000
1712 100000
1713 000226
1714

RDATRY= BIT7
RSTARY= BIT10
RXACT= BIT11
RESET= BIT0
TXACT= BIT1
TBMT= BIT2
TTLL= BIT3
TXENA= BIT4
TINTEN= BIT6
TSOM= BIT8
TEOM= BIT9
TERR= BIT15
RERR= BIT15
SYN= 226

:REC DATA READY
:REC STATUS READY
:REC ACTIVE
:MASTER RESET
:TX ACTIVE
:TX BUFFER EMPTY
:TTL LOOP BIT
:TX ENABLE
:TX INT ENABLE
:TX START OF MSG.
:TX END OF MSG.
:TX ERROR
:REC OVER RUN
:SYNC WORD

1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770

002150
002150 000001
002152 000001
002154 000001
002156 000001
002160 000100
002162 000072
002164 000101
002166 000000
002170 000001

002172
002172 002214
002174 002215
002176 002216
002200 002217
002202 002220
002204 002320
002206 002412
002210 002520
002212 002642

002214 000
002215 377
002216 252
002217 125
002220 177603 157427 031011
002226 047321 163715 105221
002234 143325 142304
002240 040041 014116 052606
002246 172334 105025 123754
002254 111337 111523
002260 030030 145064 137642
002266 143531 063617 135075
002274 066730 026575
002300 052012 053627 070071
002306 151172 165044 031605
002314 166632 016741
002320

.SBTTL GLOBAL DATA SECTION
.SBTTL DEFAULT MESSAGE DEFINITIONS AND TABLES

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

:MESSAGE BYTE COUNT TABLE

DMSGCT:
MSG0C: .WORD EMSG0-MSG0 ;BYTE COUNT OF MESSAGE #0
MSG1C: .WORD EMSG1-MSG1 ;BYTE COUNT OF MESSAGE #1
MSG2C: .WORD EMSG2-MSG2 ;BYTE COUNT OF MESSAGE #2
MSG3C: .WORD EMSG3-MSG3 ;BYTE COUNT OF MESSAGE #3
MSG4C: .WORD EMSG4-MSG4 ;BYTE COUNT OF MESSAGE #4
MSG5C: .WORD EMSG5-MSG5 ;BYTE COUNT OF MESSAGE #5
MSG6C: .WORD EMSG6-MSG6 ;BYTE COUNT OF MESSAGE #6
OPCNT: .WORD 0 ;BYTE COUNT FOR OPERATOR SPEC'D MSG.
MSG8C: .WORD EMSG8-MSG8 ;BYTE COUNT OF RECEIVE BUFFER FILL PATTERN

:MESSAGE ADDRESS TABLE

DMSGAD:
MSG0 ;ADDRESS OF MESSAGE #0
MSG1 ;ADDRESS OF MESSAGE #1
MSG2 ;ADDRESS OF MESSAGE #2
MSG3 ;ADDRESS OF MESSAGE #3
MSG4 ;ADDRESS OF MESSAGE #4
MSG5 ;ADDRESS OF MESSAGE #5
MSG6 ;ADDRESS OF MESSAGE #6
OPBUF ;ADDRESS OF OPERATOR SPEC'D MSG.
MSG8 ;ADDRESS OF RECEIVE BUFFER FILL PATTERN

MSG0: .BYTE 000 ;MESSAGE OF ALL 0'S
EMSG0:
MSG1: .BYTE 377 ;MESSAGE OF ALL 1'S
EMSG1:
MSG2: .BYTE 252 ;MESSAGE OF ALTERNATING 1'S
EMSG2:
MSG3: .BYTE 125 ;MESSAGE OF ALTERNATING 0'S
EMSG3:
MSG4: ;"CCITT" 512-BIT (VS. 511 BITS) TEST PATTERN
.WORD 177603,157427,031011,047321,163715,105221,143325,142304

.WORD 040041,014116,052606,172334,105025,123754,111337,111523

.WORD 030030,145064,137642,143531,063617,135075,066730,026575

.WORD 052012,053627,070071,151172,165044,031605,166632,016741

EMSG4:

1771 002320
1772
1773 002320 077577 040444 052040
1774 002326 042510 050440 044525
1775 002334 045503 041040 047522
1776 002342 047127 043040 054117
1777 002350 045040 046525 042520
1778 002356 020104 053117 051105
1779 002364 052040 042510 046040
1780 002372 055101 020131 047504
1781 002400 027107
1782 002402 005015 077401 077577
1783 002410 000177
1784 002412
1785 002412
1786 002412 022043 021041 023040
1787 002420 024047 025051 026053
1788 002426 027055 030460 031462
1789 002434 032464 033466 034470
1790 002442 035472 036474 037476
1791 002450 040500 041502 042504
1792 002456 043506 044510 045512
1793 002464 046514 047516 050520
1794 002472 051522 052524 053526
1795 002500 054530 132
1796 002503 057 056133 057135
1797 002510 022537 000
1798 002513
1799 002514
1800
1801
1802
1803
1804 002514 047045 040445
1805 002520 000122
1806 002642
1807
1808
1809
1810
1811 002642 033
1812 002643
1813 002644

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

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

MSG8: .BYTE 33 ;RECEIVE BUFFER FILL PATTERN
MSG6: .EVEN
MSG6: ?/[\] ^ _ % ?
MSG6: .EVEN
; *****
; THESE THREE STORAGE AREAS MUST NOT BE SEPERATED !!!!
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
MSG6: .EVEN

1814
1815
1816
1817
1818 002644 000
1819 002645 201
1820 002646 000000
1821 002650 001
1822 002651 001
1823 002652 001
1824 002653
1825 002654 002654
1826 002654 000006
1827
1828 002656 000
1829 002657 201
1830 002660 000000
1831 002662 001
1832 002663 001
1833 002664 001
1834 002666
1835

```
.....  
: THE FOLLOWING IS THE AREA USED TO TRANSMIT AND REC THE :  
: HEADER MSGS. AND THE START, STACK ACK SEQUENCES. :  
.....  
HDMMSG: .BYTE 0 ; FILLER  
          .BYTE 201 ; SOH CHAR  
HDMCC: .WORD 0 ; CHAR COUNT GOES HERE  
        .BYTE 1 ; RESPONSE NUMBER  
        .BYTE 1 ; MSG. NUMBER  
        .BYTE 1 ; ADDR TO.  
HSMSE: .EVEN  
HDMC: .WORD 6  
RHDMMSG: .BYTE 0 ; SOH GOES IN HERE  
          .BYTE 201 ; BYTE COUNT GOES HERE  
RHDMCC: .WORD  
        .BYTE 1 ; RESP NUM  
        .BYTE 1 ; MSG NUM  
        .BYTE 1 ; ADDR TO.  
        .EVEN
```



```

1836
1837
1838 002666 000122
1839 003010 000000
1840
1841 003012 000000
1842 003014 000000
1843 003016 012077
1844 003020 012112
1845 003022 012220
1846 003024 012305
1847 003026 012364
1848 003030 012442
1849 003032 012532
1850 003034
1851 003034 012665 012674 012701
1852 003042 012706 012713 012721
1853 003050 012726 012734
1854
1855
1856
1857
1858 003054 000 377 252
1859 003057 125 203 177
1860 003062 043
1861 003063
1862 003064
1863
1864 003064 012745
1865 003066 012755
1866 003070 012766
1867 003072 012776
1868 003074 013005
1869 003076 013022
1870 003100 013027
1871
1872 003102 013036
1873 003104 013046
1874 003106 013057
1875 003110 013065
1876 003112 013100
1877
1878
1879
1880 003114 000000
1881 003116 000000
1882 003120 000000
1883 003122 000000
1884 003124 000000
1885 003126 000000
1886 003130 000
1887 003131 000
1888

;COMMAND LINE BUFFER, DATA LOCATIONS AND MESSAGES FOR ACTION ROUTINES
CMDBUF: .BLKB 82. ;BUFFER FOR OPERATOR COMMANDS
KEYWD1: .WORD 0 ;THIS LOC WILL =1 IF CLEAR TYPED, 2 FOR SHOW,
; A 4 IF RUN WAS TYPED, 5 IF HELP WAS TYPED
;THIS LOC HOLDS QUALIFIER VALUE (SIZE OR COPY)
QUALFG: .WORD 0
QUALVL: .WORD 0
HLPTAB: .WORD HLP1
;WORD HLP2
;WORD HLP3
;WORD HLP4
;WORD HLP4A
;WORD HLP5
;WORD HLP6
HLPEND:
SHTYTB: .WORD SHTYP0,SHTYP1,SHTYP2,SHTYP3,SHTYP4,SHTYP5,SHTYP6,SHTYP7

; THE LIST OF BYTES BELOW ARE THE FIRST BYTES OF THE PREDEFINED MESSAGES
; USED TO "SHOW" THE TRANSMIT AND COMPARE BUFFER CONTENTS.
SHTAB: .BYTE 0,377,252,125,203,177,043
SHTEND:
.EVEN
MODES: .WORD M00 ;ADDRESSES OF MODE TYPES IN ASCII
;WORD M01
;WORD M02
;WORD M03
;WORD M04
;WORD M05
;WORD M06
LOOPS: .WORD LP0 ;ADDRESSES OF LOOP TYPES IN ASCII
;WORD LP1
;WORD LP2
;WORD LP3
;WORD LP4

;COMMAND LINE TRAVERSE LOCATIONS (USED BY "P$TRV")
PSBUFA: .WORD 0 ;LOC. TO HOLD ADDR. OF CMD LINE BUFFER
PSTREE: .WORD 0 ;LOC. TO HOLD ADDR. OF PARSING TREE
PSACT: .WORD 0 ;LOC. TO HOLD ADDR. OF ACTION ROUTINE
PSCNT: .WORD 0 ;LOC. TO BE A COUNTER LOCATION
PSNUM: .WORD 0 ;LOC. TO HOLD NUMERIC VALUE FROM PARSE
PSRADX: .WORD 0 ;LOC. TO HOLD RADIX USED(LO) AND +/- (HI BYTE)
PSNNUF: .BYTE 0 ;RETURN =0 IF ENOUGH OF COMMAND FOUND
PSGDBD: .BYTE 0 ;RETURN CODE 0 IF NO ERROR FOUND

```

Line	Address	Value	Label	Description
1889			.SBTTL	MESSAGE BUFFERS AND POINTER TABLES
1890				
1891	003132	001000	TXBUF: .BLKB	BUFLIM ; TRANSMITTER BUFFERS
1892	004132	001000	RXBUF: .BLKB	BUFLIM ; RECEIVER BUFFERS
1893	005132	001000	CMPBUF: .BLKB	BUFLIM ; COMPARISON BUFFERS
1894	006132	000036	PTRTAB: .BLKW	MSGLIM*2 ; TABLE FOR MESSAGE ADDRS. & BYTE COUNTS
1895	006220	000036	PTR13: .BLKW	MSGLIM*2
1896	006322	000036	PTR23: .BLKW	MSGLIM*2
1897	006416		PTREND:	; END OF MSG. PTR. TABLE
1898				
1899	006416	000002	.BLKW 2	; FILLER FOR OVERFLOW OF RX POINTER TABLE
1900				
1901	006422	000000	RXPTR: .WORD 0	; RECEIVER MESSAGE POINTER
1902	006424	000000	TXPTR: .WORD 0	; TRANSMITTER BUFFER POINTER
1903	006426	000000	CMPPTR: .WORD 0	; COMPARISON BUFFER POINTER
1904	006430	000000	CMPTOT: .WORD 0	; CMP MSG TOTAL
1905	006432	000000	CTOTCC: .WORD 0	; COMPARE BUFFER CHAR. COUNT
1906	006434	000000	CCURAD: .WORD 0	; CURRENT ADDR OF CMP BUFF TO ADD AT
1907				
1908	006436	000000	DVTXA: .WORD 0	; DEVICE TX ADDR
1909	006440	000000	DVTCC: .WORD 0	; DEVICE TX CHAR COUNT
1910	006442	000000	DVTCT: .WORD 0	; DEVICE TX MESSAGE COUNT
1911	006444	000000	TXMTOT: .WORD 0	; TX MSG TOTAL
1912	006446	000000	TTOTCC: .WORD 0	; TX BUFFER CHAR. COUNT
1913	006450	000000	TCURAD: .WORD 0	; CURRENT ADDR. OF TX BUFF TO ADD AT
1914				
1915	006452	000000	DVRXA: .WORD 0	; DEVICE RX ADDR
1916	006454	000000	DVRCC: .WORD 0	; DEVICE RX CHAR COUNT
1917	006456	000000	DVRCT: .WORD 0	; DEVICE RX MESSAGE COUNT
1918	006460	000000	RXMTOT: .WORD 0	; RX MSG TOTAL
1919				
1920	006462	000000	LCNCT: .WORD 0	; NUMBER OF OPERATOR AWAKE MSGS
1921	006464	000000	OPVAR: .WORD 0	; OPTIONAL VARIABLE LOCATION
1922	006466	000000	PSCNT: .WORD 0	; PASS COUNTER
1923	006470	000000	ERRCNT: .WORD 0	; ERROR COUNTER
1924	006472	000000	STADD: .WORD 0	; START ADDR.
1925	006474	000000	ENADD: .WORD 0	; END ADDR. FOR DUMP
1926	006476	000000	BYTBIT: .WORD 0	; BYTE BIT FOR DUMP ROUTINE
1927				
1928			; OTHER MESSAGE RELATED STORAGE LOCATIONS	
1929				
1930	006500	000000	MSGTYP: .WORD 0	; TYPE OF DATA 0=0'S, 1=1'S, 2=10'S, 3=01'S ; 4=CCITT, 5=QUICK FOX, 6=ALPHA/NUM, 7=OPER
1931				
1932	006502	000000	CURCC: .WORD 0	; TX/RX/CMP CHAR COUNT
1933	006504	000000	CPTRR: .WORD 0	; CURRENT RX POINTER
1934	006506	000000	CPTR: .WORD 0	; CURRENT POINTER
1935	006510	000000	CURADD: .WORD 0	; CURRENT TX/RX/CMP START ADDD
1936	006512	000000	TOTCC: .WORD 0	; TOTAL CHAR COUNT NOT MORE THEN 'BUFLIM'
1937	006514	000000	OFFSET: .WORD 0	; OFFSET COUNT
1938	006516	000000	TEMP: .WORD 0	; TEMPORARY LOCATIONS (USED A LOT)
1939	006520	000000	TEMP1: .WORD 0	
1940	006522	000000	TEMP2: .WORD 0	
1941	006524	000000	TEMP3: .WORD 0	
1942	006526	000000	TEMP4: .WORD 0	
1943	006530	000000	TEMP5: .WORD 0	
1944	006532	000000	CONOTM: .WORD 0	; CONTROL OUT ERROR MSG. ADDRESS

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 45
MESSAGE BUFFERS AND POINTER TABLES

SEQ 0044

1945 006534 000000
1946 006536 000
1947 006537 000
1948

CONTIN: .WORD 0 ;WORD FOR CONTROL IN
GOOD: .BYTE 0 ;BYTE TO HOLD EXPECTED MESSAGE DATA BYTE FOR ERR REPORT
BAD: .BYTE 0 ;BYTE TO HOLD RECEIVED MESSAGE DATA BYTE FOR ERR REPORT

1949
1950
1951 006540 000000
1952 006542 000000
1953 006544 000000
1954 006546 000000
1955
1956
1957 006550 000000
1958
1959
1960 006552 000000
1961 006554 000002
1962
1963
1964
1965
1966
1967
1968
1969 006556 000000
1970 006560 000000
1971
1972
1973 006562 027250
1974 006564 027302
1975 006566 027342
1976 006570 027376
1977 006572 030556
1978 006574 030602
1979 006576 031022
1980
1981
1982
1983 006600 000000
1984 006602 000000
1985 006604 000000
1986 006606 000074
1987 006610 000000
1988
1989 006612 000000
1990 006614 000000
1991 006616 000000
1992
1993 006620 000000
1994 006622 000000
1995 006624 000000
1996

;MORE INDEPENDENT CODE STORAGE LOCATIONS

LOGUNT: .WORD 0 ;LOC. TO HOLD LOGICAL UNIT NUMBER
PCADD: .WORD 0 ;LOC. HOLD PC OF CALLING ROUTINE
RESFLG: .WORD 0 ;LOC TO HOLD FLAG (-1) THAT A RESTART WAS GIVEN
MODTYP: .WORD 0 ;DCLT MODE OF OPERATION TYPE
; (0=REC-ONLY, 1=TX-ONLY, 2=PASSIVE-LOOPBK,
; 3=ACTIVE-LOOPBK, 4=DOWN L.L., 5=TALK, 6=LISTEN)
MLTYP: .WORD 0 ;MAINTENANCE LOOP TYPE (0=NONE, 1=INTERNAL TTL,
; 2=CABLE, 3=MODEM-ANALOG LOOPBK (LOCAL),
; 4=MODEM-DIGITAL LOOPBK (REMOTE), 5=MOP)
F4DPLX: .WORD 0 ;FULL OR HALF DUPLEX FLAG (1=FULL FROM P-TABLE)
PARAM: .WORD 2 ;PROGRAM PARAMETERS
; BIT0= STATUS MSGS TO OPR PRINTED (1=YES)
; BIT1= DATA CHECKING DONE ON RCVD MSGS (1=YES)
; BIT2= ECHO (TRANSMIT) RCV'D MSG.(PASSIVE)(1=YES)
; BIT3= MODEM STATUS CHECK (1=YES)
; BIT4= CRC CALC./CHECK DONE (1=YES)
; BIT5= PROTOCOL EMULATION (1=YES)
; BIT6= SPARE
RPASS: .WORD 0 ;PASS NUMBER FROM RUN COMMAND
FLAG: .WORD 0 ;DEVICE FLAG WORD

;MODE DISPATCH TABLE

MODE: .WORD RXONLY ;RX ONLY DISPATCH
.WORD TXONLY ;TX ONLY DISPATCH
.WORD PLCK ;PASSIVE LOOP BACK DISP
.WORD ALCK ;ACTIVE LOOP BACK DISP
.WORD DLL ;DOWN LINE LOAD DISP
.WORD TALCK ;TALK MODE DISPATCH
.WORD LISCK ;LISTEN MODE DISPATCH

.SBTTL CLOCK TABLES, EVENT LOG AND POINTERS

CLKCSR: .WORD 0 ;CLOCK CSR ADDRESS
CLKBR: .WORD 0 ;CLOCK INTERRUPT LEVEL
CLKVEC: .WORD 0 ;CLOCK INTERRUPT VECTOR
CLKHZ: .WORD 60. ;CLOCK'S HERTZ RATE
CLKEN: .WORD 0 ;CLOCK'S CSR VALUE TO INTRPT. ENABLE IT
TIMMIN: .WORD 0 ;PLACE TO KEEP TIME-SINCE-START
TIMSEC: .WORD 0
TIMTCK: .WORD 0 ;PLACE TO KEEP # OF TICKS/SEC
TIMER1: .WORD 0 ;EVENT TIMER #1 (TICKS)
TIMER2: .WORD 0 ;EVENT TIMER #2 (TICKS)
TIMERS: .WORD 0 ;EVENT TIMER #3 (SECONDS)

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 47
CLOCK TABLES, EVENT LOG AND POINTERS

SEQ 0046

1997
1998 006626 006630
1999 006630 000341
2000 007532 000001
2001
2002
2003
2004 007534 000000
2005
2006

;EVENT LOG TABLE AND ITS NEXT ENTRY POINTER
EVTPTN: .WORD EVTLOG ;POINTER TO NEXT FREE SPACE IN EVENT LOG
EVTLOG: .BLKW 225. ;EVENT LOG BUFFER
EVTEND: .BLKW 1. ;APPROXIMATE END OF EVENT TABLE (ALLOWS CIRCULAR QUE)

.SBTTL MODEM DATA SECTION
MODS: .WORD 0 ;MODEM STATUS

2007
2008
2009
2010 007536 020000
2011 007540 001000
2012 007542 010000
2013 007544 000004
2014 007546 040000
2015 007550 000040
2016 007552 000040
2017 007554
2018
2019
2020
2021 007554 015564
2022 007556 015570
2023 007560 015574
2024 007562 015600
2025 007564 015604
2026 007566 015610
2027 007570 015614
2028
2029
2030
2031
2032 007572 014202
2033 007574 014226
2034 007576 014255
2035 007600 014302
2036 007602 014330
2037 007604 014375
2038 007606 014345
2039 007610 014527
2040 007612 014423
2041 007614 014460
2042 007616 014513
2043
2044
2045
2046 007620 000000
2047 007622 000000
2048 007624 000000
2049 007626 000000
2050 007630 000000
2051 007632 000000
2052
2053
2054
2055 007634 020116
2056 007636 020116
2057 007640 020116
2058 007642 020116
2059 007644 020170
2060 007646 020264
2061 007650 020460
2062 007652 020534

;TABLE OF MODEM SIGNAL BIT DEFINITIONS

MOBITS: .WORD CTS ;CLEAR TO SEND (CIRCUIT CB)
.WORD DSR ;DATA SET READY (CIRCUIT CC)
.WORD DCD ;DATA CARRIER DETECT (CIRCUIT CF)
.WORD RTS ;REQUEST TO SEND (CIRCUIT CA)
.WORD RI ;RING INDICATOR (CIRCUIT CE)
.WORD SQD ;SIGNAL QUALITY DETECT (CIRCUIT CG)
.WORD TM ;MODEM IN TEST MODE (RS 449 ONLY CIRCUIT TM)

MOBITE:

;TABLE OF ADDRESSES OF MODEM SIGNAL MESSAGE POSITIONS

MOMSGS: .WORD EVMCTS ;CLEAR TO SEND (CIRCUIT CB)
.WORD EVMDSR ;DATA SET READY (CIRCUIT CC)
.WORD EVMDCD ;DATA CARRIER DETECT (CIRCUIT CF)
.WORD EVMRTS ;REQUEST TO SEND (CIRCUIT CA)
.WORD EVMRI ;RING INDICATOR (CIRCUIT CE)
.WORD EVMSQD ;SIGNAL QUALITY DETECT (CIRCUIT CG)
.WORD EVMTM ;MODEM IN TEST MODE (RS 449 ONLY CIRCUIT TM)

;TABLE OF ADDRESSES OF EVENT DESCRIPTION MESSAGES
; ORDER CORRESPONDS TO MESSAGE TYPE VALUES

EVTLST: .WORD EDTXQ ;TRANSMIT MESSAGE QUEUED
.WORD EDTXC ;TRANSMIT OF MESSAGE COMPLETE
.WORD EDRXQ ;RECEIVE MESSAGE SPACE QUEUED
.WORD EDRXC ;MESSAGE RECEIVED - RECEIVE COMPLETE
.WORD EDDER ;DEVICE INFORMATION
.WORD EDDVI ;DEVICE INITIALIZE STARTED
.WORD EDDCK ;DATA COMPARISON DONE
.WORD EDMOS ;MODEM STATUS CHANGE
.WORD EDDLE ;DATA COMPARE LENGTH ERROR
.WORD EDDDE ;DATA COMPARE DATA ERROR
.WORD EDEOP ;END OF PASS

;LOCATIONS USED DURING EVENT REPORTING

EVTSEC: .WORD 0 ;TEMPORARY LOCS TO KEEP EVENT TIME WHILE REPORTING
EVTMIN: .WORD 0
EVTICK: .WORD 0
EVTADD: .WORD 0 ;TEMP. LOC. TO HOLD ADDRESS DURING EVENT REPORTING
EVTBCT: .WORD 0 ;" " " " BYTE COUNT " " " "
EVTTMP: .WORD 0 ;" " " " OTHER DATA " " "

;REPORT CODING DISPATCH TABLE

RPTDSP: .WORD RPTTXQ ;TRANSMIT QUEUED ENTRY DECODING
.WORD RPTXQ ;TRANSMIT COMPLETE ENTRY DECODING
.WORD RPTTXQ ;RECEIVER QUEUED ENTRY DECODING
.WORD RPTXQ ;RECEIVER COMPLETE ENTRY DECODING
.WORD RPTDER ;DEVICE ERROR ENTRY DECODING
.WORD RPTDVI ;DEVICE INIT ENTRY DECODING
.WORD RPTDCK ;DATA COMPARISON ENTRY DECODING
.WORD RPTMSC ;REPORT MODEM STATUS CHANGE

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{J 4} 10:29 PAGE 49
MODEM DATA SECTION

SEQ 0048

2063 007654 020460
2064 007656 020404
2065 007660 020330
2066
2067
2068 007662 000000
2069 007664 000000
2070 007666 000000
2071 007670 000000
2072

.WORD RPTDLE ;DATA COMPARISON LENGH ERROR
.WORD RPTDDE ;DATA COMPARISON DATA ERROR
.WORD RPTEOP ;END OF PASS

DEV1: .WORD 0 ;TEMP LOCS TO HOLD DATA FOR EVENT REPORTING
DEV2: .WORD 0 ; AND SHOW MODE,... SUBROUTINE
DEV3: .WORD 0
DEV4: .WORD 0

2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088 007572
2089
2090
2091 007672
2092 007676
2093 007702
2094 007704
2095 007720
2096 007722
2097 007736
2098 007740
2099 007752
2100 007756
2101 007772
2102 007776
2103 010012
2104 010016
2105 010022
2106 010034
2107 010040
2108 010052
2109 010056
2110
2111
2112
2113 010060
2114 010064
2115 010100
2116 010104
2117 010122
2118 010126
2119 010144
2120 010150
2121 010166
2122 010172
2123 010210
2124 010214
2125 010240
2126 010244
2127 010250
2128 010266

.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  
-----  
: ! ASCIIZ MATCH STRING ! ONLY IF 'ASCII' ARGUMENT DEFINED  
: ! (.EVEN) !  
-----
```

CLITRE:

:FIRST KEYWORD

```
N10$: CLI CLISPA,0,N10$ :SKIP ANY LEADING SPACES  
CLI <'?'>,HLP,N42$ :IS THE FIRST NON-SP CHAR A '?'  
CLI CLIEXI,0 : IF YES DO 'HLP' AND EXIT  
N42$: CLI CLISTR,HLP,N43$,<'HELP'> :ELSE, IS FIRST WORD A 'HELP'  
CLI CLIEXI,0 : IF YES DO 'HLP' AND EXIT  
N43$: CLI CLISTR,PRNT,N45$,<'PRINT'> :ELSE, IS FIRST WORD A 'PRINT'  
CLI CLIEXI,0 : IF YES DO 'PRINT' AND EXIT  
N45$: CLI CLISTR,RUN,N46$,<'RUN'> :ELSE, IS FIRST WORD A 'RUN'  
CLI CLIBR,0,N80$ : IF YES DO 'RUN' & GOTO N80$  
N46$: CLI CLISTR,NOTNUF,N40$,<'DUMP'> :ELSE, IS FIRST WORD A 'DUMP'  
CLI CLIBR,0,N50$ : IF YES GOTO N80$  
N40$: CLI CLISTR,CLEAR,N20$,<'CLEAR'> :ELSE, IS FIRST WORD A 'CLEAR'  
CLI CLIBR,NOTNUF,N100$ : IF YES DO 'CLR' & GOTO N100$  
N20$: CLI <'S'>,NOTNUF,N30$ :ELSE, IS FIRST CHAR. A 'S'  
CLI CLISTR,SHOW,N25$,<'HOW'> : IF YES IS REST OF WORD 'HOW'  
CLI CLIBR,0,N100$ : IF YES, DO 'SHOW',BR N100$  
N25$: CLI CLISTR,0,N30$,<'ET'> : ELSE, IS REST OF WORD 'ET'  
CLI CLIBR,0,N110$ : IF YES, DO 'SET', BR N110$  
N30$: CLI CLIERR,0 :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='  
CLI <'='>,0,N30$ : IF NO, IT'S WRONG -ERR -EXIT  
CLI CLISTR,ATVMOD,N82$,<'ACTIVE'> :IS NEXT WORD 'ACTIVE'  
CLI CLIBR,0,N115$ : IF YES, DO 'ACTIVE',BR N115$  
N82$: CLI CLISTR,PASMOD,N83$,<'PASSIVE'> :IS NEXT WORD 'PASSIVE'  
CLI CLIBR,0,N115$ : IF YES, DO 'PASSIVE',BR N115$  
N83$: CLI CLISTR,RECMOD,N84$,<'RECEIVE'> :IS NEXT WORD 'RECEIVE'  
CLI CLIBR,0,N115$ : IF YES, DO 'RECVE',BR N115$  
N84$: CLI CLISTR,LISMOD,N85$,<'LISTEN'> :IS NEXT WORD 'LISTEN'  
CLI CLIBR,0,N115$ : IF YES, DO 'LISTEN',BR N115$  
N85$: CLI CLISTR,DLLOD,N86$,<'DOWNLINELOAD'> :IS NEXT WORD 'DOW'  
CLI CLIBR,0,N115$ : IF YES, DO 'DWNLL',BR N115$  
N86$: CLI <'T'>,0,N30$ :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$
```

2129 010272
2130 010304
2131
2132
2133
2134 010310
2135 010314
2136 010336
2137 010340
2138 010364
2139
2140
2141
2142
2143 010366
2144 010372
2145 010412
2146 010416
2147 010440
2148
2149
2150 010444
2151 010450
2152 010454
2153 010460
2154 010464
2155 010470
2156 010474
2157 010500
2158
2159
2160 010504
2161 010510
2162 010514
2163 010526
2164 010532
2165 010546
2166
2167
2168 010552
2169 010570
2170 010574
2171 010610
2172
2173
2174
2175 010614
2176 010630
2177
2178
2179 010634
2180 010650
2181
2182 010654
2183 010670
2184

```
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$
```


2185
2186 010674
2187
2188
2189 010700
2190 010714
2191 010720
2192 010736
2193 010742
2194 010756
2195 010762
2196 010776
2197 011002
2198 011016
2199 011022
2200 011036
2201 011042
2202 011056
2203
2204
2205 011062
2206 011066
2207 011072
2208 011076
2209 011102
2210 011106
2211 011112
2212
2213
2214 011114
2215 011120
2216 011124
2217 011140
2218 011144
2219 011160
2220
2221
2222 011164
2223 011170
2224 011174
2225
2226
2227 011200
2228
2229
2230 011204
2231 011226
2232 011232
2233 011246
2234 011252
2235 011274
2236 011300
2237 011322
2238
2239
2240 011326

;GET MESSAGE TYPE FOR SET MESSAGE COMMANDS
N120\$: CLI <'=>,0,N30\$

; LOOK FOR DEFAULT MESSAGE NAME
N60\$: CLI CLISTR,CMMSG1,N61\$,<'ONES'>
CLI CLIBR,0,N121\$
N61\$: CLI CLISTR,CMMSG0,N62\$,<'ZEROES'>
CLI CLIBR,0,N121\$
N62\$: CLI CLISTR,CMMSG2,N63\$,<'1ALT'>
CLI CLIBR,0,N121\$
N63\$: CLI CLISTR,CMMSG3,N64\$,<'0ALT'>
CLI CLIBR,0,N121\$
N64\$: CLI CLISTR,CMMSG5,N65\$,<'ITEP'>
CLI CLIBR,0,N121\$
N65\$: CLI CLISTR,CMMSG4,N66\$,<'CCITT'>
CLI CLIBR,0,N121\$
N66\$: CLI CLISTR,CMMSG6,N67\$,<'ALPHA'>
CLI CLIBR,0,N121\$

; LOOK FOR QUOTED MESSAGE
N67\$: CLI <'>,OPRMSG,N30\$
N70\$: CLI <'>,ENDQO,N71\$
CLI CLIBR,0,N121\$
N71\$: CLI CLISPA,0,N72\$
N72\$: CLI CLIALN,0,N73\$;ONLY A-Z,SP,TAB, OR 0-9 BETWEEN ''S
CLI CLIBR,0,N70\$;PRINT ERROR IF NONE LEGAL CHAR FOR ''S
N73\$: CLI CLIERR,BADCHR

;GET QUALIFIERS (SIZE OR COPY) FOR SET MESSAGE COMMANDS
N121\$: CLI CLIALP,0,N123\$
N123\$: CLI <'>,NOTNUF,N125\$
CLI CLISTR,SIZE,N122\$,<'SIZE'>
CLI CLIBR,0,N126\$
N122\$: CLI CLISTR,QCOPY,N30\$,<'COPY'>
CLI CLIBR,0,N126\$

;NUMER FOR SIZE OR COPY
N126\$: CLI <'=>,0,N30\$
CLI CLIDEC,NUM,N30\$
CLI CLIBR,0,N121\$

;GET MAINTENANCE LOOP TYPE FOR RUN 'LOOP' QUALIFIER
N140\$: CLI <'=>,0,N30\$

N141\$: CLI CLISTR,TTLLOP,N142\$,<'INTERNAL TTL'>
CLI CLIBR,0,N115\$
N142\$: CLI CLISTR,CBLLOP,N143\$,<'CABLE'>
CLI CLIBR,0,N115\$
N143\$: CLI CLISTR,LMDLOP,N144\$,<'LOCAL MODEM'>
CLI CLIBR,0,N115\$
N144\$: CLI CLISTR,RMDLOP,N30\$,<'REMOTE MODEM'>
CLI CLIBR,0,N115\$

;GET LINE NUMBER FOR 'PASS' RUN QUALIFIER
N150\$: CLI <'=>,0,N30\$

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{N 4} 10:29 PAGE 53
COMMAND LINE ACTION TREE

SEQ 0052

2241 011332
2242 011336
2243
2244
2245
2246
2247 011342
2248

CLI CLIDEC,PASC,N30\$
CLI CLIBR,0,N115\$

:END-OF-LINE
N125\$: CLI CLIEXI,0

2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292

;DEVICE DEPENDENT STORAGE LOCATIONS FOR
; CURRENT DEVICE PARAMTERS

RXCSR: .WORD 0 ;REC CONTROL AND STATUS
PCSAR: .WORD 0 ;STATUS REGISTIER
RDSR: .WORD 0 ;REC DATA AND STATUS REG
TXCSR: .WORD 0 ;TRANSMIT AND REC. CONTROL
TDSR: .WORD 0 ;TRANSMIT DATA AND STATUS REG

INVEC: .WORD 0 ;INPUT INTERRUPT VECTOR ADDRESS
OUTVEC: .WORD 0 ;OUTPUT INTERRUPT VECTOR ADDRESS
INTPRI: .WORD 0 ;INTERRUPT PRIORITY
OPTYP: .WORD 0 ;DEVICE OPTION TYPE(0=DMC,5=DMR-DMC MODE

DPVP1: .WORD 065626 ;THIS WORD IS BROKEN DOWN AS FOLLOWS
;BITS 0-7 =SYNC WORD
;BITS 8-10=ERR DET SELECTED
;BIT11 = IDLE
;BIT12 = SEC ADDR. MODE
;BIT13 = STRIP SYNC
;BIT14 = PORTO TYPE SEL(1=BCP 0=BOP)
;BIT15 = ALL PARTIES ADDRESS..

;THIS IS SET TO SYNC 262
;CRC 16 INIT TO 1
;STRIP SYNC AND BCP MODE
;IDLE SET TO MARK

CMODS: .WORD 0 ;CURRENT MODEM
IRXCSR: .WORD 0 ;IMAGE OF RXCSR
IRDSR: .WORD 0 ;IMAGE OR RDSR
MSGPTR: .WORD 0 ;MSG PTR.FOR HEADER OR CONTROL
MSGCC: .WORD 0 ;MSG COUNTER OR CC
SYNCC: .WORD 0 ;SYNC CHAR COUNT.
SYNCW: .WORD 0 ;SYNC WORD.PLUS TSOM BIT.
RMSGPT: .WORD 0 ;MSG PTR FOR REC
RMSGCC: .WORD 0 ;CHAR COUNTER FOR REC
BCCW: .WORD 0 ;CRC HOLDING LOC.
MGLCNT: .WORD 0 ;COUNT OF GLITCH ERRORS
MHRCNT: .WORD 0 ;COUNT OF HARD ERRORS
RNODE: .WORD 0 ;1=REMOTE NODE ITEP,0=NON ITEP

; ERRRTBL

2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329

.SBTTL GLOBAL TEXT SECTION

:++
: THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
: MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
: MORE THAN ONE TEST.
:--

.SBTTL DEVICE SUPPORTED
: NAMES OF DEVICES SUPPORTED BY PROGRAM
:

DEV TYP <DPV-11>

L\$DVTYP::
.ASCIZ /DPV-11/
.EVEN

011422
011422
011422 050104 026526 030461
011430 000
011432

.SBTTL PROGRAM IDENTIFICATION
: TEST DESCRIPTION
:

DESCRIPT <DPV-11 DATA COMM LINK TEST >

L\$DESC::
.ASCIZ /DPV-11 DATA COM
.EVEN

011432
011432
011432 050104 026526 030461
011440 042040 052101 020101
011446 047503 046515 046040
011454 047111 020113 042524
011462 052123 000040

.EVEN

```
2330 .SBTTL GLOBAL FORMAT STATEMENTS, MESSAGES, AND ASCII INFO
2331
2332
2333 011466 041504 052114 000076 CLISPM: .ASCIZ /DCLT>/
2334 011474 047045 040445 044477 CLIERM: .ASCIZ /%N%?ILL CMD-BAD SYNTAX?/
2335 011502 046114 041440 042115
2336 011510 041055 042101 051440
2337 011516 047131 054124 000077
2338 011524 047045 040445 044477 CLINUF: .ASCIZ /%N%?INCMPLTE CMD?/
2339 011532 041516 050115 052114
2340 011540 020105 046503 037504
2341 011546 000
2342 011547 045 022516 037501 CLINBG: .ASCIZ /%N%?NUM TOO BIG?/
2343 011554 052516 020115 047524
2344 011562 020117 044502 037507
2345 011570 000
2346 011571 045 022516 037501 CLIBRX: .ASCIZ /%N%?BAD RADIX?/
2347 011576 040502 020104 040522
2348 011604 044504 037530 000
2349 011611 045 022516 037501 CLIBDL: .ASCIZ /%N%?'LOOP' VALID ONLY IN ACTIVE?/
2350 011616 046042 047517 021120
2351 011624 053040 046101 042111
2352 011632 047440 046116 020131
2353 011640 047111 040440 052103
2354 011646 053111 037505 000
2355 011653 045 022516 037501 CLINPS: .ASCIZ /%N%?'ECHO' VALID ONLY IN PASSIVE?/
2356 011660 042442 044103 021117
2357 011666 053040 046101 042111
2358 011674 047440 046116 020131
2359 011702 047111 050040 051501
2360 011710 044523 042526 000077
2361 011716 047045 040445 044477 CLIBCR: .ASCIZ /%N%?ILL CHR- 'A-Z,0-9,SP,TAB' ONLY?/
2362 011724 046114 041440 051110
2363 011732 020055 040442 055055
2364 011740 030054 034455 051454
2365 011746 026120 040524 021102
2366 011754 047440 046116 037531
2367 011762 000
2368 011763 045 022516 037501 CLISEO: .ASCIZ /%N%?'SIZE=0' NOT VALID?/
2369 011770 051442 055111 036505
2370 011776 021060 047040 052117
2371 012004 053040 046101 042111
2372 012012 000077
2373 012014 047045 040445 044124 HLPO: .ASCIZ /%N%?THIS IS DCLT. TYPE 'H' OR '?' FOR DETAILS/
2374 012022 051511 044440 020123
2375 012030 041504 052114 020056
2376 012036 054524 042520 021040
2377 012044 021110 047440 020122
2378 012052 037442 020042 047506
2379 012060 020122 042504 040524
2380 012066 046111 000123
2381 012072 047045 052045 000 HLPF: .ASCIZ /%N%?/
2382 012077 104 046103 020124 HLP1: .ASCIZ /DCLT CMDS:/
2383 012104 046503 051504 000072
2384 012112 041440 042514 051101 HLP2: .ASCII / CLEAR OR SHOW EXPECTLIST OR TRANSMITLIST/<15><12>
2385 012120 047440 020122 044123
```


2386 012126 053517 020040 054105
2387 012134 042520 052103 044514
2388 012142 052123 047440 020122
2389 012150 051124 047101 046523
2390 012156 052111 044514 052123
2391 012164 005015
2392 012166 050040 044522 052116
2393 012174 005015
2394 012176 042040 046525 020120
2395 012204 052123 051101 026524
2396 012212 047105 027504 000102
2397 012220 051440 052105 042440
2398 012226 050130 041505 046524
2399 012234 043523 047440 020122
2400 012242 051124 047101 046523
2401 012250 052111 051515 036507
2402 012256 054524 042520 051457
2403 012264 055111 036505 020116
2404 012272 051117 027440 047503
2405 012300 054520 047075 000
2406 012305 040 020040 054524
2407 012312 042520 047475 042516
2408 012320 026123 042532 047522
2409 012326 051505 030454 046101
2410 012334 026124 040460 052114
2411 012342 044454 042524 026120
2412 012350 041503 052111 026124
2413 012356 046101 044120 000101
2414 012364 020040 020040 020040
2415 012372 047440 020122 047442
2416 012400 051120 051440 041520
2417 012406 036504 026501 026132
2418 012414 050123 052054 041101
2419 012422 030054 034455 044440
2420 012430 020116 052521 052117
2421 012436 051505 000042
2422 012442 051040 047125 046440
2423 012450 042117 036505 052115
2424 012456 050131 046057 047517
2425 012464 036520 052114 050131
2426 012472 041457 042510 045503
2427 012500 051454 040524 052524
2428 012506 026123 041505 047510
2429 012514 046454 042117 046505
2430 012522 050054 051501 036523
2431 012530 000116
2432 012532 020040 046440 054524
2433 012540 036520 051124 047101
2434 012546 051054 041505 040454
2435 012554 052103 050054 051501
2436 012562 052054 046101 046054
2437 012570 051511 042054 053517
2438 012576 006516 012
2439 012601 040 020040 052114
2440 012606 050131 044475 052116
2441 012614 041454 041101 046054

.ASCII / PRINT/<15><12>

.ASCIZ ? DUMP START-END/B?

HLP3: .ASCIZ ? SET EXPECTMSG OR TRANSMITMSG=TYPE/SIZE=N OR /COPY=N?

HLP4: .ASCIZ ? TYPE=ONES,ZEROES,1ALT,0ALT,ITEP,CCITT,ALPHA?

HLP4A: .ASCIZ / OR 'OPR SPCD=A-Z,SP,TAB,0-9 IN QUOTES'/'

HLP5: .ASCIZ ? RUN MODE=MTYP/LOOP=LTYP/CHECK,STATUS,ECHO,MODEM,PASS=N?

HLP6: .ASCII / MTYP=TRAN,REC,ACT,PAS,TAL,LIS,DOWN/<15><12>

.ASCIZ / LTYP=INT,CAB,LOC,REM/

2442	012622	041517	051054	046505		
2443	012630	000				
2444						
2445	012631	045	022516	046501	SHMSG:	.ASCIZ ?%N%AMSG: TYPE=%T%A/SIZE=%D3?
2446	012636	043523	020072	054524		
2447	012644	042520	022475	022524		
2448	012652	027501	044523	042532		
2449	012660	022475	031504	000		
2450	012665	132	051105	042517	SHTYPO:	.ASCIZ /ZEROES/
2451	012672	000123				
2452	012674	047117	051505	000	SHTYP1:	.ASCIZ /ONES/
2453	012701	061	046101	000124	SHTYP2:	.ASCIZ /1ALT/
2454	012706	040460	052114	000	SHTYP3:	.ASCIZ /OALT/
2455	012713	103	044503	052124	SHTYP4:	.ASCIZ /CCITT/
2456	012720	000				
2457	012721	111	042524	000120	SHTYP5:	.ASCIZ /ITEP/
2458	012726	046101	044120	000101	SHTYP6:	.ASCIZ /ALPHA/
2459	012734	050117	020122	050123	SHTYP7:	.ASCIZ /OPR SPEC/
2460	012742	041505	000			
2461	012745	122	041505	044505	MO0:	.ASCIZ /RECEIVE/
2462	012752	042526	000			
2463	012755	124	040522	051516	MO1:	.ASCIZ /TRANSMIT/
2464	012762	044515	000124			
2465	012766	040520	051523	053111	MO2:	.ASCIZ /PASSIVE/
2466	012774	000105				
2467	012776	041501	044524	042526	MO3:	.ASCIZ /ACTIVE/
2468	013004	000				
2469	013005	104	053517	046116	MO4:	.ASCIZ /DOWNLINELOAD/
2470	013012	047111	046105	040517		
2471	013020	000104				
2472	013022	040524	045514	000	MO5:	.ASCIZ /TALK/
2473	013027	114	051511	042524	MO6:	.ASCIZ /LISTEN/
2474	013034	000116				
2475	013036	000			LP0:	.ASCIZ //
2476	013037	057	047514	050117	LP00:	.ASCIZ ?/LOOP=?
2477	013044	000075				
2478	013046	047111	042524	047122	LP1:	.ASCIZ ?INTERNAL?
2479	013054	046101	000			
2480	013057	103	041101	042514	LP2:	.ASCIZ ?CABLE?
2481	013064	000				
2482	013065	114	041517	046101	LP3:	.ASCIZ ?LOCALMODEM?
2483	013072	047515	042504	000115		
2484	013100	042522	047515	042524	LP4:	.ASCIZ ?REMOTEMODEM?
2485	013106	047515	042504	000115		
2486	013114	047516			PNST:	.ASCII /NO/
2487	013116	052123	052101	051525	PST:	.ASCIZ /STATUS/
2488	013124	000				
2489	013125	116	117		PNCK:	.ASCII /NO/
2490	013127	103	042510	045503	PCK:	.ASCIZ /CHECK/
2491	013134	000				
2492	013135	116	117		PNEC:	.ASCII /NO/
2493	013137	105	044103	000117	PEC:	.ASCIZ /ECHO/
2494	013144	047516			PNMS:	.ASCII /NO/
2495	013146	047515	042504	000115	PMS:	.ASCIZ /MODEM/
2496						
2497						

```
2498
2499 013154 047045 040445 044514 LISP: .ASCIZ /%N%ALIS>/
2500 013162 037123 000
2501 013165 124 045514 000076 OPRMM: .ASCIZ /TLK>/
2502 013172 044124 051511 040440 L5060: .ASCIZ /THIS A 50. OR 60. HZ. LSI-11:/
2503 013200 032440 027060 047440
2504 013206 020122 030066 020056
2505 013214 055110 020056 051514
2506 013222 026511 030461 000072
2507 .EVEN
2508
2509
2510
2511
2512 :
2513 : FORMAT STATEMENTS USED IN PRINT CALLS
2514 :
2515
2516 013230 047045 040445 047504 DLLCM: .ASCIZ /%N%ADOWN LINE LOAD NOT SUPPORTED BY THIS DEVICE/
2517 013236 047127 046040 047111
2518 013244 020105 047514 042101
2519 013252 047040 052117 051440
2520 013260 050125 047520 052122
2521 013266 042105 041040 020131
2522 013274 044124 051511 042040
2523 013302 053105 041511 000105
2524
2525
2526 013310 047045 040445 046103 BDCLK: .ASCIZ /%N%ACLOCK NOT FOUND/
2527 013316 041517 020113 047516
2528 013324 020124 047506 047125
2529 013332 000104
2530 013334 047045 040445 040502 NOCLK: .ASCIZ /%N%ABAD CLOCK - PROGRAM WILL HANG ON "TIMEOUT"!!!/
2531 013342 020104 046103 041517
2532 013350 020113 020055 051120
2533 013356 043517 040522 020115
2534 013364 044527 046114 044040
2535 013372 047101 020107 047117
2536 013400 021040 044524 042515
2537 013406 052517 021124 020441
2538 013414 000
2539 013415 115 054101 020056 TABEX: .ASCIZ /MAX. CHAR. MSG COUNT EXCEEDED -/
2540 013422 044103 051101 020056
2541 013430 051515 020107 047503
2542 013436 047125 020124 054105
2543 013444 042503 042105 042105
2544 013452 026440 000
2545 013455 102 043125 042506 BUFEX: .ASCIZ /BUFFER FULL -/
2546 013462 020122 052506 046114
2547 013470 026440 000
2548 013473 045 022516 022524 MSGTRN: .ASCIZ /%N%T% A MSG. NOT BUILT !!!/
2549 013500 020101 051515 027107
2550 013506 047040 052117 041040
2551 013514 044525 052114 020440
2552 013522 000041
2553 013524 047045 040445 044103 MSGTRU: .ASCIZ /%N%ACHAR. COUNT EXCEEDS BUFF LIMIT - MSG TRUNCATED/
```

2554 013532 051101 020056 047503
2555 013540 047125 020124 054105
2556 013546 042503 042105 020123
2557 013554 052502 043106 046040
2558 013562 046511 052111 026440
2559 013570 046440 043523 052040
2560 013576 052522 041516 052101
2561 013604 042105 000
2562 013607 045 022516 032523
2563 013614 040445 047515 042504
2564 013622 022475 022524 022524
2565 013630 022524 027501 040520
2566 013636 051523 022475 032532
2567 013644 000
2568
2569 013645 045 022516 032523
2570 013652 051445 022465 032523
2571 013660 040445 022457 022524
2572 013666 027501 052045 040445
2573 013674 022457 022524 027501
2574 013702 052045 000
2575
2576 013705 045 032523 040445
2577 013712 047524 040524 020114
2578 013720 044515 046523 052101
2579 013726 044103 051505 044440
2580 013734 020116 051515 020107
2581 013742 020075 042045 000065
2582 013750 047045 051445 022463
2583 013756 041501 046101 042514
2584 013764 020104 051106 046517
2585 013772 050040 036503 047445
2586 014000 000066
2587 014002 051445 022465 041501
2588 014010 046517 040520 042522
2589 014016 041440 052517 052116
2590 014024 022475 032504 051445
2591 014032 022463 051101 041505
2592 014040 044505 042526 041440
2593 014046 052517 052116 022475
2594 014054 032504 000
2595 014057 115 042117 046505
2596 014064 051440 040524 052524
2597 014072 020123 044103 047101
2598 014100 042507 020123 047506
2599 014106 020122 044124 051511
2600 014114 050040 051501 020123
2601 014122 042527 042522 027056
2602 014130 000
2603 014131 045 032523 040445
2604 014136 040510 042122 041440
2605 014144 040510 043516 051505
2606 014152 022475 032504 040445
2607 014160 051445 022463 043501
2608 014166 044514 041524 042510
2609 014174 036523 042045 000065

SHF0: .ASCIZ ?%N%S5%AMODE=%T%T%T%A/PASS=%Z5?

SHF1: .ASCIZ ?%N%S5%S5%S5%A/%T%A/%T%A/%T%A/%T?

EFM2: .ASCIZ /%S5%ATOTAL MISMATCHES IN MSG = %D5/

PCPM: .ASCIZ /%N%S3%ACALLED FROM PC=%06/

EFM11: .ASCIZ /%S5%ACOMPARE COUNT=%D5%S3%ARECEIVE COUNT=%D5/

MSCMS: .ASCIZ /MODEM STATUS CHANGES FOR THIS PASS WERE../

EFM13: .ASCIZ /%S5%AHARD CHANGES=%D5%A%S3%AGLITCHES=%D5/


```

2610
2611
2612 ;EVENT DESCRIPTION MESSAGES
2613
2614 014202 051124 047101 046523 EDTXQ: .ASCIZ /TRANSMIT MSG QUEUED/
2615 014210 052111 046440 043523
2616 014216 050440 042525 042525
2617 014224 000104
2618 014226 051124 047101 046523 EDTXC: .ASCIZ /TRANSMIT MSG COMPLETED/
2619 014234 052111 046440 043523
2620 014242 041440 046517 046120
2621 014250 052105 042105 000
2622 014255 122 041505 044505 EDRXQ: .ASCIZ /RECEIVE SPACE QUEUED/
2623 014262 042526 051440 040520
2624 014270 042503 050440 042525
2625 014276 042525 000104
2626 014302 042522 042503 053111 EDRXC: .ASCIZ /RECEIVE MSG COMPLETED/
2627 014310 020105 051515 020107
2628 014316 047503 050115 042514
2629 014324 042524 000104
2630 014330 042504 044526 042503 EDDER: .ASCIZ /DEVICE ERROR/
2631 014336 042440 051122 051117
2632 014344 000
2633 014345 104 052101 020101 EDDCK: .ASCIZ /DATA COMPARISON STARTED/
2634 014352 047503 050115 051101
2635 014360 051511 047117 051440
2636 014366 040524 052122 042105
2637 014374 000
2638 014375 104 053105 041511 EDDVI: .ASCIZ /DEVICE INIT AND SETUP/
2639 014402 020105 047111 052111
2640 014410 040440 042116 051440
2641 014416 052105 050125 000
2642 014423 104 052101 020101 EDDLE: .ASCIZ /DATA COMPARISON LENGTH ERROR/
2643 014430 047503 050115 051101
2644 014436 051511 047117 046040
2645 014444 047105 052107 020110
2646 014452 051105 047522 000122
2647 014460 040504 040524 041440 EDDDE: .ASCIZ /DATA COMPARISON DATA ERROR/
2648 014466 046517 040520 044522
2649 014474 047523 020116 040504
2650 014502 040524 042440 051122
2651 014510 051117 000
2652 014513 105 042116 047440 EDEOP: .ASCIZ /END OF PASS/
2653 014520 020106 040520 051523
2654 014526 000
2655 014527 115 042117 046505 EDMOS: .ASCIZ /MODEM STATUS CHANGE/
2656 014534 051440 040524 052524
2657 014542 020123 044103 047101
2658 014550 042507 000
2659
2660 ;EVENT REPORTING MESSAGES
2661 014553 045 031523 047445 BASM3: .ASCIZ /%S3%03/
2662 014560 000063
2663 014562 051445 022463 033117 BASM2: .ASCIZ /%S3%06/
2664 014570 000
2665 014571 045 022516 033117 BASM1: .ASCIZ /%N%06/

```


2722 015264 051101 020105 054502
2723 015272 042524 036523 042045
2724 015300 000005
2725
2726 015302 047045 051445 022463
2727 015310 050101 051501 036523
2728 015316 042045 022465 031523
2729 015324 040445 051105 047522
2730 015332 051522 022475 032504
2731 015340 051445 022463 051501
2732 015346 051124 026524 047524
2733 015354 022475 032504 000
2734
2735 015361 045 032523 040445
2736 015366 054502 042524 020043
2737 015374 047111 046440 043523
2738 015402 036456 042045 022465
2739 015410 031523 040445 054105
2740 015416 052120 036504 047445
2741 015424 022463 031523 040445
2742 015432 042522 053103 036504
2743 015440 047445 000063
2744
2745 015444 047045 051445 022471
2746 015452 041501 040510 043516
2747 015460 042105 052040 035117
2748 015466 000
2749
2750
2751
2752
2753 015467 045 022516 034123
2754 015474 040445 047515 042504
2755 015502 020115 052123 052101
2756 015510 051525 020072 052103
2757 015516 020123 051504 020122
2758 015524 041504 020104 052122
2759 015532 020123 044522 020040
2760 015540 050523 020104 046524
2761 015546 000
2762 015547 045 022516 034523
2763 015554 051445 022471 032523
2764 015562 040445
2765 015564 130 040 040
2766 015567 040
2767 015570 130 040 040
2768 015573 040
2769 015574 130 040 040
2770 015577 040
2771 015600 130 040 040
2772 015603 040
2773 015604 130 040 040
2774 015607 040
2775 015610 130 040 040
2776 015613 040
2777 015614 130 040 040

EVT4B: .ASCIZ /%N%S3%APASS=%D5%S3%AERRORS=%D5%S3%ASTRT-TO=%D5/

EVT5A: .ASCIZ /%S5%ABYTE# IN MSG.=%D5%S3%AEXPTD=%O3%S3%ARECVD=%O3/

EVMOCG: .ASCIZ /%N%S9%ACHANGED TO: /

: *****
;DO NOT SEPERATE THE NEXT LIST OF MESSAGES - MODEM SIGNAL HEADER AND REPORT

EVMOHD: .ASCIZ /%N%S8%AMODEM STATUS: CTS DSR DCD RTS RI SQD TM/

EVMOST: .ASCII /%N%S9%S9%S5%A/

EVMCTS: .BYTE 'X,40,40,40

EVMDSR: .BYTE 'X,40,40,40

EVMDCD: .BYTE 'X,40,40,40

EVMRTS: .BYTE 'X,40,40,40

EVMRI: .BYTE 'X,40,40,40

EVMSQD: .BYTE 'X,40,40,40

EVMTM: .BYTE 'X,40,40,40


```

2778 015617 040
2779 015620 000
2780 015622
2781
2782 ;EXECUTION STATUS MESSAGES TO BE PRINTED TO KEEP OPERATOR AWAKE
2783 015622 047045 000 CR: .ASCIZ /%N/ ;CR FOR LINES IN A ROW
2784 015625 045 031523 040445 STXQ: .ASCIZ /%S3%ATXQ/ ;ABOUT TO TRANSMIT
2785 015632 054124 000121
2786 015636 051445 022463 052101 STXC: .ASCIZ /%S3%ATXC/ ;TX COMPLETED
2787 015644 041530 000
2788 015647 045 031523 040445 SRXQ: .ASCIZ /%S3%ARXQ/ ;ABOUT TO RECEIVE
2789 015654 054122 000121
2790 015660 051445 022463 042501 SDVE: .ASCIZ /%S3%AERR/ ;DEVICE ERROR
2791 015666 051122 000
2792 015671 045 031523 040445 SCM: .ASCIZ /%S3%ACMP/ ;ABOUT TO DO DATA CHECKING OF RECVD VS. EXPTD
2793 015676 046503 000120
2794 015702 051445 022463 044501 SDVI: .ASCIZ /%S3%AINI/ ;DEVICE ABOUT TO BE INITIALIZED
2795 015710 044516 000
2796 015713 045 031523 040445 SCML: .ASCIZ /%S3%ACML/ ;COMPARE LENGTH ERROR
2797 015720 046503 000114
2798 015724 051445 022463 041501 SCMD: .ASCIZ /%S3%ACMD/ ;COMPARE DATA ERROR
2799 015732 042115 000
2800 015735 045 031523 040445 SEOP: .ASCIZ /%S3%AEOP/ ;END OF PASS
2801 015742 047505 000120
2802 .EVEN
2803
2804 015746 051445 022463 046501 SMSC: .ASCIZ /%S3%AMSC/ ;MODEM STATUS CHANGE.
2805 015754 041523 000
2806
2807 015757 115 042117 046505 GLMSG: .ASCIZ /MODEM STATUS GLITCHED/
2808 015764 051440 040524 052524
2809 015772 020123 046107 052111
2810 016000 044103 042105 000
2811 016005 115 042117 046505 HRDMSG: .ASCIZ /MODEM STATUS HARD ERROR/
2812 016012 051440 040524 052524
2813 016020 020123 040510 042122
2814 016026 042440 051122 051117
2815 016034 000
2816

```

2817					
2818					
2819	016035	115	051501	042524	DVEM0: .ASCII /MASTER RESET DID NOT WORK/
2820	016042	020122	042522	042523	
2821	016050	020124	044504	020104	
2822	016056	047516	020124	047527	
2823	016064	045522			
2824	016066	005015	020040	051040	.ASCIZ <15><12>/ RXCSR TXCSR /
2825	016074	041530	051123	020040	
2826	016102	020040	054124	051503	
2827	016110	020122	000040		
2828	016114	047516	041440	042514	DVEM1: .ASCII /NO CLEAR TO SEND FROM MODEM /
2829	016122	051101	052040	020117	
2830	016130	042523	042116	043040	
2831	016136	047522	020115	047515	
2832	016144	042504	020115		
2833	016150	005015	020040	051040	.ASCIZ <15><12>/ RXCSR TXCSR /
2834	016156	041530	051123	020040	
2835	016164	020040	054124	051503	
2836	016172	020122	000040		
2837	016176	044524	042515	047440	DVEM2: .ASCII /TIME OUT WAITING FOR RX OR TX TO COMPLETE/
2838	016204	052125	053440	044501	
2839	016212	044524	043516	043040	
2840	016220	051117	051040	020130	
2841	016226	051117	052040	020130	
2842	016234	047524	041440	046517	
2843	016242	046120	052105	105	
2844	016247	015	020012	020040	.ASCIZ <15><12>/ RXCSR TXCSR/
2845	016254	054122	051503	020122	
2846	016262	020040	052040	041530	
2847	016270	051123	000		
2848	016273	103	041522	044440	DVEM3: .ASCII /CRC IN ERROR/
2849	016300	020116	051105	047522	
2850	016306	122			
2851	016307	015	020012	020040	.ASCIZ <15><12>/ RDSR RXCSR/
2852	016314	042122	051123	020040	
2853	016322	020040	051040	041530	
2854	016330	051123	000		
2855	016333	122	041505	044505	DVEM4: .ASCII /RECEIVER OVERRUN/
2856	016340	042526	020122	053117	
2857	016346	051105	052522	116	
2858	016353	015	020012	020040	.ASCIZ <15><12>/ RDSR RXCSR/
2859	016360	042122	051123	020040	
2860	016366	020040	051040	041530	
2861	016374	051123	000		
2862	016377	124	046511	042105	DVEM5: .ASCII /TIMED OUT IN START,STACK,ACK SEQ/
2863	016404	047440	052125	044440	
2864	016412	020116	052123	051101	
2865	016420	026124	052123	041501	
2866	016426	026113	041501	020113	
2867	016434	042523	121		
2868	016437	015	020012	020040	.ASCIZ <15><12>/ RDATA SDATA/
2869	016444	042122	052101	020101	
2870	016452	020040	051440	040504	
2871	016460	040524	000		
2872	016463	115	042117	046505	DVEM6: .ASCII /MODEM DID NOT RETURN MODEM READY/

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 66
GLOBAL FORMAT STATEMENTS, MESSAGES, AND ASCII INFO

SEQ 0065

2873	016470	042040	042111	047040
2874	016476	052117	051040	052105
2875	016504	051125	020116	047515
2876	016512	042504	020115	042522
2877	016520	042101	131	
2878	016523	015	020012	020040
2879	016530	054122	051503	020122
2880	016536	020040	020040	054124
2881	016544	051503	000122	
2882				

.ASCIIZ <15><12>/ RXCSR TXCSR/

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{B 6} 10:29 PAGE 67
GLOBAL FORMAT STATEMENTS, MESSAGES, AND ASCII INFO

SEQ 0066

2883
2884
2885
2886
2887
2888
2889
2890
2891
2892

.EVEN

2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905 016550
2906 016550
2907 016550
2908 016550 005046
2909 016552 153716 006537
2910 016556 005046
2911 016560 153716 006536
2912 016564 013746 006514
2913 016570 012746 015361
2914 016574 012746 000004
2915 016600 010600
2916 016602 104414
2917 016604 062706 000012
2918 016610
2919 016610
2920 016610 104423
2921
2922 016612
2923 016612
2924 016612
2925 016612 013746 006526
2926 016616 012746 013705
2927 016622 012746 000002
2928 016626 010600
2929 016630 104414
2930 016632 062706 000006
2931 016636
2932 016636
2933 016636 104423
2934
2935 016640
2936 016640
2937 016640
2938 016640 013746 006524
2939 016644 010446
2940 016646 012746 014002
2941 016652 012746 000003
2942 016656 010600
2943 016660 104414
2944 016662 062706 000010
2945 016666
2946 016666
2947 016666 104423
2948

.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 #EVT5A,OFSET,<B,GOOD>,<B,BAD>

ERR1::

;INDIVIDUAL DATA COMPARE ERROR

CLR -(SP)
BISB BAD,(SP)
CLR -(SP)
BISB GOOD,(SP)
MOV OFSET,-(SP)
MOV #EVT5A,-(SP)
MOV #4,-(SP)
MOV SP,RO
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,RO
TRAP C\$PNTB
ADD #6,SP

ENDMSG

L10002:

TRAP C\$MSG

BGNMSG ERR10

PRINTB #EFM11,R4,TEMP3

ERR10::

;LENGH COMPARISON ERROR

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

ENDMSG

L10003:

TRAP C\$MSG

2949	016670			BGNMSG	ERR4		
2950	016670						
2951	016670			PRINTB	#EFM13,MHRCNT,MGLCNT	ERR4::	
2952	016670	013746	011414			:MODEM STATUS CHANGE	
2953	016674	013746	011416			MOV	MGLCNT,-(SP)
2954	016700	012746	014131			MOV	MHRCNT,-(SP)
2955	016704	012746	000003			MOV	#EFM13,-(SP)
2956	016710	010600				MOV	#3,-(SP)
2957	016712	104414				MOV	SP,RO
2958	016714	062706	000010			TRAP	C\$PNTB
2959	016720			ENDMSG		ADD	#10,SP
2960	016720					L10004:	
2961	016720	104423				TRAP	C\$MSG
2962							
2963							
2964							
2965							
2966				:	PRINT THE 2 OCTAL #'S IN TEMP3/4		
2967				:			
2968				:			
2969	016722			BGNMSG	ERR13	ERR13::	
2970	016722						
2971	016722			PRINTB	#EVTF3C,TEMP3,TEMP4		
2972	016722	013746	006526			MOV	TEMP4,-(SP)
2973	016726	013746	006524			MOV	TEMP3,-(SP)
2974	016732	012746	015043			MOV	#EVTF3C,-(SP)
2975	016736	012746	000003			MOV	#3,-(SP)
2976	016742	010600				MOV	SP,RO
2977	016744	104414				TRAP	C\$PNTB
2978	016746	062706	000010			ADD	#10,SP
2979	016752			ENDMSG		L10005:	
2980	016752					TRAP	C\$MSG
2981	016752	104423					
2982							
2983				:	PRINT THE 2 OCTAL #'S IN TEMP3/4		
2984				:	AND THE MMSG. WHOSE ADDR. IS IN CONOTM		
2985				:			
2986				:			
2987				:			
2988	016754			BGNMSG	ERR14	ERR14::	
2989	016754						
2990	016754			PRINTB	#EVTF3D,TEMP3,TEMP4,CONOTM		
2991	016754	013746	006532			MOV	CONOTM,-(SP)
2992	016760	013746	006526			MOV	TEMP4,-(SP)
2993	016764	013746	006524			MOV	TEMP3,-(SP)
2994	016770	012746	015060			MOV	#EVTF3D,-(SP)
2995	016774	012746	000004			MOV	#4,-(SP)
2996	017000	010600				MOV	SP,RO
2997	017002	104414				TRAP	C\$PNTB
2998	017004	062706	000012			ADD	#12,SP
2999	017010			ENDMSG		L10006:	
3000	017010					TRAP	C\$MSG
3001	017010	104423					
3002							
3003	017012			EXIT	MSG	.WORD	JSJMP
3004	017012	000167					

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{E 6} 10:29 PAGE 70
GLOBAL ERROR REPORT SECTION

SEQ 0069

3005 017014 177772
3006
3007

.WORD L10006-2-.

3008
 3009
 3010
 3011
 3012
 3013
 3014
 3015
 3016
 3017
 3018
 3019
 3020
 3021
 3022
 3023
 3024
 3025
 3026
 3027
 3028
 3029
 3030
 3031
 3032
 3033
 3034
 3035
 3036
 3037
 3038
 3039
 3040
 3041
 3042
 3043
 3044
 3045
 3046
 3047
 3048
 3049
 3050
 3051
 3052
 3053
 3054

017016
 017016 012122
 017020 012112
 017022 006312
 017024 006312
 017026 006312
 017030 006312
 017032 006322
 017034 012122
 017036 012122
 017040 000207

.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

3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110

017042
017042
017042 005077 167532
017046 005337 006616
017052 001015
017054 013737 006606 006616
017062 005237 006614
017066 022737 000074 006614
017074 001004
017076 005237 006612
017102 005037 006614
017106 005737 006620
017112 001402
017114 005337 006620
017120 005737 006622
017124 001402
017126 005337 006622
017132 005737 006624
017136 001406
017140 023737 006606 006616
017146 001002
017150 005337 006624

.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, & 3: 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 "3" 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.

```

```

--
BGNSRV CLKINT CLKINT::
:
: CLR @CLKCSR ;DISABLE THE CLOCK FROM INTERRUPTING
: DEC TIMTCK ;DECREMENT THE # OF TICKS/SEC.
: BNE 1$ ;GO CHECK TIMERS (182-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.)

```



```
3111 017154 013777 006610 167416 4$: MOV CLKEN,@CLKCSR ;REENABLE THE CLOCK TO INTERRUPT
3112 017162 ENDSRV
3113 017162 L10007:
3114 017162 000002 RTI
```

3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170

```
.SBTTL      EVENT LOG SUBROUTINES

:++
: FUNCTIONAL DESCRIPTION:
: THIS SUBROUTINE HAS A DIFFERENT ENTRY POINT
: FOR EACH EVENT TO BE LOGGED AND ALWAYS PRINTS
: THE SHORT "OPERATOR AWAKE" MESSAGE TO CONSOLE THEN LOGS THE
: EVENT TYPE, TIME, AND THE OTHER 3 WORDS OF INFO PASSED TO THE
: SUBROUTINE AT CALLING TIME

: INPUTS:
: TIMMIN & TIMSEC:      CURRENT VALUE OF "TIME-SINCE-START"
: TEMP2: WORD #1 OF EVENT LOG INFORMATION (FOR MOST EVENT TYPES)
: TEMP3: WORD #2 OF EVENT LOG INFORMATION
: TEMP4: WORD #3 OF EVENT LOG INFORMATION
: MODS:  CURRENT VALUE OF THE MODEM SIGNALS AVAILABLE FROM THE DEVICE

: OUTPUTS:
: "OPERATOR AWAKE" MESSAGE SENT TO THE CONSOLE
: NEW EVENT LOGGED IN "EVTLOG" (EVENT LOG)
: UPDATED "EVTPTN" (EVENT LOG ENTRY POINTER)

: SUBORDINATE ROUTINES USED:
: "DVMODS" THE DEVICE SUBROUTINE THAT RETURNS MODEM STATUS IN "MODS"
: (FOR SOME EVENT TYPES)

: FUNCTIONAL SIDE EFFECTS:
: TEMP:  USED TO STORE ADDRESS OF "OPERATOR AWAKE" MESSAGE
: TEMP1: USED TO SETUP THE VALUE OF THE "EVENT TYPE" BYTE FOR LOGGING

: CALLING SEQUENCE:
: JSR    PC,LOGTXQ      ;CALL THE LOG EVENT SUBROUTINE WITH TEMP,TEMP1,
: ..     ..             ; TEMP2, TEMP3, AND TEMP4 SETUP
: ..     ..             ;
: JSR    PC,LOGCMP

:--

LOGTXQ:  MOV    #STXQ,TEMP1  ;SET UP MSG. TO PRINT
:        MOV    #TXQ,TEMP   ;SET UP EVENT TYPE
:        BR     LOGS1       ;GO LOG EVENT AND TIME

LOGTXC:  MOV    #STXC,TEMP1  ;SET UP MSG. TO PRINT
:        MOV    #TXC,TEMP   ;SET UP EVENT TYPE
:        BR     LOGS1       ;GO LOG EVENT AND TIME

LOGRXQ:  MOV    #SRXQ,TEMP1  ;SET UP MSG. TO PRINT
:        MOV    #RXQ,TEMP   ;SET UP EVENT TYPE
:        BR     LOGS1       ;GO LOG EVENT AND TIME

LOGRXC:  MOV    #RXC,TEMP   ;SET UP EVENT TYPE
:        BR     LOGS1       ;GO LOG EVENT AND TIME

LGDVE:  
```

```
017164
017164 012737 015625 006520
017172 012737 000000 006516
017200 000517
017202
017202 012737 015636 006520
017210 012737 000002 006516
017216 000510
017220
017220 012737 015647 006520
017226 012737 000004 006516
017234 000501
017236
017236 012737 000006 006516
017244 000475
017246
```

3171	017246	012737	015660	006520	MOV	#SDVE,TEMP1	;SET UP MSG. TO PRINT
3172	017254	012737	000010	006516	MOV	#DER,TEMP	;SET UP EVENT TYPE
3173	017262	000503			BR	LOGS3	;GO LOG EVENT AND TIME
3174							
3175	017264				LOGDVI:		
3176	017264	012737	015702	006520	MOV	#SDVI,TEMP1	;SET UP MSG. TO PRINT
3177	017272	012737	000012	006516	MOV	#DVI,TEMP	;SET UP EVENT TYPE
3178	017300	113737	006546	006522	MOV	MODTYP,TEMP2	
3179	017306	113737	006550	006523	MOV	MLTYP,TEMP2+1	
3180	017314	013737	006556	006524	MOV	RPASS,TEMP3	
3181	017322	013737	006554	006526	MOV	PARAM,TEMP4	;SET UP EVNT ENTRIES
3182	017330	000460			BR	LOGS3	;GO LOG EVENT AND TIME
3183							
3184	017332				LOGCMP:		
3185	017332	012737	015671	006520	MOV	#SCM,TEMP1	;SET UP MSG. TO PRINT
3186	017340	012737	000014	006516	MOV	#DCK,TEMP	;SET UP EVENT TYPE
3187	017346	000451			BR	LOGS3	
3188	017350				LOGCML:		
3189	017350	012737	015713	006520	MOV	#SCML,TEMP1	
3190	017356	012737	000020	006516	MOV	#DLE,TEMP	;SET UP MSG. AND TYPE
3191	017364	000442			BR	LOGS3	;GO LOG EVENT AND TIME
3192	017366				LOGCMD:		
3193	017366	012737	015724	006520	MOV	#SCMD,TEMP1	
3194	017374	012737	000022	006516	MOV	#DDE,TEMP	
3195	017402	000433			BR	LOGS3	;GO LOG MSG TYPE AND TIME
3196	017404				LOGEOP:		
3197	017404	012737	015735	006520	MOV	#SEOP,TEMP1	
3198	017412	012737	000024	006516	MOV	#EOP,TEMP	
3199	017420	000424			BR	LOGS3	;GO LOG MSG TYPE AND TIME
3200							
3201							
3202	017422				LOGMSC:		
3203	017422	012737	015746	006520	MOV	#SMSC,TEMP1	
3204	017430	012737	000016	006516	MOV	#MSC,TEMP	
3205	017436	000415			BR	LOGS3	
3206							
3207							
3208	017440	013746	006470		LOGS1:	MOV	ERRCNT,-(SP)
3209	017444	004737	032042			JSR	PC,DVMODS
3210	017450	012604				MOV	(SP)+,R4
3211	017452	020437	006470			CMP	R4,ERRCNT
3212	017456	001402				BEQ	1\$
3213	017460	000137	017674			JMP	LOGEX
3214							
3215	017464	013737	007534	006526	1\$:	MOV	MODS,TEMP4
3216							
3217	017472				LOGS3:		
3218	017472	022737	000006	006516	CMP	#RXC,TEMP	
3219	017500	001434			BEQ	LOGS5	;IF RXC DONT PRINT
3220	017502	032737	000001	006554	BIT	#STATB,PARAM	
3221	017510	001430			BEQ	LOGS5	;IF NO STATUS SELECTED
3222							;GO TO 5
3223							
3224	017512	022737	000010	006462	CMP	#10,LNCNT	;HAVE WE DONE 10?
3225	017520	001012			BNE	LOGS4	;IF NOT GO TO 4
3226	017522	005037	006462		CLR	LNCNT	;ESLE CLEAR IT


```

3227
3228 017526          PRINTF #CR          ;ELSE PRINT CR
3229 017526 012746 015622          MOV #CR,-(SP)
3230 017532 012746 000001          MOV #1,-(SP)
3231 017536 010600          MOV SP,R0
3232 017540 104417          TRAP C$PNTF
3233 017542 062706 000004          ADD #4,SP
3234 017546
LOGS4:
3235 017546 005237 006462          INC LNCNT          ;INC COUNTER OF # OF AWAKE MSGS
3236 017552          PRINTF TEMP1          ;PRINT OPERATOR AWAKE MSG.
3237 017552 013746 006520          MOV TEMP1,-(SP)
3238 017556 012746 000001          MOV #1,-(SP)
3239 017562 010600          MOV SP,R0
3240 017564 104417          TRAP C$PNTF
3241 017566 062706 000004          ADD #4,SP
3242 017572 010346
LOGS5:
3243 017574 013703 006626          MOV R3,-(SP)          ;SAVE R3 ON THE STACK
3244 017600 113723 006516          MOV EVTPTR,R3
3245 017604 013737 006606 006516          MOVB TEMP,(R3)+          ;LOG EVENT
3246 017612 163737 006616 006516          MOV CLKHZ,TEMP
3247 017620 113723 006516          SUB TIMTCK,TEMP
3248 017624 113723 006614          MOVB TEMP,(R3)+          ;LOG TIME SINCE START
3249 017630 113723 006612          MOVB TIMSEC,(R3)+
3250 017634 013723 006522          MOVB TIMMIN,(R3)+          ;TICKS,SECS AND MINS.
3251 017640 013723 006524          MOV TEMP2,(R3)+          ;LOG EVNT ENTRY 3
3252 017644 013723 006526          MOV TEMP3,(R3)+          ;LOG EVNT ENTRY 4
3253 017650 020327 007532          MOV TEMP4,(R3)+          ;LOG EVNT ENTRY 5
3254 017654 103404          CMP R3,#EVTEND
3255          BLO LOGS2          ;IF EVENT LOG FULL GO
3256 017656 012713 177777          ;CONTINUE;ELSE GO TO 2
3257 017662 012703 006630          MOV #-1,(R3)          ;LOG A TABLE END
3258 017666 010337 006626          MOV #EVTLOG,R3          ;PUT R3 TO START OF TABLE
LOGS2:
3259 017672 012603          MOV R3,EVTPTR          ;RESTORE POINTER
LOGEX:
3260 017674 000207          MOV (SP)+,R3          ;RESTORE R3
3261          RTS
3262          PC

```

```

3263
3264
3265
3266 017676 010246
3267 017700 010346
3268 017702 010446
3269
3270
3271
3272
3273 017704 013702 006626
3274 017710 023727 006630 177777
3275 017716 001034
3276 017720
3277 017720 012746 014577
3278 017724 012746 000001
3279 017730 010600
3280 017732 104416
3281 017734 062706 000004
3282 017740 000137 020624
3283
3284 017744 162702 000012
3285
3286
3287 017750 020227 006630
3288 017754 001010
3289 017756 012702 007532
3290 017762 026227 177776 177777
3291 017770 001007
3292 017772 000137 020624
3293
3294 017776 020237 006626
3295 020002 001002
3296 020004 000137 020624
3297
3298 020010 162702 000012
3299 020014
3300 020014 012746 014632
3301 020020 012746 000001
3302 020024 010600
3303 020026 104416
3304 020030 062706 000004
3305 020034 112203
3306 020036 112237 007624
3307 020042 112237 007620
3308 020046 112237 007622
3309 020052
3310 020052 016346 007572
3311 020056 013746 007624
3312 020062 013746 007620
3313 020066 013746 007622
3314 020072 012746 014730
3315 020076 012746 000005
3316 020102 010600
3317 020104 104416
3318 020106 062706 000014

```

```

.SBTTL          DUMP EVENT LOG AND BASE TABLE

REPORT: MOV      R2,-(SP)          ;SAVE R2,R3,R4 ON THE STACK
        MOV      R3,-(SP)
        MOV      R4,-(SP)

        MOV      EVTPTR,R2        ;MAKE R2 A POINTER TO EVENT TABLE
        CMP      EVTLOG,#-1      ;SEE IF EVENT TABLE IS EMPTY
        BNE      RPT0            ;BR IF NO
        PRINTS   #NULEVT        ;IF EMPTY TELL OPERATOR.

                                MOV      #NULEVT,-(SP)
                                MOV      #1,-(SP)
                                MOV      SP,R0
                                TRAP     C$PNTS
                                ADD      #4,SP

        JMP      ENDEVT          ;AND END

RPT:   SUB      #12,R2          ;NOW POINT BACK TO TOP OF ENTRY U
                                ;JUST PRINTED

        CMP      R2,#EVTLOG      ;POINTING TO TOP OF EVNT LOG QUEUE?
        BNE      RPT1            ;BR IF NO
        MOV      #EVTEND,R2     ;SET R2 TO POINT TO BOTTOM OF LOG
        CMP      -2(R2),#-1
        BNE      RPT0            ;IF END OF LOG IS NOT EMPTY
        JMP      ENDEVT          ;CONTINUE...ELSE EXIT

RPT1:  CMP      R2,EVTPTR        ;ARE WE BACK TO POINTER?
        BNE      RPT0            ;IF NOT CONTINUE
        JMP      ENDEVT          ;IF SO EXIT....

RPT0:  SUB      #12,R2          ;POINT R2 TO START OF ENTRY
RPTAA: PRINTS   #EVTFO          ;PRINT EVENT ENTRY HEADER

                                MOV      #EVTFO,-(SP)
                                MOV      #1,-(SP)
                                MOV      SP,R0
                                TRAP     C$PNTS
                                ADD      #4,SP

        MOVB     (R2)+,R3        ;PUT EVENT TYPE INTO R3
        MOVB     (R2)+,EVTICK    ;PUT EVENT TIME (TICKS,SECS,MINS IN TEMP LOC.S)
        MOVB     (R2)+,EVTSEC
        MOVB     (R2)+,EVTMIN
        PRINTS   #EVTf1,EVTMIN,EVTSEC,EVTICK,EVTlST(R3) ;PRINT EVENT TIME AND DESCRIPT.

                                MOV      EVTlST(R3),-(SP)
                                MOV      EVTTCK,-(SP)
                                MOV      EVTSEC,-(SP)
                                MOV      EVTMIN,-(SP)
                                MOV      #EVTf1,-(SP)
                                MOV      #5,-(SP)
                                MOV      SP,R0
                                TRAP     C$PNTS
                                ADD      #14,SP

```



```

3319 020112 000173 007634          JMP      @RPTDSP(R3)      ;DISPATCH TO DECODING SECTION FOR SPECIFIC TYPE
3320
3321 020116 012237 007626          RPTTXQ: MOV      (R2)+,EVTADD ;STORE MESSAGE ADDRESS FOR PRINTING
3322 020122 012237 007630          MOV      (R2)+,EVTBCT   ;STORE BYTE COUNT FOR PRINTING
3323 020126 012203                   MOV      (R2)+,R3       ;STORE MODEM STATUS FOR PRINTING
3324 020130                   PRINTS  #EVTF2,EVTADD,EVTBCT ;PRINT ADDR,BYTE CNT
3325 020130 013746 007630                   MOV      EVTBCT,-(SP)
3326 020134 013746 007626                   MOV      EVTADD,-(SP)
3327 020140 012746 014757                   MOV      #EVTF2,-(SP)
3328 020144 012746 000003                   MOV      #3,-(SP)
3329 020150 010600                   MOV      SP,R0
3330 020152 104416                   TRAP    C$PNTS
3331 020154 062706 000010                   ADD     #10,SP
3332 020160 004737 020634          JSR     PC,RPTMSB      ;GO PRINT MODEM STATUS
3333 020164 000137 017744          JMP     RPT            ;GO BACK FOR NEXT EVENT ENTRY
3334
3335 020170 012237 007632          RPTDER: MOV      (R2)+,EVTTMP ;GET ADDRESS OF DEVICE INFO MESSAGE
3336 020174 012237 007662          MOV      (R2)+,DEV1     ;STORE DEVICE REG CONTENTS FOR PRINTING
3337 020200 012237 007664          MOV      (R2)+,DEV2
3338 020204                   PRINTS  #EVTF3,EVTTMP   ;PRINT DEVICE REG CONTENTS.
3339 020204 013746 007632                   MOV      EVTTMP,-(SP)
3340 020210 012746 015031                   MOV      #EVTF3,-(SP)
3341 020214 012746 000002                   MOV      #2,-(SP)
3342 020220 010600                   MOV      SP,R0
3343 020222 104416                   TRAP    C$PNTS
3344 020224 062706 000006                   ADD     #6,SP
3345 020230                   PRINTS  #EVTF3C,DEV1,DEV2
3346 020230 013746 007664                   MOV      DEV2,-(SP)
3347 020234 013746 007662                   MOV      DEV1,-(SP)
3348 020240 012746 015043                   MOV      #EVTF3C,-(SP)
3349 020244 012746 000003                   MOV      #3,-(SP)
3350 020250 010600                   MOV      SP,R0
3351 020252 104416                   TRAP    C$PNTS
3352 020254 062706 000010                   ADD     #10,SP
3353 020260 000137 017744          JMP     RPT            ;GO BACK FOR NEXT EVENT ENTRY
3354
3355 020264 005037 007662          RPTDVI: CLR     DEV1
3356 020270 005037 007664          CLR     DEV2          ;CLEAR UPPER BYTES OF DEV1 & DEV2 BEFORE USE
3357 020274 112237 007662          MOV     (R2)+,DEV1    ;STORE SETUP OPERATION PARAMETERS FOR PRINTING
3358 020300 112237 007664          MOV     (R2)+,DEV2
3359 020304 012237 007666          MOV     (R2)+,DEV3
3360 020310 012237 007670          MOV     (R2)+,DEV4
3361 020314 010246                   MOV     R2,-(SP)      ;SAVE R2 ON THE STACK
3362 020316 004737 021326          JSR     PC,SHWOP      ;GO PRINT MODE, MAINT-LOOP TYPE, PARAMTERS.
3363 020322 012602                   MOV     (SP)+,R2     ;RESTORE R2
3364 020324 000137 017744          JMP     RPT            ;GO BACK FOR NEXT EVENT ENTRY
3365 020330 012237 007626          RPTTEOP: MOV     (R2)+,EVTADD
3366 020334 012237 007630          MOV     (R2)+,EVTBCT
3367 020340 012237 007632          MOV     (R2)+,EVTTMP
3368 020344                   PRINTS  #EVTF4B,EVTADD,EVTBCT,EVTTMP ;PRINT ADDR,RXBYTES,CMPBYTES.
3369 020344 013746 007632                   MOV     EVTTMP,-(SP)
3370 020350 013746 007630                   MOV     EVTBCT,-(SP)
3371 020354 013746 007626                   MOV     EVTADD,-(SP)
3372 020360 012746 015302                   MOV     #EVTF4B,-(SP)
3373 020364 012746 000004                   MOV     #4,-(SP)
3374 020370 010600                   MOV     SP,R0

```



```

3375 020372 104416
3376 020374 062706 000012
3377
3378 020400 000137 017744          JMP      RPT          ;THEN GO GET NEXT EVENT ENTRY
3379
3380
3381 020404 012237 007626          RPTDDE: MOV      (R2)+,EVTADD      ;STORE MESSAGE ADDRESS FOR PRINTING
3382 020410 012237 007630          MOV      (R2)+,EVTBCT      ;STORE BYTE COUNT FOR PRINTING
3383 020414 012237 007632          MOV      (R2)+,EVTTMP      ;STORE TOTAL # OF CMP ERRORS
3384 020420          PRINTS  #EVT4,EVTADD,EVTBCT,EVTMP ;PRINT ADDR, BYTE CNT, # CMP ERRS
3385 020420 013746 007632          MOV      EVTMP,-(SP)
3386 020424 013746 007630          MOV      EVTBCT,-(SP)
3387 020430 013746 007626          MOV      EVTADD,-(SP)
3388 020434 012746 015102          MOV      #EVT4,-(SP)
3389 020440 012746 000004          MOV      #4,-(SP)
3390 020444 010600          MOV      SP,R0
3391 020446 104416          TRAP     C$PNTS
3392 020450 062706 000012          ADD     #12,SP
3393 020454 000137 017744          JMP      RPT          ;THEN GO GET NEXT EVENT ENTRY
3394
3395 020460          RPTDLE:
3396 020460 012237 007626          RPTDCK: MOV      (R2)+,EVTADD      ;STORE MSG ADDR FOR PRINT
3397 020464 012237 007630          MOV      (R2)+,EVTBCT      ;STORE BYTE COUNT
3398 020470 012237 007632          MOV      (R2)+,EVTTMP      ;STORE BYTE COUNT COMP
3399 020474          PRINTS  #EVT4A,EVTADD,EVTBCT,EVTMP ;PRINT ADDR,RXBYTES,CMPBYTES.
3400 020474 013746 007632          MOV      EVTMP,-(SP)
3401 020500 013746 007630          MOV      EVTBCT,-(SP)
3402 020504 013746 007626          MOV      EVTADD,-(SP)
3403 020510 012746 015204          MOV      #EVT4A,-(SP)
3404 020514 012746 000004          MOV      #4,-(SP)
3405 020520 010600          MOV      SP,R0
3406 020522 104416          TRAP     C$PNTS
3407 020524 062706 000012          ADD     #12,SP
3408
3409 020530 000137 017744          JMP      RPT          ;THEN GO GET NEXT EVENT ENTRY
3410
3411
3412
3413
3414
3415 020534 012237 007632          RPTMSC: MOV      (R2)+,EVTMP      ;PRINT CHANGE TYPE
3416 020540          PRINTS  #EVT3,EVTMP
3417 020540 013746 007632          MOV      EVTMP,-(SP)
3418 020544 012746 015031          MOV      #EVT3,-(SP)
3419 020550 012746 000002          MOV      #2,-(SP)
3420 020554 010600          MOV      SP,R0
3421 020556 104416          TRAP     C$PNTS
3422 020560 062706 000006          ADD     #6,SP
3423 020564 012203          MOV      (R2)+,R3          ;PUT OLD MODEM STATUS IN R3 FOR PRINTING
3424 020566 004737 020634          JSR     PC,RPTMSB        ;GO PRINT OLD MODEM STATUS
3425 020572          PRINTS  #EVMOCG          ;GO PRINT "CHANGED TO:"
3426 020572 012746 015444          MOV      #EVMOCG,-(SP)
3427 020576 012746 000001          MOV      #1,-(SP)
3428 020602 010600          MOV      SP,R0
3429 020604 104416          TRAP     C$PNTS
3430 020606 062706 000004          ADD     #4,SP

```

3451 020612 012203
3432 020614 004737 020634
3433 020620 000137 017744
3434
3435
3436 020624 012604
3437 020626 012603
3438 020630 012602
3439 020632 000207
3440
3441
3442
3443
3444
3445 020634
3446 020634 012746 015467
3447 020640 012746 000001
3448 020644 010600
3449 020646 104416
3450 020650 062706 000004
3451 020654 012704 007536
3452 020660 012705 007554
3453 020664 005714
3454 020666 001004
3455 020670 112735 000130
3456 020674 005724
3457 020676 000407
3458 020700 032403
3459 020702 001403
3460 020704 112735 000061
3461 020710 000402
3462 020712 112735 000060
3463 020716 020427 007554
3464 020722 002760
3465 020724
3466 020724 012746 015547
3467 020730 012746 000001
3468 020734 010600
3469 020736 104416
3470 020740 062706 000004
3471 020744 000207
3472
3473

MOV (R2)+,R3 ;PUT NEW MODEM STATUS IN R3 FOR PRINTING
JSR PC,RPTMSB ;GO PRINT NEW MODEM STATUS
RPTMSE: JMP RPT ;THEN GO GET NEXT EVENT

ENDEVT: MOV (SP)+,R4 ;RESTORE R4,R3,R2
MOV (SP)+,R3
MOV (SP)+,R2
RTS PC ;RETURN TO CALLING ROUTINE

;REPORT MODEM STATUS SUBROUTINE
; PART OF STATISICAL REPORTING (DUMPING EVENT LOG)

RPTMSB: PRINTS #EVMOH ;PRINT MODEM STATUS HEADER
MOV #EVMOH,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C\$PNTS
ADD #4,SP

MOV #MOBITS,R4 ;MAKE R4 A POINTER TO MODEM SIG. BIT DEF. TABLE
MOV #MOMSGS,R5 ;MAKE R5 A POINTER TO MODEM MSG. POSITION TABLE
6\$: TST (R4) ;SEE IF BIT AVAILABLE FROM DEVICE
BNE 7\$;BR IF THAT MODEM SIG. AVAILABLE
MOVB #'X,@(R5)+ ;ELSE PUT 'X' IN REPORT IF SIGNAL NOT AVAILABLE
TST (R4)+ ;BUMP R4 TO POINT TO NEXT BIT DEFINITION
BR 9\$;GO SEE IF CHECKED ALL MODEM SIGNALS
7\$: BIT (R4)+,R3 ;IF THERE, SEE IF THAT BIT IN DEVICE'S ENTRY=1
BEQ 8\$;BR IF BIT (SIGNAL) VALUE =0
MOVB #'1,@(R5)+ ;IF=1, PUT '1' IN REPORT MESSAGE
BR 9\$;GO SEE IF ALL MODEM SIGNALS CHECKED
8\$: MOVB #'0,@(R5)+ ;IF BIT(SIGNAL)=0, PUT '0' IN REPORT MESSAGE
9\$: CMP R4,#MOBITE ;SEE IF ALL BITS(SIGNALS) CHECKED
BLT 6\$;LOOP UNTIL ALL SIGNALS(BITS) CHECKED
PRINTS #EVMOST ;THEN PRINT MODEM SIGNAL VALUE MESSAGE
MOV #EVMOST,-(SP)
MOV #1,-(SP)
MOV SP,R0
TRAP C\$PNTS
ADD #4,SP

RTS PC ;RETURN TO EVENT DECODING

3474
3475
3476
3477
3478
3479
3480
3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
3501 020746 013702 006472
3502 020752 005003
3503 020754
3504 020754 010246
3505 020756 012746 014571
3506 020762 012746 000002
3507 020766 010600
3508 020770 104417
3509 020772 062706 000006
3510 020776 005737 006476
3511 021002 001416
3512 021004 112237 006516
3513 021010
3514 021010 005046
3515 021012 153716 006516
3516 021016 012746 014553
3517 021022 012746 000002
3518 021026 010600
3519 021030 104417
3520 021032 062706 000006
3521 021036 000411
3522 021040
3523 021040 012246
3524 021042 012746 014562
3525 021046 012746 000002
3526 021052 010600
3527 021054 104417
3528 021056 062706 000006
3529 021062 020237 006474

.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

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{D 7} 10:29 PAGE 82
DUMP BYTES OR WORDS

SEQ 0081

3530 021066 003005
3531 021070 005203
3532 021072 022703 000010
3533 021076 001725
3534 021100 000736
3535
3536 021102 000207
3537

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

DUMEX: RTS PC

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

3538
3539
3540
3541
3542
3543
3544
3545
3546
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573
3574
3575
3576
3577
3578
3579

.SBTTL UPDATE TOTAL CHAR. COUNT SUBROUTINE

FUNCTIONAL DESCRIPTION:
UPDATES TOTAL CHAR. COUNT TOTCC BASED ON CURCC.
LAST MESSAGE IS TRUNCATED TO FIT INTO THE
BUFFER IF TOTAL CHAR. COUNT EXCEEDS 'BUFLIM' A MESSAGE
IS PRINTED TELLING THE OPERATOR THE TRUNCATION OCCURED.

INPUTS:
CURCC= CHAR. COUNT OF MESSAGE BEING ADDED
TOTCC= TOTAL CHAR COUNT OF BUFFER ITS BEING ADDED TO

OUTPUTS:
MESSAGE TO OPERATOR IF MESSAGE TRUNCATED TO FIT

FUNCTIONAL SIDE EFFECTS:
LOCATION 'TEMP' USED FOR CALCULATIONS

CALLING SEQUENCE:
JSR PC,ADCC ;UPDATED TOTAL CHAR. COUNT

--

ADDCC: ADD CURCC,TOTCC ;ADD CURRENT TO TOTAL
CMP #BUFLIM,TOTCC ; COMPARE TO 'BUFLIM'
BHS ADDC1 ;IF NOT MORE THEN 'BUFLIM' EXIT

; PRINT MESSAGE AND TRUNCATE COUNT

PRINTF #MSGTRU

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

SUB CURCC,TOTCC ;SUB CURRENT FROM TOTAL
MOV #BUFLIM,TEMP ;MOV 'BUFLIM' TO TEMP
SUB TOTCC,TEMP ;SUB TOTAL FROM 'BUFLIM'
MOV TEMP,CURCC ;AND ESTABLISH NEW CURRENT
ADD CURCC,TOTCC ;ADD 'ADJUSTED CURRENT' TO TOTAL CHAR. CNT.
ADDCC1: RTS PC ;RETURN TO CALLER

3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630

021202
021202 010246
021204 010346
021206 013702 006506

021212 013722 006510
021216 013722 006502
021222 010237 006506
021226 013702 006500
021232 006302
021234 013737 006510 006516
021242 063737 006502 006516
021250 013703 006510
021254 016237 002150 006522
021262 016204 002172
021266 060437 006522
021272 112423
021274 020337 006516
021300 001404
021302 020437 006522
021306 001762
021310 000770
021312 063737 006502 006510
021320 012603
021322 012602
021324 000207

```
.SBTTL          BUILD MESSAGE BUFFERS SUBROUTINE

:++
: FUNCTIONAL DESCRIPTION:
:   BLDBUF--  BUILD POINTER TABLE AND BUFFERS
:
:   THIS SUBROUTINE ADDS A MESSAGE TO THE TRANSMIT OR EXPECT LIST
:   USING THE POINTER, BYTE COUNT, AND ADDRESS PASSED TO IT.
:
: INPUTS:
:   CURCC=  CHAR. COUNT OF MESSAGE TO BE ADDED
:   CURADD=  ADDRESS OF MESSAGE TO BE ADDED
:   CPTR=    ADDRESS OF POINTER TABLE WORD WHERE MESSAGE POINTERS ARE
:           TO BE BUILT
:   MSGTYP=  VALUE TO USE AS AN INDEX TO FIND SOURCE OF MESSAGE DATA
:           INDEX INTO DMSGCT() AND DMSGAD().
:
: OUTPUTS:
:   A MESSAGE ADDED TO EITHER TXBUF OR CMPBUF
:   APPROPRIATE POINTERS IN PTRTAB POINTER TABLE
:
: CALLING SEQUENCE:
:   JSR PC,BLDBUF          ;BUILD MESSAGE IN BUFFER AND ADD PTRS.
:--

BLDBUF:
  MOV     R2,-(SP)          ;SAVE R2 AND R3 ON THE STACK
  MOV     R3,-(SP)
  MOV     CPTR,R2

BLDB1:  MOV     CURADD,(R2)+  ;PUT CURRENT ADD ON POINTER TAB
        MOV     CURCC,(R2)+  ;PUT CURRENT CC ON POINTER TAB
        MOV     R2,CPTR      ;PUT UPDATED R2 BACK TO CURRENT POINT
        MOV     MSGTYP,R2    ;GET MESSAGE TYPE TO USE AS INDEX
        ASL     R2           ;DOUBLE FOR WORD INDEX
        MOV     CURADD,TEMP   ;MOVE CURRENT ADD TO TEMP
        ADD     CURCC,TEMP    ;ADD CHAR COUNT TO IT TO GET END
        MOV     CURADD,R3     ;SET R3 TO CURRENT START ADD
BLDB2:  MOV     DMSGCT(R2),TEMP2 ;GET BYTE COUNT
        MOV     DMSGAD(R2),R4 ;PUT STARTING FROM ADD IN R4
        ADD     R4,TEMP2      ;ADD IT TO TEMP2 TO GET END OF FROM
BLDB3:  MOVB    (R4)+,(R3)+    ;MOV BYTE FROM PATTERN TO BUFFER
        CMP     R3,TEMP       ;ALL DONE?
        BEQ    BLDBEX        ;IF SO EXIT
        CMP     R4,TEMP2      ;IS PATTERN COUNT EXPIRED
        BEQ    BLDB2         ;IF SO GO START AGAIN
        BR     BLDB3         ;IF NOT GET ANOTHER BYTE
BLDBEX: ADD     CURCC,CURADD   ;BUMP CURADD
        MOV     (SP)+,R3      ;RESTORE R3 AND R2
        MOV     (SP)+,R2
        RTS     PC           ;RETURN TO CALLER
```


3651
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686

.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

--

```

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    #LPO0,TEMP3     ;LOAD TEMP3 TO POINT TO "/LOOP="
        TST    R2              ;SEE IF /LOOP=XXXXX OR NONE
        BNE   10$              ;BR IF /LOOP= OF SOME KIND
        MOV    #LPO,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      ;SEE IF /CHECK OR /NOCHECK
        BIT    #DATCKB,DEV4   ;BR IF /CHECK
        BNE   2$
        MOV    #PNCK,TEMP1
        MOV    #PEC,TEMP2      ;SEE IF /ECHO OR /NOECHO
        BIT    #ECHOB,DEV4    ;BR IF /ECHO
        BNE   3$
        MOV    #PNEC,TEMP2
    
```

```
3687
3688 021542 012737 013146 006530 3$:  MOV  #PMS,TEMP5
3689 021550 032737 000010 007670      BIT  #MOCHK,DEV4      ;SEE IF /MODEM OR /NOMODEM
3690 021556 001003                BNE  5$              ;BR IF MODEM
3691 021560 012737 013144 006530      MOV  #PNMS,TEMP5
3692
3693
3694 021566                5$:  PRINTS #SHF1,TEMP,TEMP1,TEMP2,TEMP5 ;,TEMP3,TEMP4 **RFU**
3695 021566 013746 006530                MOV  TEMP5,-(SP)
3696 021572 013746 006522                MOV  TEMP2,-(SP)
3697 021576 013746 006520                MOV  TEMP1,-(SP)
3698 021602 013746 006516                MOV  TEMP,-(SP)
3699 021606 012746 013645                MOV  #SHF1,-(SP)
3700 021612 012746 000005                MOV  #5,-(SP)
3701 021616 010600                MOV  SP,R0
3702 021620 104416                TRAP C$PNTS
3703 021622 062706 000014                ADD  #14,SP
3704 021626 000207                RTS   PC              ;RETURN
3705
3706
```

```

3707 .SBTTL TRVERSE COMMAND LINE SUBROUTINES
3708
3709
3710 :++
3711 : P$TRV SUBROUTINE
3712 :
3713 : PARSE THE COMMAND LINE SUBROUTINE
3714 : TAKE ACTIONS (VIA ACTION TREE) AS PARSING LINE
3715 : PARSING DIRECTIONS FROM 'CLI PARSING NODES'
3716 : REGS USED:
3717 :
3718 : R1,R5=SCRATCH P$NUM=NUMERIC CODE FROM DATA
3719 : R2=ACTION CODE PARAMETER FROM TREE
3720 : R3=PARSE TREE POINTER
3721 : R4=INPUT STRING POINTER
3722 : CALLING SEQUENCE:
3723 : JSR PC,P$TRV
3724 :--
3725 P$TRV:
3726 021630 MOV P$BUFA,R4
3727 021634 013704 003114 MOV P$TREE,R3
3728 021634 013703 003116 P$TR5: TSTB (R4) ;SEE IF ANY CHARS LEFT IN INPUT STRING
3729 021640 105714 BEQ P$EXIT ;BR IF NO
3730 021644 121327 000013 CMPB (R3),#11. ;SEE IF SPECIAL CLI CHAR CODE OR ASCII
3731 021650 003023 BGT 20$ ;BR IF REGULAR ASCII CHAR.
3732 021652 111305 MOVB (R3),R5 ;GET SPECIAL CHAR CODE INTO R5
3733 021654 006305 ASL R5
3734 021656 016505 021672 MOV 10$(R5),R5 ;BUILD TRAVERSE ROUTINE ADDRESS
3735 021662 062705 021672 ADD #10$,R5
3736 021666 004715 JSR PC,(R5) ;JSR TO SPECIAL CLI TRAVERSE ROUTINE
3737 021670 000763 BR P$TR5 ;GO SEE IF MORE OF STRING LEFT
3738
3739
3740 021672 000114 10$: .WORD TRVERR-10$ ;TRAVERSE TABLE FOR 'CLI FUNTIONS'
3741 021674 000134 .WORD TRVEXI-10$ ;1
3742 021676 000152 .WORD TRVBR-10$ ;2
3743 021700 000162 .WORD TRVBIF-10$ ;3
3744 021702 000204 .WORD TRVSPA-10$ ;4
3745 021704 000270 .WORD TRVNUM-10$ ;5
3746 021706 000604 .WORD TRVALP-10$ ;6
3747 021710 000650 .WORD TRVALN-10$ ;7
3748 021712 000270 .WORD TRVOCT-10$ ;8
3749 021714 000256 .WORD TRVDEC-10$ ;9
3750 021716 000736 .WORD TRVSTR-10$ ;10
3751
3752 ;NOT A SPECIAL CODE
3753
3754 021720 121314 20$: CMPB (R3),(R4) ;SEE IF FIRST CHAR OF STRING IS A MATCH
3755 021722 001403 BEQ 22$ ;BR IF A MATCH
3756 021724 004737 021770 JSR PC,TRVBRC ;IF NOT A MATCH, GO TAKE MISS BRANCH
3757 021730 000743 BR P$TR5 ; THEN GO BACK PT'G TO MISS NODE
3758 021732 004737 021750 22$: JSR PC,TRVACT ;IF A MATCH, GO DO ACTION DEFINED BY
3759 021736 062703 000004 ADD #4,R3 ; ACTION CODE IN CLI NODE, THEN
3760 ; ADJUST PTR TO NEXT CLI NODE
3761 021742 005204 INC R4 ;ADJUST BUF PTR TO NEXT CHAR IF MATCH
3762 021744 000735 BR P$TR5

```



```

3763
3764 021746 000207 P$EXIT: RTS PC ;RETURN FROM PARSER
3765
3766 ;-----
3767
3768 ;GOTO USER ACTION ROUTINE
3769 021750 116302 000001 TRVACT: MOV 1(R3),R2 ;GET ACTION CODE FROM CLI NODE
3770 021754 042702 177400 BIC #177400,R2 ;CLEAR ANY SIGN EXTENSION
3771 021760 013705 003120 MOV P$ACT,R5 ;GET ADDRESS OF CLI ACTION ROUTINE
3772 021764 004715 JSR PC,(R5) ;GO DO ACTION DEFINED BY CODE
3773 021766 000207 RTS PC ;RETURN TO CALLING CODE
3774
3775 ;TAKE BRANCH IN TREE
3776 021770 016305 000002 TRVBRC: MOV 2(R3),R5 ;GET BRANCH DISPLACEMENT FROM TREE
3777 021774 060503 ADD R5,R3 ; AND POINT R3 TO THE 'MISS' NODE
3778 021776 000207 RTS PC ; RETURN TO P$TRV
3779
3780 ;NO BRANCH TAKEN
3781 022000 062703 000004 TRVNOB: ADD #4,R3 ;THINGS OK, UPDATE R3 TO POINT TO NEXT
3782 022004 000207 RTS PC ; NODE AND RETURN TO P$TRV
3783
3784 ;-----
3785 022006 004737 021750 TRVERR: JSR PC,TRVACT ;TAKE ERROR ACTION
3786 022012 112737 177777 003131 MOVB #-1,P$GDBD ;SET ERROR RETURN FLAG
3787 022020 005726 TST (SP)+ ;GET RID OF 'JSR PUSH TO TRVERR'
3788 022022 000137 021746 JMP P$EXIT ;RETURN DIRECT TO EXIT OF P$TRV ROUTINE
3789
3790 022026 004737 021750 TRVEXI: JSR PC,TRVACT ;TAKE EXIT ACTION
3791 022032 105037 003131 CLRB P$GDBD ;SET GOOD/BAD FLAG TO 'SUCCESS (0)'
3792 022036 005726 TST (SP)+ ;GET RID OF 'JSR PUSH TO TRVEXI'
3793 022040 000137 021746 JMP P$EXIT ;RETURN DIRECT TO EXIT OF P$TRV ROUTINE
3794
3795 022044 004737 021750 TRVBR: JSR PC,TRVACT ;GO TAKE BRANCH ACTION
3796 022050 000137 021770 JMP TRVBRC
3797
3798 022054 004737 021750 TRVBIF: JSR PC,TRVACT
3799 022060 105737 003131 TSTB P$GDBD ;SEE IF P$GDBD SET OR CLEARED BY ACTION
3800 022064 001402 BEQ 1$ ;IF CLEAR FALL THRU TO NEXT NODE
3801 022066 000137 021770 JMP TRVBRC ;ELSE TAKE THE 'MISS' BRANCH
3802 022072 000137 022000 1$: JMP TRVNOB ;JUST UPDATE TO NEXT NODE IF THINGS OK
3803
3804 022076 005005 TRVSPA: CLR R5 ;CLEAR "SPACE OR TAB FOUND" FLAG
3805 022100 121427 000011 1$: CMPB (R4),#11 ;SEE IF CHAR. IN CMD LINE= TAB
3806 022104 001003 BNE 2$ ;BR IF NO, NOT A TAB
3807 022106 005204 INC R4 ;INC INPUT STRING POINTER
3808 022110 005205 INC R5 ;INDICATE A TAB FOUND
3809 022112 000772 BR 1$ ;GO CHECK NEXT CHAR
3810
3811 022114 121427 000040 2$: CMPB (R4),#40 ;SEE IF CHAR. IN CMD LINE= SPACE
3812 022120 001003 BNE 10$ ;BR IF NO, NON-SPACE OR NON-TAB CHAR.
3813 022122 005204 INC R4 ;INC INPUT STRING POINTER
3814 022124 005205 INC R5 ;INDICATE A SPACE FOUND
3815 022126 000764 BR 1$ ;GO CHECK NEXT CHAR
3816 022130 005705 10$: TST R5 ;SEE IF ANY SPACES OR TABS FOUND
3817 022132 001404 BEQ 15$ ;BR IF NO, TAKE NO ACTION
3818 022134 004737 021750 JSR PC,TRVACT ;GO TAKE ACTION IF ANY FOUND

```

```

3819 022140 000137 022000          JMP      TRVNOB          ;JUST GO UPDATE R3 TO NEXT NODE IF OK
3820 022144 000137 021770    15$:  JMP      TRVBRC          ;TAKE BRANCH (MISS) IF NONE FOUND
3821
3822
3823 022150 012737 000012 003126 TRVDEC: MOV      #10.,P$RADX      ;USE DECIMAL AS RADIX AND ASSUME +
3824 022156 000137 022170          JMP      TRVNMA
3825 022162          TRVOCT: ;(SAME AS TRVNUM SINCE DEFAULT RADIX IS OCTAL)
3826 022162 012737 000010 003126 TRVNUM: MOV      #8.,P$RADX      ;USE OCTAL AS RADIX AND ASSUME +
3827 022170 005005          TRVNMA: CLR      R5          ;CLEAR DIGIT COUNTER
3828 022172 121427 000053          CMPB     (R4),#'+'      ;SEE IF THERE'S A + SIGN THERE
3829 022176 001001          BNE     10$           ; BR IF NO
3830 022200 000406          BR      11$           ; ELSE P$RADX ALREADY SAYS +, JUST BR
3831 022202 121427 000055    10$:  CMPB     (R4),#'-'      ;SEE IF THERE'S A - SIGN THERE
3832 022206 001004          BNE     11$           ; BR IF NO
3833 022210 112737 177777 003127 MOVB     #-1,P$RADX+1    ;SET 'MINUS FLAG' (HI BYTE OF P$RADX)
3834 022216 005204    11$:  INC      R4          ;BUMP R4 TO POINT TO FIRST CHAR
3835
3836 022220 121427 000060    1$:  CMPB     (R4),#60      ;SEE IF CHAR. LESS THAN A '0'
3837 022224 002434          BLT     2$           ;BR IF YES (NOT NUMERIC)
3838 022226 121427 000067          CMPB     (R4),#67      ;SEE IF CHAR. GREATER THAN A '7'
3839 022232 003426          BLE     13$          ; BR IF YES
3840 022234 123727 003126 000012 CMPB     P$RADX,#10.    ;SEE IF IN DECIMAL MODE
3841 022242 001417          BEQ     12$          ; BR IF YES (CAN USE HIGHER LIMIT)
3842 022244 121427 000071          CMPB     (R4),#71      ;SEE IF DIGIT WAS A 8 OR 9
3843 022250 003022          BGT     2$           ;BR IF NON-NUMERIC
3844 022252          PRINTF #CLIBRX      ;ELSE WAS A 8 OR 9 WHEN IN OCTAL RADIX
3845 022252 012746 011571          MOV     #CLIBRX,-(SP)
3846 022256 012746 000001          MOV     #1,-(SP)
3847 022262 010600          MOV     SP,R0
3848 022264 104417          TRAP   C$PNTF
3849 022266 062706 000004          ADD     #4,SP
3850 022272 112737 177777 003131 MOVB     #-1,P$GDBD    ;SET ERROR RETURN FLAG
3851 022300 000474          BR      5$           ; PRINT ERROR AND TAKE MISS
3852
3853 022302 121427 000071    12$:  CMPB     (R4),#71      ;SEE IF CHAR. GREATER THAN A '9'
3854 022306 003003          BGT     2$           ;BR IF YES (NOT NUMERIC)
3855 022310 005204    13$:  INC      R4          ;UPDATE CMD LINE PTR TO NEXT CHAR.
3856 022312 005205          INC      R5          ;INDICATE A NUMERIC FOUND
3857 022314 000741          BR      1$           ;GO LOOK AT NEXT CHAR.
3858
3859 022316 005705    2$:  TST     R5          ;SEE IF FOUND ANY NUMERICS
3860 022320 001464          BEG     5$           ;BR IF NO, TAKE 'MISS' BRANCH
3861 022322 010401          MOV     R4,R1        ;GET POINTER TO START OF NUMERIC STRING
3862 022324 160501          SUB     R5,R1
3863 022326 005037 003124          CLR     P$NUM        ;CLEAR LOC. WHERE VALUE WILL BE STORED
3864 022332 112102    3$:  MOVB     (R1)+,R2      ;GET ASCII CHAR AND CONVERT IT TO A #
3865 022334 162702 000060          SUB     #60,R2
3866 022340 006337 003124          ASL     P$NUM        ;SHIFT CURRENT VALUE TO MAKE ROOM
3867 022344 103437          BCS     7$           ;ERROR IF NUMBER TOO BIG
3868 022346 013737 003124 003122 MOV     P$NUM,P$CNT    ;SAVE FOR LATER IN CASE DECIMAL RADIX
3869 022354 006337 003124          ASL     P$NUM
3870 022360 103431          BCS     7$           ;ERROR IF NUMBER TOO BIG
3871 022362 006337 003124          ASL     P$NUM
3872 022366 103426          BCS     7$           ;ERROR IF NUMBER TOO BIG
3873 022370 123727 003126 000012 CMPB     P$RADX,#10.    ;SEE IF DECIMAL RADIX
3874 022376 001004          BNE     4$           ;BR IF NOT EQUAL

```



```

3875 022400 063737 003122 003124      ADD    P$CNT,P$NUM
3876 022406 103416                      BCS    7$
3877 022410 060237 003124      4$:   ADD    R2,P$NUM
3878 022414 103413                      BCS    7$
3879 022416 005305                      DEC    R5
3880 022420 001344                      BNE    3$
3881 022422 105737 003127      TSTB   P$RADX+1
3882 022426 001402                      BEQ    15$
3883 022430 005437 003124      NEG    P$NUM
3884 022434 004737 021750      15$:  JSR    PC,TRVACT
3885 022440 000137 022000      JMP    TRVNOB
3886
3887 022444      7$:   PRINTF #CLINBG
3888 022444 012746 011547                      MOV    #CLINBG,-(SP)
3889 022450 012746 000001                      MOV    #1,-(SP)
3890 022454 010600                      MOV    SP,R0
3891 022456 104417                      TRAP   C$PNTF
3892 022460 062706 000004                      ADD    #4,SP
3893 022464 112737 177777 003131      MOVB   #-1,P$GDBD
3894 022472 000137 021770      5$:   JMP    TRVBRC
3895
3896
3897 022476 005005      TRVALP: CLR    R5
3898 022500 121427 000101      1$:   CMPB   (R4),#101
3899 022504 002406                      BLT    2$
3900 022506 121427 000132      CMPB   (R4),#132
3901 022512 003003                      BGT    2$
3902 022514 005204                      INC    R4
3903 022516 005205                      INC    R5
3904 022520 000767                      BR     1$
3905 022522 005705      2$:   TST    R5
3906 022524 001404                      BEQ    3$
3907 022526 004737 021750      JSR    PC,TRVACT
3908 022532 000137 022000      JMP    TRVNOB
3909 022536 000137 021770      3$:   JMP    TRVBRC
3910
3911 022542 005005      TRVALN: CLR    R5
3912 022544 121427 000060      10$:  CMPB   (R4),#60
3913 022550 002417                      BLT    2$
3914 022552 121427 000072      CMPB   (R4),#72
3915 022556 003003                      BGT    1$
3916 022560 005204                      INC    R4
3917 022562 005205                      INC    R5
3918 022564 000767                      BR     10$
3919 022566 121427 000101      1$:   CMPB   (R4),#101
3920 022572 002406                      BLT    2$
3921 022574 121427 000132      CMPB   (R4),#132
3922 022600 003003                      BGT    2$
3923 022602 005204                      INC    R4
3924 022604 005205                      INC    R5
3925 022606 000756                      BR     10$
3926 022610 005705      2$:   TST    R5
3927 022612 001404                      BEQ    3$
3928 022614 004737 021750      JSR    PC,TRVACT
3929 022620 000137 022000      JMP    TRVNOB
3930 022624 000137 021770      3$:   JMP    TRVBRC

```



```

3931
3932
3933
3934 022630 010401          TRVSTR: MOV      R4,R1          ;POINT R1 TO CMD STRING
3935 022632 010305          MOV      R3,R5
3936 022634 062705 000006   ADD      #6,R5          ;POINT R5 TO MATCH STRING FROM CLI NODE
3937 022640 005037 003122   CLR      P%CNT         ;CLEAR CHAR MATCH COUNT
3938 022644 105715          2$: TSTB   (R5)         ;SEE IF END OF MATCH STRING YET
3939 022646 001411          BEQ     10$            ;BR IF YES
3940 022650 105711          TSTB   (R1)         ;SEE IF END OF CMD LINE YET
3941 022652 001407          BEQ     10$            ;BR IF YES
3942 022654 121115          CMPB   (R1),(R5)     ;SEE IF CHARACTERS MATCH
3943 022656 001005          BNE     10$            ;BR IF NO
3944 022660 005237 003122   INC     P%CNT         ;MATCH -INCREMENT MATCH COUNT
3945 022664 005201          INC     R1            ;UPDATE STRING POINTERS
3946 022666 005205          INC     R5
3947 022670 000765          BR      2$            ;BR TO CONTINUE CHECKING CHARS.
3948
3949 022672 005737 003122   10$: TST   P%CNT         ;WHEN DONE SEE IF ANY MATCHES FOUND
3950 022676 001406          BEQ     15$            ;BR IF NO, GO TAKE THE MISS BRANCH
3951 022700 010104          MOV     R1,R4         ;POINT CMD POINTER TO END OF STRING &
3952 022702 004737 021750   JSR    PC,TRVACT     ;IF A MATCH FOUND, GO DO MATCH ACTION
3953 022706 066303 000004   ADD     4(R3),R3     ;UPDATE R3 TO NEXT NODE (NO BRANCH)
3954 022712 000207          RTS     PC            ; (NO RETURN THRU TRVNOB SINCE DIFFERNT
3955                                     ; DISPLACEMENT DUE TO MATCH STRING)
3956 022714 000137 021770   15$: JMP   TRVBRC     ; GO TAKE BRANCH
3957
3958                                     ; (PARSED OK), -1 IF ILL CMD.....
3959 -----
3960

```

.SBTTL REPORT CODING SECTION

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

3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981

022720
022720

022720 004737 017676

022724
022724
022724 104425

BGNRPT

JSR PC,REPORT

ENDRPT

L\$RPT::

;CALL SUBROUTINE TO DUMP EVENT LOG
; AND BASE TABLE

L10010: TRAP C\$RPT

3982
3983
3984
3985
3986
3987
3988
3989 022726
3990 022726
3991
3992 022726 177777
3993 022730 177777
3994 022732 177777
3995
3996 022734
3997

.SBTTL PROTECTION TABLE

:++
: THIS TABLE IS USED BY THE RUNTIME SERVICES
: TO PROTECT THE LOAD MEDIA.
:--

BGNPROT

L\$PROT::

-1 ;OFFSET INTO P-TABLE FOR CSR ADDRESS
-1 ;OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
-1 ;OFFSET INTO P-TABLE FOR DRIVE NUMBER

ENDPROT

3998
3999
4000
4001
4002
4003
4004
4005
4006
4007
4008
4009
4010
4011
4012
4013
4014
4015
4016
4017
4018
4019
4020
4021
4022
4023
4024
4025
4026
4027
4028
4029
4030
4031
4032
4033
4034
4035
4036
4037
4038
4039
4040
4041
4042
4043
4044
4045
4046
4047
4048
4049
4050
4051
4052
4053

022734
022734

022734 012737 177777 006544
022742
022742 012700 000040
022746 104447
022750
022750 103417
022752
022752 012700 000037
022756 104447
022760
022760 103513
022762
022762 012700 000036
022766 104447
022770
022770 103002
022772 000137 023442
022776
022776 012700 000035
023002 104447
023004
023004 103521
023006 000523

023010 005037 006544
023014 005037 006604

023020 012702 006600
023024
023024 012700 000114
023030 104462
023032 010001
023034
023034 103006
023036 004737 017016
023042 012737 000100 006610
023050 000457

023052
023052 012700 000120
023056 104462
023060 010001
023062
023062 103017
023064 004737 017016

```
.SBTTL INITIALIZE SECTION

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

BGNINIT

LSINIT::

MOV #-1,RESFLG ;SET RESTART FLAG
REDEF #EF.START ;IF HERE CAUSE OF START,DO SOME INIT
                                MOV #EF.START,RO
                                TRAP CSREFG

BCOMPLETE START
REDEF #EF.RESTART ;IF HERE CAUSE OF RESTART, DO SOME INIT
                                MOV #EF.RESTART,RO
                                TRAP CSREFG

BCOMPLETE RESTRT
REDEF #EF.CONTINUE ;SEE IF WE'RE HERE CAUSE OF A CONTINUE
                                MOV #EF.CONTINUE,RO
                                TRAP CSREFG

BNCOMPLETE S1 ;BR IF NOT HERE CAUSE OF CONITNUE
                                BCC S1
                                JMP ENDIT ;JMP IF HERE CAUSE OF A CONTINUE
S1: REDEF #EF.NEW ;SEE IF THIS IS A 'NEW PASS'
                                MOV #EF.NEW,RO
                                TRAP CSREFG
BCOMPLETE NEW ;IF YES, BR AROUND LOGUNIT # SETUP
                                BCS NEW
BR GETPRM

START: CLR RESFLG ;CLEAR RESTART FLAG SINCE HERE ON START
CLR CLKVEC ;CLEAR CLK VECTOR PTR. AS A FLAG IN
; NO CLOCK IS FOUND.
MOV #CLKCSR,R2 ;SETUP R2 AS A PTR. TO CLOCK INFO BLOCK
CLOCK L,R1 ;LOOK FOR A LINE CLOCK
                                MOV #'L,RO
                                TRAP C$CLK
                                MOV RO,R1
BNCOMPLETE S2 ; IF NONE THERE GO LOOK FOR A P-CLOCK
                                BCC S2
                                JSR PC,CLKSET ; GO SET UP CLOCK INFO TABLE & CLK VEC.
MOV #LCLKEN,CLKEN ;SETUP THE ENABLE LINE CLOCK DATA
BR RESTRT

S2: CLOCK P,R1 ;LOOK FOR A P-CLOCK SINCE NO LINE CLOCK
                                MOV #'P,RO
                                TRAP C$CLK
                                MOV RO,R1
BNCOMPLETE S3 ; IF NONE THERE GO SEE IF THIS IS LSI
                                BCC S3
JSR PC,CLKSET ; ELSE GO SET UP CLOCK INFO & VECTOR
```

```

4054 023070 062737 000002 006600      ADD      #2,CLKCSR      ;POINT CLKCSR TO P-CLK COUNT SET REG.
4055 023076 012777 001600 163474      MOV      #PCLKCT,@CLKCSR ;LOAD CLK SET REG. WITH COUNT VALUE
4056 023104 162737 000002 006600      SUB      #2,CLKCSR      ;POINT CLKCSR BAC TO P-CLK CSR
4057 023112 012737 000111 006610      MOV      #PCLKEN,CLKEN  ;SETUP THE ENABLE THE P-CLK DATA
4058 023120 000433                          BR      RESTRT
4059
4060 023122                          S3:    READBUS          ;READ BUS TYPE TO SEE IF ON AN LSI
4061 023122 104407                          ;TRAP   CSRDBU
4062 023124                          BNCOMPLETE      S4      ;BR IF NOT, NO CHANCE OF A CLOCK
4063 023124 103021                          ;BCC   S4
4064 023126 012737 000100 006604      MOV      #100,CLKVEC    ;LOAD 100 AS CLK VECTOR
4065 023134 005037 006602                          CLR      CLKBR          ;LOAD 0 AS CLK INT. LEVEL
4066 023140 012737 006610 006600      MOV      #CLKEN,CLKCSR ;KLUDGE UP THE CSR & ENABLE DATA LOCS
4067 023146                          GMANID  L5060,CLKHZ,D,377,50.,60.,YES
4068 023146 104443                          ;TRAP   CS$GMAN
4069 023150 000406                          BR      10000$
4070 023152 006606                          .WORD  CLKHZ
4071 023154 000052                          .WORD  T$CODE
4072 023156 013172                          .WORD  L5060
4073 023160 000377                          .WORD  377
4074 023162 000062                          .WORD  T$LOLIM
4075 023164 000074                          .WORD  T$HILIM
4076 023166
4077 023166 000410                          BR      RESTRT          10000$:
4078
4079 023170                          S4:    PRINTF #BDCLK    ;INFORM OPR. NO CLOCK, & EXIT INIT
4080 023170 012746 013310                          MOV      #BDCLK,-(SP)
4081 023174 012746 000001                          MOV      #1,-(SP)
4082 023200 010600                          MOV      SP,RO
4083 023202 104417                          TRAP   C$PNTF
4084 023204 062706 000004                          ADD      #4,SP
4085
4086 023210 005037 006612      RESTRT: CLR      TIMMIN    ;CLEAR TIME SINCE START LOCATIONS
4087 023214 005037 006614      CLR      TIMSEC
4088 023220 013737 006606 006616      MOV      CLKHZ,TIMTCK    ;LOAD TICKS/SEC
4089 023226 012702 006630      MOV      #EVTLOG,R2     ;INIT EVENT TABLE TO ALL 1'S AFTER EACH
4090 023232 010237 006626      MOV      R2,EVTPTR     ; START OR RES AND INIT TABLE POINTER
4091 023236 012722 177777      1$:    MOV      #-1,(R2)+
4092 023242 020227 007532      CMP      R2,#EVTEND    ;SEE IF REACHED END OF TABLE
4093 023246 001373      BNE     1$             ;LOOP UNTIL DONE
4094
4095 023250 012737 177777 006540      NEW:    MOV      #-1,LOGUNT ;INITIALIZE LOGICAL UNIT #
4096
4097 023256 005237 006540      GETPRM: INC      LOGUNT    ;POINT TO NEXT LOGICAL UNIT
4098 023262 023737 006540 002012      CMP      LOGUNT,L$UNIT  ;SEE IF PAST MAX. LOG. UNIT #
4099 023270 002367      BGE     NEW           ;BR IF YES, AND START OVER
4100
4101 023272                          GPHARD  LOGUNT,R1      ;GET THE P-TABLE FOR THIS LOG. UNIT
4102 023272 013700 006540                          MOV      LOGUNT,RO
4103 023276 104442                          TRAP   C$GPHRD
4104 023300 010001                          MOV      RO,R1
4105 023302                          BNCOMPLETE      GETPRM ;IF NO P-TABLE AVAIL., GO GET NEXT ONE
4106 023302 103365                          BCC     GETPRM
4107
4108 023304 011137 006552      MOV      (R1),FHDPLX    ;PUT FULL OR HALF DUPLEX ANSWER IN LOC.
4109

```



```

4110
4111 ;DEVICE DEPENDENT PART OF GETTING INFO FROM P-TABLE
4112
4113 023310 016137 000002 011344 MOV 2(R1),RXCSR ;STORE AWAY CSR ADDRESSES
4114
4115
4116 023316 016137 000002 011346 MOV 2(R1),PCSAR
4117 023324 062737 000002 011346 ADD #2,PCSAR
4118 023332 016137 000002 011350 MOV 2(R1),RDSR
4119 023340 062737 000002 011350 ADD #2,RDSR
4120 023346 016137 000002 011352 MOV 2(R1),TXCSR
4121 023354 062737 000004 011352 ADD #4,TXCSR
4122 023362 016137 000002 011354 MOV 2(R1),TDSR
4123 023370 062737 000006 011354 ADD #6,TDSR
4124
4125 023376 016137 000004 011356 MOV 4(R1),INVEC ;STORE AWAY INPUT INTERRUPT VECTOR
4126 023404 016137 000004 011360 MOV 4(R1),OUTVEC
4127 023412 062737 000004 011360 ADD #4,OUTVEC ;BUILD OUTPUT INTERRUPT VECTOR
4128 023420 016137 000006 011362 MOV 6(R1),INTPRI ;STORE AWAY INTERRUPT PRIORITY
4129 023426 016137 000012 011364 MOV 12(R1),OPTYP ;STORE AWAY DEVICE OPTION TYPE
4130 023434 016137 000014 011420 MOV 14(R1),RNODE ;STORE AWAY THE REMOTE NODE TYPE
4131 023442
4132 023442 ENDIT: SETVEC CLKVEC,#CLKINT,#340 ;SETUP CLOCK VECTOR
4133 023442 012746 000340 MOV #340,-(SP)
4134 023446 012746 017042 MOV #CLKINT,-(SP)
4135 023452 013746 006604 MOV CLKVEC,-(SP)
4136 023456 012746 000003 MOV #3,-(SP)
4137 023462 104437 TRAP C$SVEC
4138 023464 062706 000010 ADD #10,SP
4139
4140 ;DEVICE DEPENDENT VECTOR SETUP
4141
4142 023470 SETVEC INVEC,#DVRXI,#PRI04 ;SETUP INPUT INTERRUPT VECTOR
4143 023470 012746 000200 MOV #PRI04,-(SP)
4144 023474 012746 032472 MOV #DVRXI,-(SP)
4145 023500 013746 011356 MOV INVEC,-(SP)
4146 023504 012746 000003 MOV #3,-(SP)
4147 023510 104437 TRAP C$SVEC
4148 023512 062706 000010 ADD #10,SP
4149 023516 SETVEC OUTVEC,#DVTXI,#PRI04 ;SETUP OUTPUT INTERRUPT VECTOR
4150 023516 012746 000200 MOV #PRI04,-(SP)
4151 023522 012746 033166 MOV #DVTXI,-(SP)
4152 023526 013746 011360 MOV OUTVEC,-(SP)
4153 023532 012746 000003 MOV #3,-(SP)
4154 023536 104437 TRAP C$SVEC
4155 023540 062706 000010 ADD #10,SP
4156
4157 023544 SETPRI #PRI00 ;SET THE 'RUN' PRIORITY TO 0
4158 023544 012700 000000 MOV #PRI00,R0
4159 023550 104441 TRAP C$SPRI
4160 023552
4161 023552 104432 TRAP C$EXIT
4162 023554 000002 .WORD L10012-.
4163
4164
4165 .EVEN

```


CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{F 8} 10:29 PAGE 97
INITIALIZE SECTION

SEQ 0096

4166
4167 023556
4168 023556
4169 023556 104411

ENDINIT

L10012: TRAP CSINIT

4170
4171
4172
4173
4174
4175
4176
4177
4178
4179 023560
4180 023560
4181
4182
4183 023560
4184 023560
4185 023560 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

LSAUTO::

ENDAUTO

L10013: TRAP CSAUTO

4186
4187
4188
4189
4190
4191
4192
4193 023562
4194 023562
4195
4196 023562 005077 163012
4197 023566
4198 023566 012700 000340
4199 023572 104441
4200 023574
4201 023574 104433
4202
4203 023576
4204 023576 104432
4205 023600 000002
4206
4207
4208
4209
4210 023602
4211 023602
4212 023602 104412

.SBTTL CLEANUP CODING SECTION

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

BGNCLN

L\$CLEAN::

CLR @CLKCSR
SETPRI #PRI07

;DISABLE CLOCK
;SET PROCESSOR PRIORITY BACK TO 7

MOV #PRI07,R0
TRAP C\$SPRI

BRESET

;CLEAR ALL BEFORE END

TRAP C\$RESET

EXIT CLN

TRAP C\$EXIT
.WORD L10014-

.EVEN

ENDCLN

L10014:

TRAP C\$CLEAN

4213
4214
4215
4216
4217
4218
4219
4220 023604
4221 023604
4222
4223
4224 023604
4225 023604 000167
4226 023606 000000
4227
4228
4229
4230
4231 023610
4232 023610
4233 023610 104453

.SBTTL DROP UNIT SECTION

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

BGNDU

LSDU::

EXIT DU

.WORD JSJMP
.WORD L10015-2-

.EVEN

ENDDU

L10015:
TRAP CSDU

4234
4235
4236
4237
4238
4239
4240
4241
4242 023612
4243 023612
4244
4245
4246 023612
4247 023612 000167
4248 023614 000000
4249
4250
4251
4252
4253 023616
4254 023616
4255 023616 104452
4256
4257

.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

LSAU::

EXIT AU

.WORD JSJMP
.WORD L10016-2-.

.EVEN

ENDAU

L10016: TRAP CSAU


```

4314
4315 024012 013737 006426 006506      MOV      CMPPTR,CPTR      ;SET UP START OF COMPARE POINTER TABLE
4316 024020 013737 006434 006510      MOV      CCURAD,CURADD   ;SET UP CURRENT ADDR. TO START OF CMPBUF
4317 024026 012737 000005 006500      MOV      #5,MSGTYP
4318 024034 013737 002162 006502      MOV      MSG5C,CURCC
4319 024042 004737 021202                JSR      PC,BLDBUF      ;PUT DEFAULT MESSAGE INTO CMPBUF
4320 024046 012737 000001 006430      MOV      #1,CMPTOT      ;BUMP THE COMP MMSG COUNT
4321 024054 012737 000003 006546      MOV      #ACT,MODTYP    ;SET DEFAULT MODE= ACTIVE
4322 024062 005037 006550                CLR      MLTYP          ;SET DEFAULT MAINTENANCE LOOP MODE =NONE
4323 024066 012737 000001 006556      MOV      #1,RPASS       ;SET UP DEFAULT 'RUN PASS' COUNT TO 1
4324 024074 012737 000002 006554      MOV      #2,PARAM       ;SET UP PROG. PARAMETERS - DATACHECKING ENABLED
4325                                     ;
4326 024102                PRINTF  #HLPO          ;
4327 024102 012746 012014                MOV      #HLPO,-(SP)
4328 024106 012746 000001                MOV      #1,-(SP)
4329 024112 010600                MOV      SP,RO
4330 024114 104417                TRAP    C$PNTF
4331 024116 062706 000004                ADD     #4,SP
4332 024122 013737 006546 007662  GTRAS:  MOV      MODTYP,DEV1
4333 024130 013737 006550 007664      MOV      MLTYP,DEV2
4334 024136 013737 006556 007666      MOV      RPASS,DEV3
4335 024144 013737 006554 007670      MOV      PARAM,DEV4
4336 024152 004737 021326      JSR      PC,SHWOP      ;PRINT TO OPERATOR THE CURRENT MODE.....
4337
4338 024156                MANUAL          ;SEE IF MANUAL INTERVENTION ALLOWED
4339 024156 104450                TRAP    C$MANI
4340 024160                BCOMPLETE      GETCL ; BR IF YES (UAM=0 AND NOT CHAINED)
4341 024160 103412                BCS     GETCL
4342 024162 005737 006556      TST     RPASS        ;SEE IF THIS IS FIRST 'DCLT PASS'
4343 024166 001002                BNE     1$          ; BR IF NOT COMPLETED 1 PASS
4344 024170                EXIT     TST        ; IF DONE 1 PASS IN UNATTENDED MODE - EXIT
4345 024170 104432                TRAP    C$EXIT
4346 024172 007510                .WORD  L10017-.
4347 024174 012737 000001 006550  1$:      MOV      #TTL,MLTYP    ;SET UP DEFAULT FOR UNATTENDED MODE
4348 024202 000137 027054      JMP     GTR9          ; 'R M=ACT/LO=1/PAS=1/NOST/CH' AND RUN
4349
4350                .SBTTL          COMMAND LINE FETCH & INTERPRETATION SECTION
4351
4352 024206 105037 003131      GETCL:  CLRB   P$GDBD    ;CLEAR CMD LINE PARSING ERROR FLAGS
4353 024212 105037 003130      CLRB   P$NNUF
4354 024216                GMANID  CLISPM,CMDBUF,A,0,1,72.,NO ;GET A COMMAND LINE FROM OPR.
4355 024216 104443                TRAP    C$GMAN
4356 024220 000406                BR     10000$
4357 024222 002666                .WORD  CMDBUF
4358 024224 000142                .WORD  T$CODE
4359 024226 011466                .WORD  CLISPM
4360 024230 000000                .WORD  0
4361 024232 000001                .WORD  T$LOLIM
4362 024234 000110                .WORD  T$HILIM
4363 024236                10000$:
4364 024236 012737 002666 003114      MOV      #CMDBUF,P$BUFA
4365 024244 012737 007672 003116      MOV      #CLITRE,P$TREE
4366 024252 012737 025130 003120      MOV      #CLIACT,P$ACT
4367 024260 005037 003012                CLR     QUALFG        ;CLEAR QUALIFIER FLAG LOCATION
4368 024264 004737 021630      JSR     PC,P$TRV      ;GO PARSE COMMAND LINE
4369 024270 105737 003131      TSTB   P$GDBD        ;SEE IF PARSED OK OR AN ERROR

```


4426	024570	012746	000002						MOV #2,-(SP)
4427	024574	010600							MOV SP,RO
4428	024576	104417							TRAP C\$PNTF
4429	024600	062706	000006						ADD #6,SP
4430	024604	000137	024206						: THEN GO GET A NEW COMMAND.
4431	024610	006301		17\$:	JMP	GETCL			:# OF MSGS *4 = NEXT FREE PTR BLOCK
4432	024612	006301			ASL	R1			
4433	024614	060137	006424		ASL	R1			
4434	024620	013737	006424	006506	ADD	R1, TXPTR			: SETUP CHAR. COUNT, CURRENT ADDR, & PTR
4435	024626	013737	006450	006510	MOV	TXPTR, CPTR			
4436	024634	004737	021104		MOV	TCURAD, CURADD			: ADD IN CHAR. COUNT AND CHECK TOTAL
4437	024640	004737	021202		JSR	PC, ADDCC			: GO BUILD MESSAGE IN BUFFER AND PTRS.
4438	024644	013737	006506	006424	JSR	PC, BLDBUF			
4439	024652	013737	006512	006446	MOV	CPTR, TXPTR			: UPDATE CHAR. COUNT, CURR ADDR, & PTR
4440	024660	013737	006510	006450	MOV	TOTCC, TTOTCC			
4441	024666	005237	006444		MOV	CURADD, TCURAD			
4442	024672	005337	003014		INC	TXMTOT			: DEC THE COPY COUNT
4443	024676	001270			DEC	QUALVL			
4444	024700	000137	024206		BNE	5\$			
4445					JMP	GETCL			
4446	024704	013737	006432	006512	2\$:	MOV	CTOTCC, TOTCC		: SETUP CHAR. COUNT, CURR. ADDR. & PTR
4447	024712	023727	006512	001000	CMP	TOTCC, #BUFLIM			: SEE IF BUFFER ALREADY FULL
4448	024720	002414			BLT	16\$: BR IF NOT FULL (BUFLIM # OF CHARS.)
4449	024722				PRINTF	#MSGTRN, #BUFEX			: ELSE TELL OPR. AND DON'T BUILD MSG.
4450	024722	012746	013455						MOV #BUFEX, -(SP)
4451	024726	012746	013473						MOV #MSGTRN, -(SP)
4452	024732	012746	000002						MOV #2, -(SP)
4453	024736	010600							MOV SP,RO
4454	024740	104417							TRAP C\$PNTF
4455	024742	062706	000006						ADD #6,SP
4456	024746	000137	024206						: THEN GO GET A NEW COMMAND
4457	024752	005737	006432	16\$:	JMP	GETCL			: IF FIRST "SET" THEN GET RID OF DEFAULT
4458	024756	001002			TST	CTOTCC			
4459	024760	005037	006430		BNE	7\$			
4460	024764			7\$:	CLR	CMPTOT			
4461	024764	012737	006226	006426	MOV	#PTR13, CMPPTR			: INIT COMPARE MESSAGE POINTER
4462	024772	013701	006430		MOV	CMPTOT, R1			
4463	024776	020127	000017		CMP	R1, #MSGLIM			: SEE IF MSG COUNT EXCEEDED.
4464	025002	002414			BLT	18\$: BR IF NO
4465	025004				PRINTF	#MSGTRN, #TABEX			: ELSE TELL ORR. AND DON'T BUILD MSG.
4466	025004	012746	013415						MOV #TABEX, -(SP)
4467	025010	012746	013473						MOV #MSGTRN, -(SP)
4468	025014	012746	000002						MOV #2, -(SP)
4469	025020	010600							MOV SP,RO
4470	025022	104417							TRAP C\$PNTF
4471	025024	062706	000006						ADD #6,SP
4472	025030	000137	024206						: THEN GO GET A NEW COMMAND.
4473	025034	006301		18\$:	JMP	GETCL			:# OF MSGS *4 = NEXT FREE PTR BLOCK
4474	025036	006301			ASL	R1			
4475	025040	060137	006426		ASL	R1			
4476	025044	013737	006426	006506	ADD	R1, CMPPTR			
4477	025052	013737	006434	006510	MOV	CMPPTR, CPTR			
4478	025060	004737	021104		MOV	CCURAD, CURADD			: ADD IN XHAR. COUNT AND CHECK TOTAL
4479	025064	004737	021202		JSR	PC, ADDCC			
4480	025070	013737	006506	006426	JSR	PC, BLDBUF			
4481	025076	005237	006430		MOV	CPTR, CMPPTR			
					INC	CMPTOT			

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 106
COMMAND LINE FETCH & INTERPRETATION SECTION

SEQ 0105

4482 025102 013737 006510 006434
4483 025110 013737 006512 006432
4484 025116 005337 003014
4485 025122 001270
4486 025124 000137 024206
4487
4488
4489
4490
4491

MOV CURADD,CCURAD
MOV TOTCC,CTOTCC
DEC QUALVL
BNE 2\$
JMP GETCL

;UPDATE CHAR. COUNT, CURR ADDR. & PTR
;IF COPY WAS GIVEN, PUT MSG IN BUFF
; AGAIN
;GO BACK UNTIL GET A 'RUN'

4442
4493
4494
4495
4496 025130
4497 025130 006302
4498 025132 016202 025146
4499 025136 062702 025146
4500 025142 004712
4501 025144 000207
4502
4503 025146 000144
4504 025150 000146
4505 025152 000156
4506 025154 001476
4507 025156 000246
4508 025160 000166
4509 025162 000272
4510 025164 000364
4511 025166 000706
4512 025170 000716
4513 025172 000734
4514 025174 000744
4515 025176 000754
4516 025200 001046
4517 025202 001504
4518 025204 001066
4519 025206 001146
4520 025210 001154
4521 025212 001164
4522 025214 001174
4523 025216 001204
4524 025220 001214
4525 025222 001232
4526 025224 001262
4527 025226 001272
4528 025230 001312
4529 025232 001320
4530 025234 001330
4531 025236 001340
4532 025240 001350
4533 025242 001376
4534 025244 001406
4535 025246 001512
4536 025250 001526
4537 025252 001560
4538 025254 001570
4539 025256 001600
4540 025260 001610
4541 025262 001620
4542 025264 001630
4543 025266 000136
4544 025270 001124
4545 025272 000642
4546 025274 000672
4547 025276 000664

.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
MOV 10\$(R2),R2 ;FORM ADDRESS OF ACTION ROUTINE
ADD #10\$,R2
JSR PC,(R2)
RTS PC

10\$: .WORD ACTNUL-10\$
.WORD ACTCLR-10\$
.WORD ACTSHO-10\$
.WORD ACTCHK-10\$
.WORD ACTRUN-10\$
.WORD ACTHLP-10\$
.WORD ACTCSE-10\$
.WORD ACTCST-10\$
.WORD ACTSTE-10\$
.WORD ACTSTT-10\$
.WORD ACTSZL-10\$
.WORD ACTCOP-10\$
.WORD ACTNUM-10\$
.WORD ACTOPM-10\$
.WORD ACTSTS-10\$
.WORD ACTEQO-10\$
.WORD ACTMSO-10\$
.WORD ACTMS1-10\$
.WORD ACTMS2-10\$
.WORD ACTMS3-10\$
.WORD ACTMS4-10\$
.WORD ACTMS5-10\$
.WORD ACTMS6-10\$
.WORD ACTATV-10\$
.WORD ACTPAS-10\$
.WORD ACTREC-10\$
.WORD ACTLIS-10\$
.WORD ACTDLL-10\$
.WORD ACTTRA-10\$
.WORD ACTTAL-10\$
.WORD ACTNO-10\$
.WORD ACTECH-10\$
.WORD ACTCRC-10\$
.WORD ACTPRO-10\$
.WORD ACTRPS-10\$
.WORD ACTMOP-10\$
.WORD ACTTLP-10\$
.WORD ACTCLP-10\$
.WORD ACTLLP-10\$
.WORD ACTRLP-10\$
.WORD ACTNUF-10\$
.WORD ACTBCR-10\$
.WORD ACTDMS-10\$
.WORD ACTDME-10\$
.WORD ACTDMQ-10\$

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{D 9} 10:29 PAGE 108
ACTION TABLE AND ROUTINES

SEQ 0107

4548 025300 000232
4549 025302 001520
4550

.WORD ACTPRT-10\$
.WORD ACTMOS-10\$


```

4552 025304 112737 177777 003130 ACTNUF: MOVB #-1,PSNNUF ;SET FLAG TO SAY NEED MORE OF COMMAND
4553 025312 000207 ACTNUL: RTS PC ;RETURN TO PARSER
4554
4555 025314 012737 000001 003010 ACTCLR: MOV #CLEAR,KEYWD1 ;SET LOC TO SAY A CLEAR WAS TYPED
4556 025322 000207 RTS PC
4557
4558 025324 012737 000002 003010 ACTSHO: MOV #SHOW,KEYWD1 ;SET LOC. TO SAY A SHOW WAS TYPED
4559 025332 000207 RTS PC
4560
4561 025334 012702 003016 ACTHLP: MOV #HLPTAB,R2 ;SETUP R2 AS A POINTER TO HELP MSG TABLE
4562 025340 1$: PRINTF #HLPF,(R2)+ ;PRINT HELP INFORMATION MESSAGES
4563 025340 012246 MOV (R2)+,-(SP)
4564 025342 012746 012072 MOV #HLPF,-(SP)
4565 025346 012746 000002 MOV #2,-(SP)
4566 025352 010600 MOV SP,R0
4567 025354 104417 TRAP C$PNTF
4568 025356 062706 000006 ADD #6,SP
4569 025362 020227 003034 CMP R2,#HLPEND ;SEE IF ALL INFO PRINTED YET
4570 025366 001364 BNE 1$ ;IF NO KEEP PRINTING
4571 025370 012737 000005 003010 MOV #HLP,KEYWD1 ;SET LOC. TO SAY A HELP WAS TYPED
4572 025376 000207 RTS PC
4573
4574 025400 012737 000055 003010 ACTPRT: MOV #PRNT,KEYWD1 ;SET LOC. TO SAY A HELP WAS TYPED
4575 025406 004737 017676 JSR PC,REPORT ;CALL ROUTINE TO PRINT EVENT LOG AND BASE TABLE
4576 025412 000207 RTS PC
4577
4578 025414 012737 000004 003010 ACTRUN: MOV #RUN,KEYWD1 ;SET RUN FLAG
4579 025422 112737 177777 003130 MOVB #-1,PSNNUF ;SET FLAG TO SAY NEED MORE OF COMMAND
4580 025430 012737 000001 006556 MOV #1,RPASS ;SET DEFAULT RUN 'PASS' TO 1
4581 025436 000207 RTS PC
4582
4583 025440 012737 006226 006426 ACTCSE: MOV #PTR13,CMPPTR ;INIT COMPARE MESSAGE POINTER
4584 025446 013701 006426 MOV CMPPTR,R1
4585
4586 025452 013702 006430 MOV CMPTOT,R2
4587 025456 105037 003130 CLRB PSNNUF ;FLAG THAT HAVE VALID COMMAND AT THIS PT.
4588 025462 023727 003010 000002 CMP KEYWD1,#SHOW ;SEE IF A CLEAR OR SHOW WAS TYPED
4589 025470 001471 BEQ ACTSHW ;BR IF A SHOW WAS TYPED
4590 025472 012737 000001 006430 MOV #1,CMPTOT ;CLEAR COMPARE MESSAGE COUNT, CHAR. COUNT
4591 025500 005037 006432 CLR CTOTCC ; AND RESET POINTER
4592
4593 025504 012737 006226 006426 MOV #PTR13,CMPPTR ;INIT COMPARE MESSAGE POINTER
4594 025512 013737 006426 006506 MOV CMPPTR,CPTR ;SET UP TO FILL IN DEFAULT MESSAGE
4595 025520 012701 005132 MOV #CMPBUF,R1
4596 025524 010137 006434 MOV R1,CCURAD
4597 025530 000431 BR ACTCLB
4598
4599 025532 012701 006132 ACTCST: MOV #PTRTAB,R1
4600 025536 013702 006444 MOV TXMTOT,R2
4601 025542 105037 003130 CLRB PSNNUF ;FLAG THAT HAVE VALID COMMAND AT THIS PT.
4602 025546 023727 003010 000002 CMP KEYWD1,#SHOW ;SEE IF A CLEAR OR SHOW WAS TYPED
4603 025554 001437 BEQ ACTSHW ;BR IF A SHOW WAS TYPED
4604 025556 012737 000001 006444 MOV #1,TXMTOT ;CLEAR TRANSMIT MESSAGE COUNT, CHAR. COUNT
4605 025564 005037 006446 CLR TTOTCC ; AND RESET POINTER
4606 025570 012737 006132 006424 MOV #PTRTAB,TXPTR

```

4607	025576	013737	006424	006506	MOV	TXPTR,CPTR		
4608	025604	012701	003132		MOV	#TXBUF,R1		
4609	025610	010137	006450		MOV	R1,TCURAD		
4610								
4611	025614	012702	001000		ACTCLB: MOV	#BUFLIM,R2		
4612	025620	010137	006510		MOV	R1,CURADD		;SET UP TO PUT DEFAULT MSG IN LIST AFTER 033'S
4613	025624	012737	000005	006500	MOV	#5,MSGTYP		
4614	025632	013737	002162	006502	MOV	MSG5C,CURCC		
4615	025640	105021			1\$: CLR	(R1)+		;FILL EXPT OR TRAN BUFFER WITH 0'S IF A CLEAR
4616	025642	005302			DEC	R2		;DO 'BUFLIM' NUMBER OF BYTE LOCATIONS
4617	025644	001375			BNE	1\$		
4618	025646	004737	021202		JSR	PC,BLDBUF		;'CLEAR' REALLY MEANS TO PUT DEFAULT MSG IN
4619	025652	000207			RTS	PC		;WHEN DONE, RETURN TO PARSER
4620								
4621								
4622	025654	012705	003054		ACTSHW: MOV	#SHTAB,R5		
4623	025660	122571	000000		5\$: CMP	(R5)+,@(R1)		;LOOK AT FIRST BYTE OF MSG TO DECIPHER TYPE
4624	025664	001404			BEQ	6\$		
4625	025666	020527	003063		CMP	R5,#SHTEND		;SEE IF LOOKED AT ALL OF DEFAULTS YET
4626	025672	001372			BNE	5\$		
4627	025674	005205			INC	R5		;MUST BE OPR. SPEC'D THEN
4628	025676	162705	003055		6\$: SUB	#SHTAB+1,R5		
4629	025702	006305			ASL	R5		
4630	025704	016137	000002	006516	MOV	2(R1),TEMP		
4631	025712				PRINTF	#SHMSG,SHTYTB(R5),TEMP		;PRINT MSG SIZE & TYPE
4632	025712	013746	006516				MOV	TEMP,-(SP)
4633	025716	016546	003034				MOV	SHTYTB(R5),-(SP)
4634	025722	012746	012631				MOV	#SHMSG,-(SP)
4635	025726	012746	000003				MOV	#3,-(SP)
4636	025732	010600					MOV	SP,R0
4637	025734	104417					TRAP	C\$PNTF
4638	025736	062706	000010				ADD	#10,SP
4639	025742	062701	000004		ADD	#4,R1		;BUMP R1 TO NEXT SET OF POINTERS
4640	025746	005302			DEC	R2		
4641	025750	001341			BNE	ACTSHW		
4642	025752	013737	006546	007662	MOV	MODTYP,DEV1		
4643	025760	013737	006550	007664	MOV	MLTYP,DEV2		
4644	025766	013737	006556	007666	MOV	RPASS,DEV3		
4645	025774	013737	006554	007670	MOV	PARAM,DEV4		
4646	026002	004737	021326		JSR	PC,SHWOP		;SHOW THE OPERATOR THE CURRENT MODE..... ALSO
4647	026006	000207			RTS	PC		
4648								
4649	026010	013737	003124	006472	ACTDMS: MOV	P\$NUM,STADD		;SETUP STARTING ADDRESS FOR DUMP
4650	026016	005037	006476		CLR	BYTBIT		;SET DEFAULT OF WORD DUMP
4651	026022	012737	000052	003010	MOV	#DMPS,KEYWD1		;FLAG THAT A DUMP WAS TYPED
4652	026030	000403			BR	ACTDME		
4653								
4654	026032	012737	177777	006476	ACTDMQ: MOV	#-1,BYTBIT		;SET DUMP FLAG TO 'DUMP-WORD'
4655	026040	013737	003124	006474	ACTDME: MOV	P\$NUM,ENADD		;SETUP END ADDRESS FOR DUMP (=START IF NO 'EEE')
4656	026046	105037	003130		ACTDMX: CLR	P\$NUF		;CLEAR NOT-ENOUGH FLAG, 'DUMP N-N/B' IS VALID
4657	026052	000207			RTS	PC		
4658								


```

4659
4660
4661 026054 012737 000010 003010 ACTSTE: MOV #SETEXP,KEYWD1
4662 026062 000403 BR ACTSTX
4663
4664 026064 012737 000011 003010 ACTSTT: MOV #SETTRN,KEYWD1
4665 026072 012737 000001 003014 ACTSTX: MOV #1,QUALVL ;SET UP DEFAULT COPY TO 1 (/COPY=0)
4666 026100 000207 RTS PC
4667
4668 026102 012737 000012 003012 ACTSIZE: MOV #SIZE,QUALFG
4669 026110 000207 RTS PC
4670
4671 026112 012737 000013 003012 ACTCOP: MOV #QCOPY,QUALFG
4672 026120 000207 RTS PC
4673
4674 026122 023727 003012 000012 ACTNUM: CMP QUALFG,#SIZE ;SEE IF A SIZE OR COPY TYPED
4675 026130 001023 BNE 1$ ;BR IF IT WAS A COPY
4676 026132 005737 003124 TST P$NUM ;CHECK TO BE SURE DIDN'T TRY SIZE=0
4677 026136 001014 BNE 3$ ; BR IF NO
4678 026140 PRINTF #CLISEO
4679 026140 012746 011763 MOV #CLISEO,-(SP)
4680 026144 012746 000001 MOV #1,-(SP)
4681 026150 010600 MOV SP,R0
4682 026152 104417 TRAP C$PNTF
4683 026154 062706 000004 ADD #4,SP
4684 026160 112737 177777 003131 MOVB #-1,P$GDBD ;SEE ERROR-IN-CMD FLAG
4685 026166 000411 BR 2$
4686 026170 013737 003124 006502 3$: MOV P$NUM,CURCC ;IF A SIZE LOAD CURCC WITH BYTE COUNT
4687 026176 000405 BR 2$
4688 026200 013737 003124 003014 1$: MOV P$NUM,QUALVL ;IF A COPY, LOAD COPY COUNT
4689 026206 005237 003014 INC QUALVL ;INCREMENT SO FIRST DEC MAKES IT REAL #
4690 026212 000503 2$: BR ACTMEX
4691
4692 026214 012737 000007 006500 ACTOPM: MOV #7,MSGTYP
4693 026222 010437 006516 MOV R4,TEMP ;KEEP TRACK OF START OF QUOTED TEXT
4694 026226 005237 006516 INC TEMP ; SO CAN CALC OPCNT AT END OF QUOTES
4695 026232 000207 RTS PC
4696
4697 026234 010402 ACTEQO: MOV R4,R2
4698 026236 163702 006516 SUB TEMP,R2
4699 026242 010237 006502 MOV R2,CURCC ;CALC BYTE COUNT FOR QUOTED TEXT
4700 026246 010237 002166 MOV R2,OPCNT
4701 026252 013701 006516 MOV TEMP,R1
4702 026256 012705 002520 MOV #OPBUF,R5
4703 026262 112125 1$: MOVB (R1)+,(R5)+ ;COPY QUOTED TEXT TO OPBUF
4704 026264 005302 DEC R2
4705 026266 001375 BNE 1$
4706 026270 000454 BR ACTMEX
4707
4708 026272 ACTBCR: PRINTF #CLIBCR ;BAD CHAR. IN OPR. QUOTED STRING
4709 026272 012746 011716 MOV #CLIBCR,-(SP)
4710 026276 012746 000001 MOV #1,-(SP)
4711 026302 010600 MOV SP,R0
4712 026304 104417 TRAP C$PNTF
4713 026306 062706 000004 ADD #4,SP
4714 026312 000207 RTS PC

```


4736	026430	012737	000003	006546	ACTATV: MOV	#ACT,MODTYP	
4737	026436	000432			BR	ACTM2X	
4738							
4739	026440	012737	000002	006546	ACTPAS: MOV	#PAS,MODTYP	
4740	026446	105037	003130		CLRB	PSNNUF	:CLEAR NOT-ENOUGH FLAG
4741	026452	005037	006550		CLR	MLTYP	:CLEAR MAINT LOOP TYPE
4742	026456	000207			RTS	PC	
4743							
4744	026460	005037	006546		ACTREC: CLR	MODTYP	
4745	026464	000417			BR	ACTM2X	
4746							
4747	026466	012737	000006	006546	ACTLIS: MOV	#LIS,MODTYP	
4748	026474	000413			BR	ACTM2X	
4749							
4750	026476	012737	000004	006546	ACTDLL: MOV	#DOW,MODTYP	
4751	026504	000407			BR	ACTM2X	
4752							
4753	026506	012737	000001	006546	ACTTRA: MOV	#TRA,MODTYP	
4754	026514	000403			BR	ACTM2X	
4755							
4756	026516	012737	000005	006546	ACTTAL: MOV	#TAL,MODTYP	
4757							
4758	026524	042737	000004	006554	ACTM2X: BIC	#ECHOB,PARAM	:DISABLE /ECHO (ALL BUT PASSIVE MODE)
4759	026532	105037	003130		CLRB	PSNNUF	:CLEAR NOT-ENOUGH FLAG
4760	026536	005037	006550		CLR	MLTYP	:CLEAR MAINT LOOP TYPE
4761	026542	000207			RTS	PC	
4762							

4763	026544	012737	000036	003012	ACTNO:	MOV	#NO,QUALFG	
4764	026552	000207				RTS	PC	
4765								
4766	026554	022737	000036	003012	ACTECH:	CMP	#NO,QUALFG	
4767	026562	001422				BEQ	1\$	
4768	026564	052737	000004	006554		BIS	#ECHOB,PARAM	
4769	026572	022737	000002	006546		CMP	#PAS,MODTYP	
4770	026600	001416				BEQ	2\$	
4771	026602					PRINTF	#CLINPS	
4772	026602	012746	011653					MOV #CLINPS,-(SP)
4773	026606	012746	000001					MOV #1,-(SP)
4774	026612	010600						MOV SP,R0
4775	026614	104417						TRAP C\$PNTF
4776	026616	062706	000004					ADD #4,SP
4777	026622	112737	177777	003131		MOVB	#-1,P\$GDBD	
4778	026630	042737	000004	006554	1\$:	BIC	#ECHOB,PARAM	
4779	026636	005037	003012		2\$:	CLR	QUALFG	
4780	026642	000501				BR	ACTLXX	;CLEAR 'NO' OUT OF QUALIFIER FLAG
4781								
4782	026644	012701	000002		ACTCHK:	MOV	#DATCKB,R1	
4783	026650	000413				BR	ACTQFG	;SET DATA CHECK BIT
4784								
4785	026652	012701	000001		ACTSTS:	MOV	#STATB,R1	
4786	026656	000410				BR	ACTQFG	;SET THE STATUS BIT
4787								
4788	026660	012701	000020		ACTCRC:	MOV	#CRCB,R1	
4789	026664	000405				BR	ACTQFG	;SET THE CRC BIT
4790								
4791	026666	012701	000010		ACTMOS:	MOV	#MOCHK,R1	
4792	026672	000402				BR	ACTQFG	;SET THE MODEM BIT
4793								
4794	026674	012701	000040		ACTPRO:	MOV	#PROTOB,R1	
4795								;SET THE PROTOCOL BIT
4796	026700	050137	006554		ACTQFG:	BIS	R1,PARAM	
4797	026704	022737	000036	003012		CMP	#NO,QUALFG	
4798	026712	001002				BNE	1\$	
4799	026714	040137	006554			BIC	R1,PARAM	
4800	026720	005037	003012		1\$:	CLR	QUALFG	
4801	026724	000450				BR	ACTLXX	;CLEAR 'NO' OUT OF QUALIFIER FLAG
4802								
4803	026726	013737	003124	006556	ACTRPS:	MOV	P\$NUM,RPASS	
4804	026734	000444				BR	ACTLXX	;GET NUMBER OF 'RUN PASSES'
4805								
4806	026736	012737	000005	006550	ACTMOP:	MOV	#5,MLTYP	
4807	026744	000417				BR	ACTLPX	
4808	026746	012737	000001	006550	ACTTLP:	MOV	#1,MLTYP	
4809	026754	000413				BR	ACTLPX	
4810	026756	012737	000002	006550	ACTCLP:	MOV	#2,MLTYP	
4811	026764	000407				BR	ACTLPX	
4812	026766	012737	000003	006550	ACTLLP:	MOV	#3,MLTYP	
4813	026774	000403				BR	ACTLPX	
4814	026776	012737	000004	006550	ACTRLP:	MOV	#4,MLTYP	
4815								
4816	027004	022737	000003	006546	ACTLPX:	CMP	#ACT,MODTYP	
4817	027012	001415				BEQ	ACTLXX	
4818	027014	112737	177777	003131		MOVB	#-1,P\$GDBD	;BE SURE IN ACTIVE IF TRYING TO SET LOOP ; BR IF IN ACTIVE

4819 027022 005037 006550
4820 027026
4821 027026 012746 011611
4822 027032 012746 000001
4823 027036 010600
4824 027040 104417
4825 027042 062706 000004
4826 027046 105037 003130
4827 027052 000207
4828

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: CLRB P\$NNUF
RTS PC

;CLEAR NOT-ENOUGH FLAG

```

4829
4830
4831 027054 012737 006132 006424 GTR9: ; RX ALLOCATE CODE
4832 027062 012737 006226 006426      MOV #PTRTAB, TXPTR ; INIT TRANSMIT MESSAGE POINTER
4833 027070 012737 006322 006422      MOV #PTR13, CMPPTR ; INIT COMPARE MESSAGE POINTER
4834      MOV #PTR23, RXPTR ; INIT RECEIVE MESSAGE POINTER
4835 027076 013737 006430 006460      MOV CMPTOT, RXMTOT ; MAKE COMPARE AND RX MESSAGE COUNTS EQUAL
4836
4837
4838 027104 005037 006560      GTREX: CLR FLAG ; CLEAR FLAG
4839 027110 005037 006464      CLR OPVAR ; CLEAR OPTIONAL VARIABLE COUNTER
4840 027114 005037 006466      CLR PSCNT ; CLEAR PASS COUNT
4841 027120 005037 006470      CLR ERRCNT ; CLEAR ERROR COUNT
4842 027124 005037 011414      CLR MGLCNT ; CLEAR GLITCH COUNT
4843 027130 005037 011416      CLR MHRCNT ; CLEAR HARD ERR. COUNT
4844 027134 012737 000626 011404      MOV #626, SYNCW ; SET UP SYNCW FOR 226 SYNC + TSOM
4845 027142 052737 000200 011366      BIS #BIT7, DPVP1 ; SET UP PARAM WORD FOR 226 RX SYNC
4846 027150 005737 011420      TST RNODE
4847 027154 001406      BEQ 1$ ; IF NON ITEMP GO TO 1
4848 027156 042737 000200 011366      BIC #BIT7, DPVP1 ; SET UP FOR 26 SYNC WORD ON RX.
4849 027164 012737 000426 011404      MOV #426, SYNCW ; ELSE SET UP SYNC FOR 26 AND TSOM
4850 027172 004737 017264      1$: JSR PC, LOGDVI ; LOG ABOUT TO INIT DEVICE
4851 027176 004737 031232      JSR PC, DVINIT ; INIT DEVICE
4852
4853 027202 012737 001000 006502 GTRX2: MOV #BUFLIM, CURCC ; SET CHAR COUNT TO 'BUFLIM' NO. OF BYTES
4854 027210 012737 004132 006510      MOV #RXBUF, CURADD ; SET UP RX BUFFER AS CURRENT ADD.
4855 027216 013737 006422 006506      MOV RXPTR, CPTR
4856 027224 012737 000010 006500      MOV #10, MSGTYP ; SET UP FOR 33 TO FILL RX BUFFERS
4857 027232 004737 021202      JSR PC, BLDBUF ; CLEAR RX BUFFER
4858 027236 013702 006546      MOV MODTYP, R2
4859 027242 006302      ASL R2
4860 027244 000172 006562      JMP @MODE(R2) ; MODE DISPATCH
4861

```

4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884

027250
027250 013737 006422 006504
027256 013737 006460 006456
027264 052737 000104 006560
027272 005037 006506
027276 000137 027434

```
.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:
RXON2:  MOV    RXPTR,CPTRR
        MOV    RXMTOT,DVRCT ;SET UP MESSAGE COUNT
        BIS    #QRX+#ERX,FLAG ;SET UP RX QUE
        CLR    CPTR ;CLEAR THE TX POINTER
        JMP    ALLTR ;GO RX.
```


4885
4886
4887
4888
4889
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900
4901
4902
4903
4904
4905
4906

.SBTTL TRANSMIT MODE SECTION

..++
: FUNCTIONAL DESCRIPTION:
: TRANSMIT-ONLY (OR ONE-WAY-OUT) ROUTINE
: IN THIS MODE OF TESTING A LIST OF MESSAGES IS TRANSMITTED WITHOUT
: EXPECTING ANY DATA TO BE RECEIVED. A REPETITION COUNT CAN BE
: SPECIFIED TO REPETITIVELY TRANSMIT THE LIST.

: SUBORDINATE ROUTINES USED:
: "ALLTR"

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

..--
TXONLY: BIC #DATCKB,PARAM ;SET NOCHECK
TXON: MOV TXPTR,CPTR
MOV TXMTOT,DVTCT ;COPY COUNTER FOR THIS PASS
BIS #QTX+#ETX,FLAG ;SET THE QUE TX FLAG
CLR CPTRR ;CLEAR RX POINTER
JMP ALLTR ;GO TX.

027302 042737 000002 006554
027310 013737 006424 006506
027316 013737 006444 006442
027324 052737 000210 006560
027332 005037 006504
027336 000137 027434

4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919
4920
4921
4922
4923
4924
4925
4926
4927
4928
4929
4930
4931

.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

:--

027342
027342 013737 006444 006442
027350 013737 006424 006506
027356 013737 006422 006504
027364 052737 000104 006560
027372 000137 027434

PLCK:
PLCK2: MOV TXMTOT,DVTCT ;SET UP THE TRANSMIT COUNT
MOV TXPTR,CPTR ;SET UP CPTR TO TRANSMIT POINTER
PLCK3: MOV RXPTR,CPTRR ;SET UP CPTRR TO REC POINTER
BIS #QRX+#ERX,FLAG ;SET UP Q AND EXPECT RX
JMP ALLTR ;AND GO RX FIRST MSG.

4932
4933
4934
4935
4936
4937
4938
4939
4940
4941
4942
4943
4944
4945
4946
4947
4948
4949
4950
4951
4952
4953
4954
4955
4956
4957
4958
4959

.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:	MOV	TXMTOT,DVTCT	
	MOV	TXPTR,CPTR	;SET UP TX COUNTS
	MOV	RXMTOT,DVRCCT	;SET UP COUNTS
	MOV	RXPTR,CPTRR	
	BIS	#QRX+#QTX+#ETX+#ERX,FLAG	

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
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

```
.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 RECIVE 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 RECIEVE 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
: --

: ALLTR:
: ALCK5: BIT      #QRX,FLAG
:        BEQ     ALCK1          ;IF NOT RX GO TO TX'S
:        MOV     CPTRR,R2
:        MOV     (R2),TEMP2
:        MOV     (R2)+,DVRXA
:        MOV     (R2),TEMP3
:        MOV     (R2),DVRCC
:        MOV     R2,CPTRR
:        JSR     PC,DVRXQ        ;GO QUE DEVICE
:        JSR     PC,LOGRXQ      ;LOG REC QUED
: ALCK1: BIT      #QTX,FLAG
```

```
027434
027434 032737 000004 006560
027442 001420
027444 013702 006504
027450 011237 006522
027454 012237 006452
027460 011237 006524
027464 011237 006454
027470 010237 006504
027474 004737 032104
027500 004737 017220
027504 032737 000010 006560
```

```

5016 027512 001416          BEQ     ALCK2          ;IF NO TX'S GO TO 2
5017 027514 013702 006506   MOV     CPTR,R2
5018 027520 011237 006522   MOV     (R2),TEMP2
5019 027524 012237 006436   MOV     (R2)+,DVTXA
5020 027530 011237 006524   MOV     (R2),TEMP3
5021 027534 012237 006440   MOV     (R2)+,DVTCC
5022 027540 010237 006506   MOV     R2,CPTR
5023 027544 004737 017164   JSR     PC,LOGTXQ
5024
5025 027550 004737 032206   ALCK2: JSR     PC,DVTXRX ;GO TO TX AND RX SUB ROUT.
5026
5027 027554 032737 000004 006560   BIT     #QRX,FLAG      ;CHECK FOR REC. MSG.
5028 027562 001514          BEQ     ALCK3
5029 027564 013737 006452 006522   MOV     DVRXA,TEMP2
5030 027572 013737 006454 006524   MOV     DVRCC,TEMP3
5031 027600 004737 017236          JSR     PC,LOGRXC      ;LOG REC COMPLETE
5032 027604 032737 000004 006554   UPTABL: BIT     #ECHOB,PARAM ;IS THIS ECHO MODE(PASSIVE)
5033 027612 001406          BEQ     UPTA4          ;IF NOT GO TO 4
5034 027614 013702 006506          MOV     CPTR,R2       ;ELSE SET R2 TO PRESENT TX TABL
5035 027620 013722 006522          MOV     TEMP2,(R2)+   ;STORE OFF RX ADD
5036 027624 013712 006524          MOV     TEMP3,(R2)   ;AND CC
5037 027630 032737 000002 006554   UPTA4: BIT     #DATCKB,PARAM ;IS DATA CHECKING ASKED FOR
5038 027636 001015          BNE     UPTA1         ;IF SO GO TO 1
5039 027640 012737 000001 006456   MOV     #01,DVRCT     ;ELSE SET DVRCT TO A 1
5040 027646 013737 006422 006504   MOV     RXPTR,CPTRR   ;RESET POINTER
5041 027654 022737 000003 006546   CMP     #ACT,MODTYP   ;IS THIS ACTIVE
5042 027662 001002          BNE     UPTA3
5043 027664 005237 006456          INC     DVRCT         ;IF YES BUMP COUNT
5044 027670 000424          UPTA3: BR      UPTEX
5045 027672 013702 006504          UPTA1: MOV     CPTRR,R2
5046 027676 011237 006516          MOV     (R2),TEMP     ;LOAD TEMP WITH PREV. COUNT
5047 027702 163737 006524 006516   SUB     TEMP3,TEMP    ;LOAD TEMP WITH PREV.COUNT-CURRENT
5048 027710 013722 006524          MOV     TEMP3,(R2)+
5049 027714 063737 006524 006522   ADD     TEMP3,TEMP2
5050 027722 013722 006522          MOV     TEMP2,(R2)+  ;STORE OF NEW ADD
5051 027726 013712 006516          MOV     TEMP,(R2)    ;AND NEW CC
5052 027732 162702 000002          SUB     #2,R2        ;PUT POINTER BACK TO ADDR.
5053
5054 027736 010237 006504          MOV     R2,CPTRR     ;AND RESTORE IT.
5055
5056 027742          UPTEX:
5057 027742 022737 000002 006546   CMP     #PAS,MODTYP
5058 027750 001007          BNE     ALCK2A        ;IF NOT PASSIVE LOOP THEN GO TO 2A
5059 027752 042737 000104 006560   BIC     #QRX+#ERX,FLAG ;CLEAR BOTH EXPECTED AND COMPLETED FLAGS
5060 027760 052737 000210 006560   BIS     #QTX+#ETX,FLAG ;SET THE TX FLAGS
5061 027766 000646          BR      ALCK1
5062
5063 027770 005337 006456          ALCK2A: DEC     DVRCT   ;DEC REC COUNT
5064 027774 005737 006456          TST     DVRCT        ;IS IT ALL DONE
5065 030000 001005          BNE     ALCK3        ;NO. GO CHECK TX
5066 030002 042737 000004 006560   BIC     #QRX,FLAG    ;CLEAR THE RX FLAG
5067 030010 005037 006504          CLR     CPTRR        ;YES. CLEAR POINTER
5068 030014 032737 000010 006560   ALCK3: BIT     #QTX,FLAG ;IS IT TX
5069 030022 001447          BEQ     ALCK4        ;IF NOT TX THEN GO BACK
5070 030024 013737 006436 006522   MOV     DVTXA,TEMP2
5071 030032 013737 006440 006524   MOV     DVTCC,TEMP3  ;LOG TX COMPLETED

```

5072	030040	004737	017202		JSR	PC,LOGTXC	
5073	030044	005337	006442		DEC	DVICT	;DEC TX COUNT
5074	030050	022737	000002	006546	CMP	#PAS,MODTYP	
5075	030056	001013			BNE	ALCK3A	;IF NOT PASSIVE MODE GO TO 3A
5076	030060	042737	000210	006560	BIC	#QTX+ETX,FLAG	;CLEAR THE TX FLAGS
5077	030066	052737	000104	006560	BIS	#QRX+ERX,FLAG	;AND SET THE RX FLAGS
5078	030074	005737	006442		TST	DVICT	
5079	030100	001005			BNE	ALCK3C	;IF MORE RX'S DO IT
5080	030102	000137	030162		JMP	CMPSR	; ELSE COMPARE
5081	030106	005737	006442		ALCK3A: TST	DVICT	;IS IT ALL DONE
5082	030112	001402			BEQ	ALCK3B	;IF NOT GO BACK TO 5
5083	030114	000137	027434		ALCK3C: JMP	ALCK5	
5084	030120	005037	006506		ALCK3B: CLR	CPTR	;IF SO CLEAR POINTER
5085	030124	042737	000010	006560	BIC	#QTX,FLAG	;CLEAR TX FLAG
5086	030132	032737	000002	006554	BIT	#DATCKB,PARAM	;IS IT DAT CK
5087	030140	001403			BEQ	ALCK4A	;IF NOT THEN END WORKING RX.
5088	030142	005737	006504		ALCK4: TST	CPTRR	
5089							
5090	030146	001362			BNE	ALCK3C	;IF SOME RX'S LEFT GO BACK
5091	030150	005737	006506		ALCK4A: TST	CPTR	
5092	030154	001402			BEQ	ALCK4B	;BRANCH IF ANY TX'S LEFT
5093	030156	000137	027550		JMP	ALCK2	
5094	030162				ALCK4B:		
5095							
5096							
5097							

5098
5099
5100
5101
5102
5103
5104
5105
5106
5107
5108
5109
5110
5111
5112
5113
5114
5115
5116
5117
5118
5119
5120
5121
5122
5123
5124
5125
5126
5127
5128
5129
5130
5131
5132
5133
5134
5135
5136
5137
5138
5139
5140
5141
5142
5143
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153

.SBTTL DATA COMPARISON CODE

++
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

--

5130	030162	032737	000002	006554	CMPSR:	BIT	#DATCKB,PARAM	;IS DATA CHECKING TO BE DONE
5131	030170	001522				BEQ	CMPSX	;IF NOT THEN EXIT
5132	030172	013737	006422	006506		MOV	RXPTR,CPTR	;PUT START OF RX POINTERS TO CPTR
5133	030200	013737	006426	006504		MOV	CMPTTR,CPTRR	; AND START OF COMPARE POINTS TO CPTRR
5134	030206	013737	006460	006456		MOV	RXMTOT,DVRCT	
5135								
5136	030214				CMPS3:			
5137	030214	013702	006506			MOV	CPTR,R2	;MOVE CURRET RX PT.TO R2
5138	030220	011237	006522			MOV	(R2),TEMP2	;MOVE RX ADD TO EVENT LOG
5139	030224	012201				MOV	(R2)+,R1	;SET R1 TO START ADD OF RX
5140	030226	012237	006524			MOV	(R2)+,TEMP3	;SET CHAR COUNT TO EVENT LOG
5141	030232	010237	006506			MOV	R2,CPTR	;RESTORE RX POINT
5142								
5143	030236	013702	006504			MOV	CPTRR,R2	;PUT R2 AT COMPARE TABLE
5144	030242	012203				MOV	(R2)+,R3	;SET R3 TO COMPARE ADD
5145	030244	012204				MOV	(R2)+,R4	;SET R4 TO COMP CC
5146	030246	010237	006504			MOV	R2,CPTRR	;RESTORE POINTER
5147	030252	010437	006526			MOV	R4,TEMP4	
5148	030256	004737	017332			JSR	PC,LOGCMP	;LOG COMPARE START.
5149								
5150	030262	020437	006524			CMP	R4,TEMP3	;IS COMPARE COUNT = TO RX COUNT
5151	030266	001410				BEQ	CMPS7	;IF SO GO TO 7
5152	030270	005237	006470			INC	ERRCNT	
5153	030274					ERRSOFT	1,EDDL,ERR10	;PRINT ERROR

5194
5195
5196
5197
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207
5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220
5221
5222
5223
5224
5225
5226
5227
5228
5229
5230
5231
5232
5233
5234

.SBTTL MODEM CHANGE REPORTS

```

:++
:FUNCTIONAL DESCRIPTION:
:THIS SECTION REPORTS THE NUMBER OF MOMDEM STATUS CHANGES
:THAT OCCUR ON EACH PASS. THE ERROR IS ONLY REPORTED IF
:THERE WERE ANY CHANGES IN OTHER WORDS A COUNT OF ZERO IS
:NOT REPORTED. THE CHANGES ARE REPORTED IN TWO CLASSES ..
:HARD ERRORS AND GLITCHES. HARD ERRORS ARE WHEN THE DEVICE
:IS ABLE TO LATCH UP THE BAD MODEM STATUS. GLITCHES OCCUR
:WHEN THE MODEM STATUS CHANGES TO CAUSE A DATA SET CHANGE
:INTERRUPT BUT THE CHANGE DOES NOT OCCUR LONG ENOUGH FOR
:THE DEVICE TO LATCH THE DATA

```

```

:INPUTS:
: 'MGLCNT' - CONTAINS NUMBER OF GLITCH ERRORS
: 'MHRCNT' - CONTAINS NUMBER OF HARD ERRORS

```

```

:OUTPUTS:
: 'MGLCNT' -ZEROED BY THIS SECTION
: 'MHRCNT' -ZEROED BY THIS SECTION

```

:--

```

CMPSEX: TST MGLCNT ;CHECK FOR ANY GLITCH ERRORS
        BNE MCREP ;IF NON ZERO REPORT THEM
        TST MHRCNT ;CHECK FOR ANY HARD ERRORS
        BEQ ENDPS ;IF NONE GO TO END OF PASS

```

:REPORT ANY MODEM ERRORS HERE

```

MCREP: INC ERRCNT ;BUMP ERROR COUNT
        ERRSOFT 4, MSCMS, ERR4

```

```

TRAP CSERSOFT
.WORD 4
.WORD MSCMS
.WORD ERR4

```

```

CLR MGLCNT ;CLEAR GLITCH COUNT
CLR MHRCNT ;CLEAR THE HARD COUNT

```

```

030436 005737 011414
030442 001003
030444 005737 011416
030450 001412
030452 005237 006470
030456 104457
030460 000004
030462 014057
030464 016670
030466 005037 011414
030472 005037 011416

```


5235
5236
5237
5238
5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250
5251
5252
5253
5254
5255
5256
5257
5258
5259
5260
5261
5262
5263
5264
5265

.SBTTL

INTERNAL END OF PASS CODE

:**

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 PROMT
IN NOT = TO 0 THEN GO BACK TO MODE DISPATCHER

SUBORDINATE ROUTINES USED:

'LOGEOP' - LOG END OF PASS TO EVENT LOG

030476	005237	006466	ENDPS:	INC	PSCNT	:BUMP PASS COUNT
030502	013737	006464		MOV	OPVAR,TEMP4	
030510	013737	006466		MOV	PSCNT,TEMP2	
030516	013737	006470		MOV	ERRCNT,TEMP3	
030524	004737	017404		JSR	PC,LOGEOP	:LOG END OF PASS
030530	022737	177777		CMP	#-1,RPASS	:SEE IF RPASS=-1
030536	001403			BEQ	1\$:IF IT IS DON'T DECRMNT, LOOP FOREVER
030540	005337	006556		DEC	RPASS	:DEC PASS COUNT
030544	001402			BEQ	2\$:IF DONE EXIT TEST
030546	000137	027202	1\$:	JMP	GTRX2	:ELSE GO BACK AND DISPATCH
030552	000137	024122	2\$:	JMP	GTRA5	:WHEN RPASS=0 GO BACK TO 'DCLT>'

5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277 030556
5278 030556
5279 030556 012746 013230
5280 030562 012746 000001
5281 030566 010600
5282 030570 104417
5283 030572 062706 000004
5284 030576 000137 024122
5285

.SBTTL DOWN-LINE-LOAD SECTION
:++
: FUNCTIONAL DESCRIPTION:
: DOWN LINE LOAD IS NOT SUPPORTED BY THIS DEVICE..
: IF THIS MODE IS CALLED BY THE COMMAND LINE INTERPRETER
: THEN A MESSAGE WILL BE PRINTEDTHAT SAYS DOWN LINE
: LOAD IS NOT!! SUPPORTED BY THIS DEVICE.
:--

D'L: PRINTF #DLLCM

JMP GTRA5

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

5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305 030602
5306 030602 042737 000002 006554
5307 030610 012702 002520
5308 030614 012722 177777
5309 030620 022702 002642
5310 030624 001373
5311 030626
5312 030626 104443
5313 030630 000406
5314 030632 002520
5315 030634 000142
5316 030636 013165
5317 030640 000000
5318 030642 000001
5319 030644 000110
5320 030646
5321 030646 005002
5322 030650 122762 000377 002520 2\$:
5323 030656 001402
5324 030660 005202
5325 030662 000772
5326 030664 010237 002166 3\$:
5327
5328 030670 012737 002520 006436
5329 030676 012737 002520 006522
5330 030704 013737 002166 006524
5331 030712 013737 002166 006440
5332 030720 004737 017164
5333 030724 052737 000210 006560
5334 030732 005037 006504
5335
5336 030736 004737 032206
5337
5338 030742 013737 006436 006522
5339 030750 013737 006440 006524
5340 030756 004737 017202
5341 030762 022737 054105 002520

.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:
BIC #DATCKB,PARAM ;SET NOCHECK
MOV #OPBUF,R2
1\$: MOV #-1,(R2)+ ;CLEAR OUT OPBUFFER FIRST
CMP #OPEND,R2
BNE 1\$
GMANID OPRMM,OPBUF,A,0,1,72.,NO ;GET TALK MESSAGE
TRAP C\$GMAN
BR 10001\$
.WORD OPBUF
.WORD T\$CODE
.WORD OPRMM
.WORD 0
.WORD T\$LOLIM
.WORD T\$HILIM
10001\$:
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
CMP #'EX,OPBUF ;CHECK FOR EXIT

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^{M 10} 10:29 PAGE 130
TALK MODE SECTION

SEQ 0129

5342	030770	001304		
5343	030772	022737	052111	002522
5344	031000	001300		
5345	031002	042737	000210	006560
5346	031010	012737	000006	006546
5347	031016	000137	027202	

BNE	TALCK	
CMP	#'IT,OPBUF+2	
BNE	TALCK	
BIC	#QTX+#ETX,FLAG	;CLEAR THE TX BITS
MOV	#LIS,MODTYP	;CHANGE TO LISTEN MODE
JMP	GIRX2	;AND GO BACK TO DISPATCH

5348
5349
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359
5360
5361
5362
5363
5364
5365
5366
5367
5368 031022 042737 000002 006554
5369 031030
5370 031030 012746 013154
5371 031034 012746 000001
5372 031040 010600
5373 031042 104417
5374 031044 062706 000004
5375 031050 012737 002520 006452
5376 031056 012737 002520 006522
5377 031064 012737 000122 006454
5378 031072 012737 000122 006524
5379 031100 052737 000104 006560
5380 031106 005037 006506
5381
5382 031112 004737 032104
5383 031116 004737 017220
5384
5385 031122 004737 032206
5386
5387 031126 013737 006452 006522
5388 031134 013737 006454 006524
5389 031142 004737 017236
5390 031146 063737 006452 006454
5391 031154 105077 155274
5392 031160
5393 031160 012746 002514
5394 031164 012746 000001
5395 031170 010600
5396 031172 104417
5397 031174 062706 000004
5398 031200 022737 054105 002520
5399 031206 001320
5400 031210 022737 052111 002522
5401 031216 001314
5402 031220 012737 000005 006546
5403 031226 000137 027202

.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: BIC #DATCKB,PARAM ;CLEAR CHECK BIT
PRINTF #LISP ;PRINT PROMPT FOR OPR.
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
MOV #TAL,MODTYP ;CHANGE MODE TO TALK
JMP GTRX2 ;RETURN TO DISPATCHER

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80^B 11 10:29 PAGE 132
LISTEN MODE SECTION

SEQ 0131

5404
5405

5406
5407
5408
5409
5410
5411
5412
5413
5414
5415
5416
5417
5418
5419
5420
5421
5422
5423
5424
5425
5426
5427
5428
5429
5430
5431
5432
5433
5434
5435
5436
5437
5438
5439
5440
5441
5442
5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461

.SBTTL DEVICE FUNCTION SUBROUTINES

.SBTTL DEVICE INIT SUBROUTINE

++
FUNCTIONAL DESCRIPTION:
DVINIT- DEVICE INIT ROUTINE
THIS ROUTINE IS DEVICE DEPENDENT CODE THAT INITIS
THE DEVICE BEING TESTED.
IT SETS THE DEVICE UP TO THE MODE IT IS TO RUN IN AND
INITIATES THE START,STACK,ACK SEQUENCE IF THE 'RNODE'(REMOTE
NODE)INPUT INDICATES THE REMOTE NODE IS NON-ITEP.
INPUTS: 'FHDPLX' INDICATES IF MODE IS FULL OR HALF DUPLEX. (1=FULL)
ADDRESS POINTERS (SELO,...) ALREADY POINT TO DEVICE'S REG.S
'MLTYP' INDICATES THE LOOP TYPE (1=TTL,2=CAB,3=RM,4=LM)
'RNODE' INDICATES THE TYPE OF REMOTE NODE (ITEP=1,NON-ITEP=0)
SUBORDINATE ROUTINES USED:
'CTSSR' - CLEAR TO SEND SUB ROUTINE
'DVIN31' - SEND CONTROL AND REC OR TIME OUT
'CLRRTS' - CLEAR REQUEST TO SEND ROUTINE
'LGDVE' - LOG DEVICE ERROR TO EVENT LOG
CALLING SEQUENCE:
JSR PC,DVINIT
--

DVINIT:

```
;MASTER CLEAR DEVICE
MOV #RESET,@TXCSR ;DO A MASTER CLEAR
TSTB @RXCSR ;SEE IF IT WORKED
BEQ DVINI ;BRANCH IF OK
BREAK
TRAP CSBRK

;REPORT ERROR IF RESET
;DOES NOT WORK

MOV #DVEMO,TEMP2
MOV @RXCSR,TEMP3
MOV @TXCSR,TEMP4 ;LOAD UP ERRM. AND REG OUTPUTS
JSR PC,LGDVE ;LOG TIME OUT WAITING FOR RUN
INC ERRCNT
ERRSOFT 5,DVEMO,ERR13
TRAP CSERSOFT
.WORD 5
.WORD DVEMO
```

031232
031232 012777 000001 160112
031240 105777 160100
031244 001423
031246
031246 104422
031250 012737 016035 006522
031256 017737 160062 006524
031264 017737 160062 006526
031272 004737 017246
031276 005237 006470
031302
031302 104457
031304 000005
031306 016035

```

5462 031310 016722
5463 031312 000747          BR DVINIT          ;GO BACK AND TRY MSTR CLR AGAIN IF ERROR
5464
5465          ;SET TTL LOOP IF REQU'D
5466
5467 031314 042737 000003 006560 DVIN1: BIC      #3,FLAG      ;CLEAR INPUT AND OUTPUT INT FLAGS
5468 031322 042777 000010 160022      BIC      #TTL,@TXCSR    ;CLEAR INTERNAL LOOP
5469 031330 022737 000001 006550      CMP      #TTL,MLTYP     ;IS TTL SELECTED
5470 031336 001004          BNE      DVIN3         ; IF NOT GO TO 3
5471 031340 052777 000010 160004      BIS      #TTL,@TXCSR    ;ELSE SET INTERNAL LOOP
5472 031346 000455          BR       DVIN37
5473
5474 031350 022737 000002 006550 DVIN3: CMP      #CABLE,MLTYP
5475 031356 001451          BEQ      DVIN37        ; IF CABLE LOOP SKIP CHECK
5476          ;FOR MODEM READY
5477
5478 031360 022737 000004 006546          CMP      #DOW,MODTYP    ;CHECK IF DLL
5479 031366 001002          BNE      DVIN3A        ;BRANCH IF NOT DLI
5480 031370 000137 032020          JMP      DVINEX         ;ELSE EXIT
5481
5482 031374 012777 000002 157742 DVIN3A: MOV     #DTR,@RXCSR ;SET UP DTR.
5483
5484 031402 012737 002000 006620          MOV     #2000,TIMER1
5485 031410 005737 006620 DVIN38: TST     TIMER1
5486 031414 001022          BNE      DVIN39        ;IF TIMER NOT OUT GO TO 39
5487
5488          ;SET ERROR FOR NO MODEM READY
5489
5490 031416 012737 016463 006522      MOV     #DVEM6,TEMP2
5491 031424 017737 157714 006524      MOV     @RXCSR,TEMP3
5492 031432 017737 157714 006526      MOV     @TXCSR,TEMP4
5493 031440 004737 017246          JSR     PC,LGDVE
5494 031444 005237 006470          INC     ERRCNT
5495 031450          ERRSOFT 11,DVEM6,ERR13
5496 031450 104457
5497 031452 000013
5498 031454 016463
5499 031456 016722
5500 031460 000745
5501 031462
5502 031462 104422
5503 031464 017737 157654 011372      MOV     @RXCSR,IRXCSR   ;GET COPY OR RXCSR
5504 031472 032737 001000 011372      BIT     #BIT9,IRXCSR    ;IS MODEM READY SET
5505 031500 001743          BEQ     DVIN38
5506 031502 013777 011366 157636 DVIN37: MOV     DPVP1,@PCASAR ;SET UP PCASAR
5507 031510 005737 011420          TST     RNODE           ;CHECK REMOTE NODE
5508 031514 001141          BNE     DVINEX          ;EXIT IF ITP
5509 031516 005737 006552          TST     FHDPLX         ;IS THIS FULL DUPLEX
5510 031522 001536          BEQ     DVINEX          ;BANCH IF NOT
5511
5512          ;SET UP TO SEND STRT
5513 031524 112737 000005 002645      MOV     #5,HDMMSG+1     ;SET UP ENQ
5514 031532 052737 000060 006560      BIS     #RXM!TXM,FLAG   ;SET FLAG WORD
5515 031540 012737 000074 006624      MOV     #60.,TIMERS     ;SET TIMER FOR 1 MINUTE
5516 031546 004737 033476          JSR     PC,CTSSR        ;SET CTS IF NESC.
5517 031552 012737 000006 002646 DVIN41: MOV     #6,HDMCC    ;SET UP STRT CODE

```

```

TRAP  CSERSOFT
.WORD 11
.WORD DVEM6
.WORD ERR13

```

```

TRAP  CSBRK

```



```

5518 031560 004737 033346      JSR    PC,DVIN31      ;GO TX STRT AND CHK FOR RX.
5519 031564 005737 006624      TST    TIMERS
5520 031570 001466              BEQ    DVIN81        ;IF TIMER EXPIERED EXIT
5521
5522 031572 022737 000006 002660 DVIN4:  CMP    #6,RHDMCC      ;IS THE RCVD=STRT
5523 031600 001441              BEQ    DVIN8         ;IF SO GO TO ASTRT
5524 031602 022737 000007 002660      CMP    #7,RHDMCC      ;IS IT A STACK
5525 031610 001360              BNE    DVIN41        ;IF NOT STACK ETIHER GO BACK
5526
5527 031612 004737 033476      DVIN9: JSR    PC,CTSSR      ;SET REQUEST TO SEND
5528 031616 042737 001010 006560      BIC    #QTX!PAD,rLAG  ;CLEAR TX COMPT FLAG.
5529 031624 012737 000001 002646      MOV    #1,HDMCC       ;SET UP ACK
5530 031632 012737 002645 011376      MOV    #HDMMSG+1,MSGPTR ;SET UP POINTER
5531 031640 013737 002654 011400      MOV    HDMC,MSGCC
5532 031646 012737 000010 011402      MOV    #8.,SYNCC      ;SET UP SYNC COUNT
5533 031654 052777 000120 157470      BIS    #TXENA!TINTEN,@TXCSR
5534 031662 032737 000010 006560 DIVN91: BIT    #QIX,FLAG
5535 031670 001053              BNE    DVINEX        ;EXIT IF ACK SENT
5536 031672
5537 031672 104422              BREAK
5538 031674 005737 006624              TST    TIMERS
5539 031700 001370              BNE    DIVN91        ;IF NOT TIMER EXPIRED RECHK TX.
5540 031702 000421              BR     DVIN81        ;IF TIMER OUT REPORT IT
5541
5542 031704 012737 000007 002646 DVIN8:  MOV    #7,HDMCC       ;SET POTINTER TO STACK
5543 031712 004737 033346              JSR    PC,DVIN31      ;AND GO SEND STACK
5544 031716 005737 006624              TST    TIMERS
5545 031722 001411              BEQ    DVIN81        ;REPORT ERROR IF TIME OUT
5546 031724 022737 000001 002660      CMP    #1,RHDMCC      ;IS IT ACK RCVD?
5547 031732 001432              BEQ    DVINEX        ;IF SO EXIT
5548 031734 022737 000007 002660      CMP    #7,RHDMCC      ;IS IT STACK RCVD
5549 031742 001723              BEQ    DVIN9         ;IF SO SEND ACK
5550 031744 000757              BR     DVIN8         ;IF NEITHER SEND ANOTHER ACK
5551
5552              ;DO ERROR AND REPEAT
5553
5554 031746 012737 016377 006522 DVIN81: MOV    #DVEM5,TEMP2
5555 031754 013737 002660 006524      MOV    RHDMCC,TEMP3
5556 031762 013737 002646 006526      MOV    HDMCC,TEMP4
5557 031770 004737 017246              JSR    PC,LGDVE
5558 031774 005237 006470              INC    ERRCNT
5559 032000              ERRSOFT 10.,DVEM5,ERR13
5560 032000 104457
5561 032002 000012
5562 032004 016377
5563 032006 016722
5564 032010 005237 006464              INC    OPVAR
5565 032014 000137 031232              JMP    DVINIT        ;COUNT HOW MANY TIMES WE DO THIS.
5566
5567 032020 004737 033650      DVINEX: JSR    PC,CLRRTS      ;CLEAR RTS IF NESC
5568 032024 042737 173777 006560      BIC    #173777,FLAG  ;CLEAR FLAG WORD
5569 032032 052737 002000 006560      BIS    #INOVR,FLAG   ;SET THE INITT OVER FLAG
5570 032040 000207              RTS    PC             ;RETURN TO CALLER
5571
5572

```

TRAP CSBRK

TRAP CSERSOFT
.WORD 10
.WORD DVEM5
.WORD ERR13

5573
5574
5575
5576
5577
5578
5579
5580
5581
5582
5583
5584
5585
5586
5587
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597
5598
5599
5600
5601

.SBTTL DEVICE GET MODEM STATUS SUBROUTINE

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

032042	017737	157276	007534	DVMODS: MOV	@RXCSR,MODS	
032050	042737	000040	007534	BIC	#BIT5,MODS	: CLEAR BIT 5
032056	032777	000040	157266	BIT	#BIT5,@TXCSR	: SEE IT TM OR SQ SET
032064	001403			BEQ	DVMEX	: IF NOT EXIT
032066	052737	000040	007534	BIS	#BIT5,MODS	: IF SET SET BIT 5 IN MODS
032074	042737	106720	007534	DVMEX: BIC	#106720,MODS	: CLEAR ALL UNUSED BITS
032102	000207			RTS	PC	: RETURN TO CALLER

5602
5603
5604
5605
5606
5607
5608
5609
5610
5611
5612
5613
5614
5615
5616
5617
5618
5619
5620
5621
5622
5623
5624
5625
5626
5627
5628
5629
5630
5631
5632
5633
5634
5635
5636
5637
5638
5639
5640
5641

```
.SBTTL                DEVICE QUEUE RECEIVE SPACE SUBROUTINE

:++
: FUNCTIONAL DESCRIPTION:
:   DVRXQ - THIS SUBROUTINE QUEUES THE RECIEVER 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
:
: CALLING SEQUENCE:
:   JSR    PC,DVRXQ
:--
```

```
032104
032104 032737 000004 006560
032112 001434
032114 042737 000444 006560
032122 005737 011420
032126 001415
032130 052737 000440 006560
032136 013737 006452 011406
032144 012737 000072 011410
032152 012737 000070 006454
032160 000406
032162 012737 002657 011406
032170 013737 002654 011410
032176 052777 000160 157140
032204 000207
```

```
DVRXQ:
BIT    #QRX,FLAG
BEQ    DVREX                ;IF NOT RX THEN EXIT
                        ;ELSE QUE RX
BIC    #QRX+#BCC+#RXM,FLAG ;CLEAR FLAG FOR RX
TST    RNODE                ;IF NON ITEMP GO TO 2
BEQ    DVRX2
BIS    #RXM+#BCC,FLAG        ;GET JUST THE DATA NO CRC.
MOV    DVRXA,RMSGPT
MOV    #72,RMSGCC            ;SET UP RX TO GET ITEMP MSG.
MOV    #70,DVRCC
BR     DVRX3

;ENABLE RX, RX INTERRUPTS,AND DATA SET INTERRUPTS
DVRX2: MOV    #RHMSG+1,RMSGPT ;SET UP POINTER
MOV    HDMC,RMSGCC           ;AND CC
DVRX3: BIS    #RINTEN!RXENA!#DSITEN,@RXCSR
DVREX: RTS    PC             ;RETURN TO CALLER
```

5642
5643
5644
5645
5646
5647
5648
5649
5650
5651
5652
5653
5654
5655
5656
5657
5658
5659
5660
5661
5662
5663
5664
5665
5666
5667
5668
5669
5670
5671
5672
5673
5674
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697

.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 COMPLE,
: RX COMPLETE OR BOTH. THE CODE REPORTS A TIME OUT
: ERROR IF NO OUTPUT INTERRUPT IS RECIEVED BEFORE
: 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 TRNASMIT EXPECTED
: 'ERX' BIT = SET IF RECIEVE 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:
: 'LGDVE' - LOG DEVICE ERROR TO EVENT LOG

```

```

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

```

```

DVTXRX: BIT #QTX,FLAG ;ANY TX TO QUE
        BEQ DVTR3 ;IF NOT GO WAIT FOR OUPUT
        BIC #QTX+#TXM+PAD,FLAG ;CLEAR FLAG
        JSR PC,CTSSR ;GO SET CTS
        TST RNODE
        BEQ DVTR1 ;IF NON-ITEP GO TO 1
        BIS #TXM,FLAG ;SET THE BODY BIT
        MOV DVTXA,MSGPTR
        MOV DVTCC,MSGCC ;AND SET UP FOR ACTUAL DATA
        BR DVTR2
        ;ENABLE TX AND TX INTER.

DVTR1: MOVB #201,HDMSG+1 ;SET UP SOH
        MOV #HDMSG+1,MSGPTR ;SET POINTER TO HEADER
        MOV DVTCC,HDACC
        MOV HDMC,MSGCC ;SET CC FOR HEADER
        MOV #177,SYNCC ;SET UP FOR 177 SYNCs.
        BIS #TXENA:#TINTEN,@TXCSR

```



```

5698 032326 012737 000074 006624 DVTR3: MOV #60.,TIMERS ;SET TIMER FOR 60 SECS
5699
5700 032334 DVTR8: BREAK
5701 032334 104422 TRAP C$BRK
5702 032336 005737 006624 TST TIMERS ;IS TIMER EXPIRED
5703 032342 001022 BNE TOINOT
5704
5705 ;LOG ERROR TIME OUT RX OR TX NOT COMPLETED
5706
5707 032344 012737 016176 006522 MOV #DVEM2,TEMP2
5708 032352 017737 156766 006524 MOV @RXCSR,TEMP3
5709 032360 017737 156766 006526 MOV @TXCSR,TEMP4
5710 032366 004737 017246 JSR PC,LGDVE
5711 032372 005237 006470 INC ERRCNT
5712 032376 ERRSOF T 7,DVEM2,ERR13
5713 032376 104457 TRAP C$ERSOF T
5714 032400 000007 .WORD 7
5715 032402 016176 .WORD DVEM2
5716 032404 016722 .WORD ERR13
5717 032406 000747 BR DVTR3 ;RETURN TO CHECK TIMER
5718
5719 032410 032737 000010 006560 TOINOT: BIT #QTX,FLAG ;IS IT TX COMPL?
5720 032416 001406 BEQ DVTR4 ;BRANCH IF TX NOT DONE.
5721 032420 004737 033650 JSR PC,CLRRTS
5722 032424 032737 000100 006560 BIT #ERX,FLAG ;ARE WE EXPECTING TO RX
5723 032432 001416 BEQ DVTREX ;BRANCH IF NOT.
5724
5725 032434 032737 000004 006560 DVTR4: BIT #QRX,FLAG ;IS RX DONE
5726 032442 001734 BEQ DVTR8 ;GO BACK AND TIME IF NOT
5727
5728 032444 032737 000200 006560 BIT #ETX,FLAG ;ARE WE EXPECTG TO TX.
5729 032452 001406 BEQ DVTREX ;BRANCH IF NOT.
5730
5731 032454 032737 000010 006560 BIT #QTX,FLAG ;IS IT TX COMPLETED
5732 032462 001724 BEQ DVTR8 ;GO BACK AND TIME OUT
5733 032464 004737 033650 JSR PC,CLRRTS ;CLEAR RTS IF NESC.
5734 032470 000207 DVTREX: RTS PC ;AND EXIT
5735

```

5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779

; DEVICE DEPENDENT SUBROUTINES

.SBTTL DEVICE INTERRUPT SERVICE ROUTINES

```

:++
: FUNCTIONAL DESCRIPTION:
: RECEIVER INTERRUPT ROUTINE. WHEN A RX INT. OCCURS
: THIS ROUTINE DECIDES IF IT IS A RX STATUS, DATA SET
: CHANGE OR DATA INTERRUPT. IF IT IS A DATA SET CHANGE
: INTERRUPT IT PUTS THE STATUS IN 'CMODS' AND COMPARES
: THAT STATUS TO THE OLD STATUS IN 'MODS'. IF THEY ARE
: THE SAME THAT MEANS THE INTERRUPT WAS CAUSED BY A GLITCH
: ON ONE OF THE LINES. IF THEY ARE DIFFERENT THEN A HARD
: MODEM ERROR HAS OCCURED. IN ANY EVENT THE MODEM STATUS
: CHANGE IS LOGGED.
: IF A DATA INT. OCCURS THE ROUTINE PUTS THE DATA AWAY
: IN A BUFFER POINTED TO BY 'RMSGPT' THE MSG. COUNT IS
: DECREMENTED BY ONE BYTE. IF COUNT IS EQUAL TO ZERO AND
: 'BCC' BIT AND 'RXM' BIT IS SET THEN RX IS DISABLED AND
: 'QRX' BIT IS SET. IF COUNT IS ZERO AND 'BCC' BIT IS SET
: BUT 'RXM' BIT IS NOT SET THEN MSG COUNT IS SET TO LENGHT
: RECD IN HEADER AND 'RMSGPT' IS SET TO RX BUFFER LOCATION
: AND 'RXM' BIT IS SET.
: IF COUNT IS EQUAL TO ZERO AND 'BCC' IS NOT SET THEN
: COUNT IS SET TO 2 AND 'RMSGPT' IS SET TO 'BCCW' AND
: 'BCC' BIT IS SET.
:
: IF A STATUS INTERRUPT OCCURS THEN OVERRUN ERROR BIT IS CHECKED.
: AN ERROR IS LOGGED AND 'QRX' IS SET AND THE RX IS DISABLED.

```

```

: INPUTS:
: RMSGPT - ADDRESS OF RX BUFFER
: RMSCC - COUNT OF DATA TO BE RXED.

```

```

: SUBORDINATE ROUTINES USED:
: 'LOGMSC' - LOG MODEM STATUS CHANGE
: 'LGDVE' - LOG DEVICE ERROR

```

5780 032472
5781 032472
5782 032472 017737 156646 011372
5783 032500 032737 000010 006554
5784 032506 001456
5785 032510 032737 002000 006560
5786 032516 001452
5787 032520 005737 011372
5788 032524 100047
5789 032526 013737 011372 011370
5790 032534 042737 106760 011370
5791 032542 032777 000040 156602

```

BGNSRV DVRXI
MOV @RXCSR,IRXCSR ;MOV RX CSR TO IMAGE
BIT #MOCHK,PARAM ;ANY MODEM CHANGES TO REPORT
BEQ RXIN21 ;IF NOT IGNORE DS CHANGE.
BIT #INOVF,FLAG ;IS INIT OVER
BEQ RXIN21 ;NO THEN IGNORE DS CHANGE.
TST IRXCSR
BPL RXIN21 ;IF DATA SET CHANGE IS NOT SET BR
MOV IRXCSR,CMODS ;MOV THE NEW MODEM STATUS IN
BIC #106760,CMODS
BIT #TM,@TXCSR
DVRXI::

```

```

5792 032550 001403          BEQ    RXIN2          ;IF TEST MODE SET
5793 032552 052737 000040 011370  RXIN2: BIS    #TM,CMODS    ;SET IT IN NEW STATUS
5794 032560 013737 011370 006524  MOV    CMODS,TEMP3
5795 032566 013737 007534 006526  MOV    MODS,TEMP4
5796 032574 023737 006526 006524  CMP    TEMP4,TEMP3    ;COMPARE OLD TO CURRENT
5797 032602 001406          BEQ    GLINC          ;INC GLITCH COUNT
5798 032604 005237 011416          INC    MHRCNT         ;INC HARD COUNT
5799 032610 012737 016005 006522  MOV    #HRDMSG,TEMP2  ;SET UP HARD MESG.
5800 032616 000405          BR     RXIN1
5801 032620 005237 011414          GLINC: INC    MGLCNT    ;INC GLITCH COUNT
5802 032624 012737 015757 006522  MOV    #GLMSG,TEMP2   ;SET UP GLITCH
5803 032632 004737 017422          FXIN1: JSR   PC,LOGMSC ;GO LOG MODEM STATUS CHANGE
5804 032636 013737 011370 007534  MOV    CMODS,MODS     ;MOVE CURRENT TO OLD
5805
5806          ;
5807          ;TEST FOR STATUS OR DATA
5808 032644 032737 002200 011372  RXIN21: BIT   #RSTARY!RDATRY,IRXCSR
5809 032652 001544          BEQ    RXINEX         ;IF NEITHER EXIT
5810 032654 017737 156470 011374  MOV    @RDSR,IRDSR
5811 032662 032737 000200 011372  BIT   #RDATRY,IRXCSR  ;IS THIS DATA
5812 032670 001455          BEQ    RXIN3         ;IF NOT GO TO 3
5813
5814          ;
5815          ;GET HERE WITH GOOD DATA
5816 032672 013702 011406          RXIN4: MOV    RMSGPT,R2
5817 032676 113722 011374          MOVB   IRDSR,(R2)+    ;STORE DATA AWAY
5818 032702 010237 011406          MOV    R2,RMSGPT     ;PUT POINTER BACK
5819
5820
5821 032706 005337 011410          DEC    RMSGCC
5822 032712 001124          BNE    RXINEX         ;GET OUT IF NOT ALL DONE
5823 032714 032737 000400 006560  BIT   #BCC,FLAG      ;IS THE BCC FLAG ALREADY SE
5824 032722 001066          BNE    RXIN5         ;BRANCH IF YES.
5825 032724 032737 100000 011374  BIT   #RERR,IRDSR    ;IS THE ERR CHK BIT SET INDICATING
5826          ;GOOD BCC.
5827 032732 001022          BNE    RXIN6         ;BRANCH IF GOO
5828 032734 013737 011374 006524  MOV    IRDSR,TEMP3
5829 032742 013737 011372 006526  MOV    IRXCSR,TEMP4
5830 032750 012737 016273 006522  MOV    #DVEM3,TEMP2
5831 032756 004737 017246          JSR   PC,LGDVE
5832 032762 005237 006470          INC    ERRCNT
5833 032766          ERRSOFT 8,DVEM3,ERR13
5834 032766 104457          TRAP   CSERSOFT
5835 032770 000010          .WORD 8
5836 032772 016273          .WORD DVEM3
5837 032774 016722          .WORD ERR13
5838
5839 032776 000467          BR     RXIN8         ;DISABLE INTERRUPTS AND EXIT
5840
5841 033000 052737 000400 006560  RXIN6: BIS    #BCC,FLAG  ;SET FLAG
5842 033006 012737 000002 011410  MOV    #2.,RMSGCC     ;SET THE COUNT TO 2
5843 033014 012737 011412 011406  MOV    #BCCW,RMSGPT   ;SET POINTER TO BCC WORD
5844 033022 000460          BR     RXINEX
5845
5846          ;STATUS CHECK
5847

```



```

5848 033024 032737 002000 011372 RXIN3: BIT #RSTARY,IRXCSR ;IS THIS A STATUS INT.
5849 033032 001454 BEQ RXINEX ;EXIT IF NOT
5850
5851 ;LOG OVERRUN ERROR
5852
5853 033034 012737 016333 006522 MOV #DVEM4,TEMP2
5854 033042 013737 011374 006524 MOV IRDSR,TEMP3
5855 033050 013737 011372 006526 MOV IRXCSR,TEMP4
5856 033056 004737 017246 JSR PC,LGDVE
5857 033062 005237 006470 INC ERRCNT
5858 033066 ERRSOFT 9,DVEM4,ERR13
5859 033066 104457 TRAP CSERSOFT
5860 033070 000011 .WORD 9
5861 033072 016333 .WORD DVEM4
5862 033074 016722 .WORD ERR13
5863 033076 000424 BR RXIN7
5864
5865 033100 032737 000040 006560 RXIN5: BIT #RXM,FLAG ;IS THE RX M BODY BIT SET
5866 033106 001020 BNE RXIN7 ;IF YES THEN ALL DONE
5867 033110 052737 000040 006560 BIS #RXM,FLAG
5868 033116 042737 000400 006560 BIC #BCC,FLAG ;CLEAR BCC AND SET RXM
5869 033124 013737 006452 011406 MOV DVRXA,RMSGPT ;MOVE ADDRESS TO POINTER
5870 033132 013737 002660 011410 MOV RDMCC,RMSGCC ;MOVE THE CHAR COUNT IN
5871 033140 013737 002660 006454 MOV RDMCC,DVRCC ;SET THE CC TO AMOUNT IN HEADER
5872 033146 000406 BR RXINEX ;AND FINISH.
5873
5874 033150 052737 000004 006560 RXIN7: BIS #QRX,FLAG ;SET FLAG BIT
5875
5876 033156 042777 000120 156160 RXIN8: BIC #RINTEN+RXENA,@RXCSR ;CLEAR INTAND RX ENABLE
5877
5878 033164 RXINEX:
5879 033164 ENDSRV
5880 033164
5881 033164 000002 L10020: RTI
  
```

5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925
5926
5927
5928
5929
5930
5931
5932
5933
5934
5935
5936
5937

```
.SBTTL                DEVICE TRANSMIT INTERRUPT ROUTINE

:++
: FUNCTIONAL DESCRIPTION:
:   DEVICE TRANSMIT INT. ROUTINE
:
:   WHEN A TRANSMIT BUFFER EMPTY CAUSES AN INTERRUPT TO OCCUR
:   THE PROGRAM COMES TO THIS ROUTINE.
:   IF THE SYNC COUNT 'SYNCC' IS NON ZERO TSOM IS SET
:   A SYNC CHAR IS LOADED TO TDSR AND THE SYNC COUNT IS
:   DECREMENTED.
:
:   IF THE SYNC COUNT IS ZERO TSOM AND TEOM ARE RESET
:   AND THE 'PAD' BIT IN FLAG WORD IS CHECKED IF IT IS
:   SET THEN A PAD(377) CHAR IS LOADED TO TDSR AND TX
:   INTERRUPT ENABLE IS CLEARD.
:
:   IF THE SYNC COUNT IS ZERO AND THE 'PAD' FLAG IS
:   CLEAR THEN A BYTE IS PUT IN TDSR FROM THE ADDRESS
:   IN MSGPTR AND THE MSG COUNT IS DECREMENTED
:
:   IF THE MSG COUNT GOES TO ZERO THE 'TXM' BIT IS
:   CHECKED IF IT IS SET THE 'PAD' FLAG IS SET
:   IF IT IS CLEAR THEN IT GETS SET AND MSGPTR IS
:   LOADED WITH THE ADDRESS OF TXBUFF AND THE MSG
:   COUNT IS LOADED WITH THE COUNT OF THE MSG TO
:   BE TRANSMITTED.
:
: INPUTS:
:   MSGPTR - IS SET TO THE ADDRESS OF THE MSG OR HEADER TO BE TX'D
:   MSGCC  - IS SET TO THE COUNT OF MSG TO BE TX'D
:
: OUTPUTS:
:   QTX - THIS BIT IS SET WHEN MSG IS TX'D OK.
:--

BGNSRV  DVTXI
DVTXI::
TST     SYNCC           ;ANY SYNC'S TO SEND
BEQ     TXIN1           ;IF NOT GO TO 1
MOV     SYNCW,@TDSR    ;ELSE SET TSOM AND SYNC WORD
DEC     SYNCC           ;DEC SYNC COUNT
BNE     TXINEX         ;IF NOT ZERO EXIT
TXIN1:  BIC     #TEOM!TSOM,@TDSR
BIT     #PAD,FLAG      ;IS THE PAD BIT SET
BEQ     TXIN2           ;GO TO 2 IF NOT SET
MOV     #377,@TDSR     ;LOAD FF TO TX DATA REG.
BIC     #TINTEN,@TXCSR ;CLEAR TX INT ENABLE
BIS     #QTX,FLAG      ;SET THE TX COMPLETE IN FLAG
BR      TXINEX         ;AND EXIT
TXIN2:  MOV     MSGPTR,R2 ;LOAD R2 WITH TX ADDR.
        MOVB    (R2)+,@TDSR ;LOAD DATA BYTE
        MOV     R2,MSGPTR ;RESTORE POINTER
        DEC     MSGCC      ;DEC CC
        BNE     TXINEX
        BIS     #TEOM,@TDSR
        BIT     #TXM,FLAG  ;IS THIS THE END OF DATA MSG.
```

```
033166
033166 005737 011402
033172 001406
033174 013777 011404 156152
033202 005337 011402
033206 001056
033210 042777 001400 156136 TXIN1:
033216 032737 001000 006560
033224 001412
033226 012777 000377 156120
033234 042777 000100 156110
033242 052737 000010 006560
033250 000435
033252 013702 011376 TXIN2:
033256 112277 156072
033262 010237 011376
033266 005337 011400
033272 001024
033274 052777 001000 156052
033302 032737 000020 006560
```

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 144
N 11
DEVICE TRANSMIT INTERRUPT ROUTINE

SEQ 0143

```
5938 033310 001012          BNE    TXIN3      ;IF SO SET THE PAD BIT
5939 033312 052737 000020 006560  BIS    #TXM,FLAG  ;IF NOT MUST BE END OF HEADER
5940 033320 013737 006436 011376  MOV    DVTXA,MSGPTR ;SO SET UP MSGPTR FOR MSG
5941 033326 013737 006440 011400  MOV    DVTCC,MSGCC  ;AND THE CC FOR MSG.
5942 033334 000403          BR     TXINEX     ;
5943 033336 052737 001000 006560  TXIN3: BIS    #PAD,FLAG ;SET THE PAD BIT
5944
5945 033344          TXINEX:
5946 033344          ENDSRV
5947 033344
5948 033344 000002
```

L10021: RTI

5949
5950
5951
5952
5953
5954
5955
5956
5957
5958
5959
5960
5961
5962
5963
5964
5965
5966
5967
5968
5969
5970
5971
5972
5973
5974
5975
5976
5977

.SBTTL DEVICE TRANSMIT CONTROL MSG

```

++
FUNCTIONAL DESCRIPTION:
THIS ROUTINE DOES THE FOLLOWING
QUES A RX SPACE AT RHDMSG+1
QUES A TX MSG FROM HDMSG+1
CHECKS FOR A TIMER EXPIRED
IF EXPIRED RETURN TO CALLER
ELSE CHECK FOR A TX MSG COMPLETED
IF TX COMPLETED CHECK FOR RX COMPLETED
ELSE RECHECK TIMER AND TX COMPLETED UNTIL
EITHER TX COMPLETE OR TIME OUT
IF TX COMPLETE AND RX NOT COMPLETE THEN
REQUE TX MSG.
ELSE IF RX COMPLETE RETURN.
    
```

```

INPUTS:
TXM - SET IN FLAG WORD
HDMSG+2 - TYPE OF CONTROL MSG..
    
```

```

SUBORDINATE ROUTINES USED:
"CLRRTS" - CLEAR REQUEST TO SEND IF HALF DUP.
    
```

```

CALLING SEQUENCE:
JSR PC,DVIN31
    
```

```

RETURN:
RETURN TO CALLER IF SOMETHING RX'D OR TIMER OUT.
    
```

--

```

5978 033346 042737 000004 006560 DVIN31: BIC #QRX,FLAG ;CLEAR RX COMPLE.
5979
5980 033354 012737 002657 011406 MOV #RHDMSG+1,RMSGPT ;SET UP POINTER
5981 033362 013737 002654 011410 MOV HDMC,RMSGCC ;AND CC
5982 ;ENABLE RCVR.
5983
5984 033370 052777 000160 155746 BIS #RINTEN!RXENA!DSITEN,@RXCSR
5985 ;SET UP TRANSMITTER TO SEND
5986
5987
5988 033376 004737 033476 DVIN32: JSR PC,CTSSR ;SET RTS
5989 033402 042737 001010 006560 BIC #QTX!PAD,FLAG ;CLEAR TX COMPT FLAG.
5990 033410 012737 002645 011376 MOV #HDMSG+1,MSGPTR ;MOVE THE CURRENT POINTER TO MSGPTR.
5991 033416 013737 002654 011400 MOV HDMC,MSGCC
5992 033424 012737 000010 011402 MOV #8.,SYNCC ;SET UP SYNC COUNT
5993 033432 052777 000120 155712 BIS #TXENA!TINTEN,@TXCSR
5994 ;NOW WAIT FOR TIME OUT OR TX COMPLETE
5995
5996
5997 033440 DVIN35: BREAK
5998 033440 104422 TRAP CSBRK
5999 033442 005737 006624 TST TIMERS ;IS IT TIMED OUT
6000 033446 001412 BEQ DVIN34 ;IF YES EXIT
6001 033450 032737 000010 006560 BIT #QTX,FLAG ;IS TX DONE
6002 033456 001770 BEQ DVIN35 ;IF NOT GO BACK AND CK TIME OUT
6003 033460 004737 033650 JSR PC,CLRRTS ;GO CLEAR RTS IF NESC.
6004 033464 032737 000004 006560 BIT #QRX,FLAG ;DID WE RX ANYTHING
    
```

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 146
DEVICE TRANSMIT CONTROL MSG

C 12

SEQ 0145

6005 033472 001741
6006 033474 000207
6007

DVIN34: BEQ DVIN32
RTS PC

;IF NOT RETRANSMIT LAST
;RETURN TO CALLER

6008
6009
6010
6011
6012
6013
6014
6015
6016
6017
6018
6019
6020
6021
6022
6023
6024
6025
6026
6027
6028
6029
6030
6031
6032
6033
6034
6035
6036
6037
6038
6039
6040
6041
6042
6043
6044
6045
6046
6047
6048
6049
6050
6051
6052
6053
6054
6055
6056
6057
6058
6059
6060
6061
6062
6063

```

.SBTTL                DEVICE RTS TO CTS DELAY
:++
: FUNCTIONAL DESCRIPTION:
:   CTSSR--THIS ROUTINE SETS REQUEST TO SEND TO MODEM
:   AND CHECKS FOR CLEAR TO SEND TO COME BACK
:   IF CTS DOES NOT COME BACK BEFORE TIMER EXPIRES
:   AND ERROR IS REPORTED AND WE TRY AGAIN.
:   THE ROUTINE IS SKIPPED IF INTERNAL LOOP IS SET.
:
:
: OUTPUTS:
:
: SUBORDINATE ROUTINES USED:
:   'LGDVE' - LOG DEVICE ERROR
:
: CALLING SEQUENCE:
:   JSR     PC,CTSSR
:--
CTSSR:  CMP     #1,MLTYP      ;IS THIS TTL LOOP
        BEQ     DVTXR9      ;BR IF YES
        ;SET RTS AND WAIT FOR CTS
DVTXR3: BIT     #FIRST,FLAG  ;IF NOT FIRST TIME SKIP DELY
        BNE     CTSS3
        MOV     #0,TEMP
CTSS4:  INC     TEMP
        BREAK
        TRAP    C$BRK
        TST     TEMP
        BNE     CTSS4      ;IF NOT ZERO GO BACK
        BIS     #FIRST,FLAG ;SET FIRST FLAG.
CTSS3:  BIS     #RTS,@RXCSR  ;SET REQUEST TO SEND
        MOV     #1000.,TIMER1 ;SET UP TIMER
DVTXR2: BREAK
        TRAP    C$BRK
        BIT     #CTS,@RXCSR  ;IS CLEAR TO SEND BACK
        BNE     DVTXR1      ;BR. IF CTS IS SET
        TST     TIMER1      ;ELSE TEST IF TIME EXPIRED
        BNE     DVTXR2      ;BR IF TIME NOT EXPRIED.
        ;SET ERROR FOR NO CTS
        MOV     #DVEM1,TEMP2
        MOV     @RXCSR,TEMP3
        MOV     @TXCSR,TEMP4
        JSR     PC,LGDVE
        INC     ERRCNT
        ERRSOFT 6,DVEM1,ERR13
        TRAP    C$ERSOFT
        .WORD  6
        .WORD  DVEM1
        .WORD  ERR13
        BR     DVTXR3      ;THEN TRY TO SET RTS AGAIN
    
```


CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 148
E 12
DEVICE RTS TO CTS DELAY

SEQ 0147

6064 033646
6065 033646 000207

DVTXR1:
DVTXR9: RTS PC ;

6066
6067
6068
6069
6070
6071
6072
6073
6074
6075
6076
6077
6078
6079
6080
6081
6082
6083
6084
6085

033650
033650 005737 006552
033654 001011
033656 012737 002000 006522
033664 005337 006522
033670 001375
033672 042777 000004 155444
033700 000207

```
.SBTTL                                DEVICE CLEAR REQUEST TO SEND
:++
: FUNCTIONAL DESCRIPTION:
:   THIS ROUTINE CLEARS REQUEST TO SEND IF
:   IN HALF DUPLEX MODE
: CALLING SEQUENCE:
:   JSR    PC,CLRRTS
:--

CLRRTS:
      TSI    FHDPLX                ;IS THIS FULL DUPLEX
      BNE    DVTR5                ;BRANCH IF YES
      MOV    #2000,TEMP2
CLRR1: DEC    TEMP2
      BNE    CLRR1                ;DELAY A WHILE TO ALLOW CRC
                                      ;TO BE TXMITT'D BEFORE YOU
      BIC    #RTS,@RXCSR          ;CLEAR REQUEST TO SEND
DVTR5: RTS    PC                  ;RETURN TO CALLER
```

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 150
G 12
DEVICE CLEAR REQUEST TO SEND

SEQ 0149

6086
6087
6088
6089 033702
6090 033702
6091 033702 104401
6092

.EVEN
ENDTST

L10017: TRAP CSETST

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 151
H 12
DEVICE CLEAR REQUEST TO SEND

SEQ 0150

6043
6094

6095
6096
6097
6098
6099
6100
6101
6102
6103
6104
6105
6106
6107
6108
6109
6110
6111
6112
6113
6114
6115
6116
6117
6118
6119
6120
6121
6122
6123
6124
6125
6126
6127
6128
6129
6130
6131
6132
6133
6134
6135
6136
6137
6138
6139
6140
6141
6142
6143

.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

033704
033704 000016
033706

.WORD L10022-L\$HARD/2
L\$HARD::

.SBTTL DEVICE INDEPENDENT SECTION

GPRML DPLX,0,1,YES

033706
033706 000130
033710 033742
033712 000001

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

.SBTTL DEVICE DEPENDENT SECTION

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

033714
033714 001031
033716 033773
033720 160000
033722 177776

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

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

033724
033724 002031
033726 034021
033730 000300
033732 000776

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

GPRML RNODM,14,1,YES

033734
033734 006130
033736 034054
033740 000001

.WORD T\$CODE
.WORD RNODM
.WORD 1

ENDHRD

033742
033742

.EVEN
L10022:

.NLIST BEX

:DEVICE INDEPENDENT QUESTIONS

033742 052506 046114 042040 DPLX: .ASCIZ /FULL DUPLEX OPERATION : /

:DEVICE DEPENDENT QUESTION

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

J 12
MACY11 30A(1052) 25-JUL-80 10:29 PAGE 153
DEVICE DEPENDENT SECTION

SEQ 0152

033773	104	053105	041511	CSRADR: .ASCIZ	/DEVICE CSR ADDRESS : /
034021	111	052116	051105	VECTOR: .ASCIZ	/INTERRUPT VECTOR ADDRESS: /
034054	042522	047515	042524	RNODM: .ASCIZ	/REMOTE NODE "ITEP":/
				.LIST	REX
					.EVEN

6144
6145
6146

6147
6148
6149
6150
6151
6152
6153
6154
6155
6156
6157
6158
6159
6160
6161
6162
6163
6164
6165
6166
6167
6168
6169
6170
6171
6172
6173
6174
6175
6176
6177
6178
6179
6180
6181

034100
034100 000030

034160

034160 000000
034162 000000
034164
034164

000001

;.SBTTL SOFTWARE PARAMETER CODING SECTION

;++
: THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
: THAT ARE USED BY THE SUPERVISOR TO BUILD F-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

.EVEN 0
.WORD 0
.WORD 0

L\$LAST:: ENDMOD

.END

ACT = 000003	1546#	4321	4736	4816	5041
ACTATV 026430	4526	4736#			
ACTBCR 026272	4544	4708#			
ACTCHK 026644	4506	4782#			
ACTCLB 025614	4597	4611#			
ACTCLP 026756	4540	4810#			
ACTCLR 025314	4504	4555#			
ACTCOP 026112	4514	4671#			
ACTCRC 026660	4535	4788#			
ACTCSE 025440	4509	4583#			
ACTCST 025532	4510	4599#			
ACTDLL 026476	4530	4750#			
ACTDME 026040	4546	4652	4655#		
ACTDMQ 026032	4547	4654#			
ACTDMS 026010	4545	4649#			
ACTDMX 026046	4656#				
ACTECH 026554	4534	4766#			
ACTEQO 026234	4518	4697#			
ACTHLP 025334	4508	4561#			
ACTLIS 026466	4529	4747#			
ACTLLP 026766	4541	4812#			
ACTLPX 027004	4807	4809	4811	4813	4816#
ACTLXX 027046	4780	4801	4804	4817	4826#
ACTMEX 026422	4690	4706	4728	4733#	
ACTME1 026414	4717	4719	4721	4723	4725 4732#
ACTMOP 026736	4538	4806#			
ACTMOS 026666	4549	4791#			
ACTMSO 026314	4519	4716#			
ACTMS1 026322	4520	4718#			
ACTMS2 026332	4521	4720#			
ACTMS3 026342	4522	4722#			
ACTMS4 026352	4523	4724#			
ACTMS5 026362	4524	4726#			
ACTMS6 026400	4525	4729#			
ACTM2X 026524	4737	4745	4748	4751	4754 4758#
ACTNO 026544	4533	4763#			
ACTNUF 025304	4543	4552#			
ACTNUL 025312	4503	4553#			
ACTNUM 026122	4515	4674#			
ACTOPM 026214	4516	4692#			
ACTPAS 026440	4527	4739#			
ACTPRO 026674	4536	4794#			
ACTPRT 025400	4548	4574#			
ACTQFG 026700	4783	4786	4789	4792	4796#
ACTREC 026460	4528	4744#			
ACTRLP 026776	4542	4814#			
ACTRPS 026726	4537	4803#			
ACTRUN 025414	4507	4578#			
ACTSHO 025324	4505	4558#			
ACTSHW 025654	4589	4603	4622#	4641	
ACTSTE 026054	4511	4661#			
ACTSTS 026652	4517	4785#			
ACTSTT 026064	4512	4664#			
ACTSTX 026072	4662	4665#			
ACTSZE 026102	4513	4668#			
ACTTAL 026516	4532	4756#			

ACTTLP	026746	4539	4808#							
ACTTRA	026506	4531	4753#							
ADDCC	021104	3561#	4436	4478						
ADDC1	021200	3563	3578#							
ADR	= 000020	1519#								
ALCK	027376	1976	4951#							
ALCK1	027504	5006	5015#	5061						
ALCK2	027550	5016	5025#	5093						
ALCK2A	027770	5058	5063#							
ALCK3	030014	5028	5065	5068#						
ALCK3A	030106	5075	5081#							
ALCK3B	030120	5082	5084#							
ALCK3C	030114	5079	5083#	5090						
ALCK4	030142	5069	5088#							
ALCK4A	030150	5087	5091#							
ALCK4B	030162	5092	5094#							
ALCK5	027434	5005#	5083							
ALLTR	027434	4883	4906	4930	5004#					
ASSEMB	= 000010	1291								
ATVMOD	= 000027	1654#	2117							
BAD	006537	1947#	2909	5169*						
BADCHR	= 000051	1672#	2212							
BASM1	014571	2665#	3505							
BASM2	014562	2663#	3524							
BASM3	014553	2661#	3516							
BCC	= 000400	1608#	5626	5629	5823	5841	5868			
BCCW	011412	2286#	5843							
BDCLK	013310	2526#	4080							
BIT0	= 000001	1492#	1566	1596	1703					
BIT00	= 000001	1481#	1492							
BIT01	= 000002	1480#	1491							
BIT02	= 000004	1479#	1490							
BIT03	= 000010	1478#	1489							
BIT04	= 000020	1477#	1488							
BIT05	= 000040	1476#	1487							
BIT06	= 000100	1475#	1486							
BIT07	= 000200	1474#	1485							
BIT08	= 000400	1473#	1484							
BIT09	= 001000	1472#	1483							
BIT1	= 000002	1491#	1567	1597	1696	1704				
BIT10	= 002000	1471#	1612	1701						
BIT11	= 004000	1470#	1614	1702						
BIT12	= 010000	1469#	1687							
BIT13	= 020000	1468#	1685							
BIT14	= 040000	1467#	1689							
BIT15	= 100000	1466#	1711	1712						
BIT2	= 000004	1490#	1568	1598	1688	1705				
BIT3	= 000010	1489#	1569	1599	1706					
BIT4	= 000020	1488#	1570	1604	1697	1707				
BIT5	= 000040	1487#	1571	1606	1690	1691	1693	5594	5595	5597
BIT6	= 000100	1486#	1600	1699	1708					
BIT7	= 000200	1485#	1601	1700	4845	4848				
BIT8	= 000400	1484#	1608	1709						
BIT9	= 001000	1483#	1610	1636	1710	5504				
BLDBEX	021312	3622	3626#							
BLDBUF	021202	3604#	4312	4319	4437	4479	4618	4857		

LOGS2	017666	3254	3258#						
LOGS3	017472	3173	3182	3187	3191	3195	3199	3205	3217#
LOGS4	017546	3225	3234#						
LOGS5	017572	3219	3221	3242#					
LOGTXC	017202	3157#	5072	5340					
LOGTXQ	017164	3152#	5023	5332					
LOGUNT	006540	1951#	4095*	4097*	4098	4102			
LOOPS	003102	1872#	3661						
LOT =	000010 G	1518#							
LPO	013036	1872	2475#	3660					
LPO0	013037	2476#	3657						
LP1	013046	1873	2478#						
LP2	013057	1874	2480#						
LP3	013065	1875	2482#						
LP4	013100	1876	2484#						
LSACP	002110 G	1385#							
LSAPT	002036 G	1343#							
LSAU	023612 G	1370	4243#						
LSAUT	002070 G	1369#							
LSAUTO	023560 G	1386	4180#						
LSCCP	002106 G	1383#							
LSCLEA	023562 G	1384	4194#						
LSCO	002032 G	1339#							
LSDEPO	002011 G	1321#							
LSDESC	011432 G	1376	2318#						
LSDESP	002076 G	1375#							
LSDEVP	002060 G	1361#							
LSDISP	002124 G	1346	1405#						
LSDLY	002116 G	1391#							
LSDTP	002040 G	1345#							
LSDTYP	002034 G	1341#							
LSDU	023604 G	1372	4221#						
LSDUT	002072 G	1371#							
LSDVTY	011422 G	1362	2308#						
LSEF	002052 G	1356#							
LSENV1	002044 G	1349#							
LSETP	002102 G	1379#							
LSEXP1	002046 G	1351#							
LSEXP4	002064 G	1365#							
LSEXP5	002066 G	1367#							
LSHARD	033706 G	1328	6108	6109#					
LSHIME	002120 G	1393#							
LSHPCP	002016 G	1327#							
LSHPTP	002022 G	1331#							
LSHW	002130 G	1332	1418	1419#					
LSICP	002104 G	1381#							
LSINIT	022734 G	1382	4006#						
LSLADP	002026 G	1335#							
LSLAST	034164 G	1336	6178#						
LSLOAD	002100 G	1377#							
LSLUN	002074 G	1373#							
LSMREV	002050 G	1353#							
LSNAME	002000 G	1310#							
LSPRIO	002042 G	1347#							
LSPROT	022726 G	1388	3990#						
LSPRT	002112 G	1387#							

NOD130	011072	2208#
NOD131	011076	2209#
NOD132	011102	2210#
NOD133	011106	2211#
NOD134	011112	2212#
NOD135	011114	2215#
NOD136	011120	2216#
NOD137	011124	2217#
NOD14	010012	2104#
NOD140	011140	2218#
NOD141	011144	2219#
NOD142	011160	2220#
NOD143	011164	2223#
NOD144	011170	2224#
NOD145	011174	2225#
NOD146	011200	2228#
NOD147	011204	2231#
NOD15	010016	2105#
NOD150	011226	2232#
NOD151	011232	2233#
NOD152	011246	2234#
NOD153	011252	2235#
NOD154	011274	2236#
NOD155	011300	2237#
NOD156	011322	2238#
NOD157	011326	2241#
NOD16	010022	2106#
NOD160	011332	2242#
NOD161	011336	2243#
NOD162	011342	2248#
NOD17	010034	2107#
NOD2	007702	2094#
NOD20	010040	2108#
NOD21	010052	2109#
NOD22	010056	2110#
NOD23	010060	2114#
NOD24	010064	2115#
NOD25	010100	2116#
NOD26	010104	2117#
NOD27	010122	2118#
NOD3	007704	2095#
NOD30	010126	2119#
NOD31	010144	2120#
NOD32	010150	2121#
NOD33	010166	2122#
NOD34	010172	2123#
NOD35	010210	2124#
NOD36	010214	2125#
NOD37	010240	2126#
NOD4	007720	2096#
NOD40	010244	2127#
NOD41	010250	2128#
NOD42	010266	2129#
NOD43	010272	2130#
NOD44	010304	2131#
NOD45	010310	2135#

2998	3001	3004	3005	3114	3229	3230	3231	3232	3233	3237	3238	3239
3240	3241	3277	3278	3279	3280	3281	3300	3301	3302	3303	3304	3310
3311	3312	3313	3314	3315	3316	3317	3318	3325	3326	3327	3328	3329
3330	3331	3339	3340	3341	3342	3343	3344	3346	3347	3348	3349	3350
3351	3352	3369	3370	3371	3372	3373	3374	3375	3376	3385	3386	3387
3388	3389	3390	3391	3392	3400	3401	3402	3403	3404	3405	3406	3407
3417	3418	3419	3420	3421	3422	3426	3427	3428	3429	3430	3446	3447
3448	3449	3450	3466	3467	3468	3469	3470	3504	3505	3506	3507	3508
3509	3514	3515	3516	3517	3518	3519	3520	3523	3524	3525	3526	3527
3528	3568	3569	3570	3571	3572	3664	3665	3666	3667	3668	3669	3670
3671	3672	3695	3696	3697	3698	3699	3700	3701	3702	3703	3845	3846
3847	3848	3849	3888	3889	3890	3891	3892	3981	4011	4012	4014	4016
4017	4019	4021	4022	4024	4027	4028	4030	4038	4039	4040	4042	4048
4049	4050	4052	4061	4063	4068	4069	4070	4071	4072	4073	4074	4075
4080	4081	4082	4083	4084	4102	4103	4104	4106	4133	4134	4135	4136
4137	4138	4143	4144	4145	4146	4147	4148	4150	4151	4152	4153	4154
4155	4158	4159	4161	4162	4169	4185	4198	4199	4201	4204	4205	4212
4225	4226	4233	4247	4248	4255	4285	4286	4287	4288	4289	4327	4328
4329	4330	4331	4339	4341	4345	4346	4355	4356	4357	4358	4359	4360
4361	4362	4372	4373	4374	4375	4376	4381	4382	4383	4384	4385	4409
4410	4411	4412	4413	4414	4424	4425	4426	4427	4428	4429	4450	4451
4452	4453	4454	4455	4466	4467	4468	4469	4470	4471	4563	4564	4565
4566	4567	4568	4632	4633	4634	4635	4636	4637	4638	4679	4680	4681
4682	4683	4709	4710	4711	4712	4713	4772	4773	4774	4775	4776	4821
4822	4823	4824	4825	5154	5155	5156	5157	5172	5173	5174	5175	5185
5186	5187	5188	5228	5229	5230	5231	5279	5280	5281	5282	5283	5312
5313	5314	5315	5316	5317	5318	5319	5370	5371	5372	5373	5374	5393
5394	5395	5396	5397	5448	5459	5460	5461	5462	5496	5497	5498	5499
5502	5537	5560	5561	5562	5563	5701	5713	5714	5715	5716	5834	5835
5836	5837	5859	5860	5861	5862	5881	5948	5998	6038	6045	6059	6060
6061	6062	6091	6108	6115	6116	6117	6125	6126	6127	6128	6130	6131
6132	6133	6135	6136	6137	6140	6175	6176	6177				
1291#												
SVCSUB= 000001	1291#											
SVCTAG= 000001	1291#	1446	2919	2932	2946	2960	2980	3000	3113	3980	4076	4168
	4211	4232	4254	4363	5320	5880	5947	6090	6141			4184
SVCTST= 000001	1291#	4270										
SYN = 000226	1713#											
SYNCC 011402	2282#	5532*	5695*	5919	5922*	5992*						
SYNCW 011404	2283#	4844*	4849*	5921								
SLSYM= 010000	1291#	1447#	2920#	2933#	2947#	2961#	2981#	3001#	3114#	3981#	4069	4076
	4169#	4185#	4212#	4233#	4255#	4356	4363	4364#	5313	5320	5321#	5881#
	6091#	6142#										4077#
S1 022776	4024	4026#										5948#
S2 023052	4042	4047#										
S3 023122	4052	4060#										
S4 023170	4063	4079#										
TABEX 013415	2539#	4424	4466									
TAL = 000005	1548#	4756	5402									
TALCK 030602	1978	5305#	5342	5344								
TALMOD= 000035	1660#	2130										
TBMT = 000004	1705#											
TCURAD 006450	1913#	4307*	4310	4435	4440*	4609*						
TDSR 011354	2259#	4122*	4123*	5921*	5924*	5927*	5932*	5936*				
TEMP 006516	1938#	3154*	3159*	3164*	3168*	3172*	3177*	3186*	3190*	3194*	3198*	3204*
		3244	3245*	3246*	3247	3512*	3515	3574*	3575*	3576	3614*	3615*
		3667	3675*	3678*	3698	4630*	4632	4693*	4694*	4698	4701	5046*
												5047*
												3218
												3654*
												5051

SVCSUB= 000001
SVCTAG= 000001
SVCTST= 000001
SYN = 000226
SYNCC 011402
SYNCW 011404
SLSYM= 010000
S1 022776
S2 023052
S3 023122
S4 023170
TABEX 013415
TAL = 000005
TALCK 030602
TALMOD= 000035
TBMT = 000004
TCURAD 006450
TDSR 011354
TEMP 006516

TSSAUT= 010013	4180#	4184												
TSSCLE= 010014	4194#	4204	4211											
TSSDU = 010015	4221#	4225	4232											
TSSHAR= 010022	6108#	6141												
TSSHW = 010000	1418#	1446												
TSSINI= 010012	4006#	4161	4168											
TSSMSG= 010006	2906#	2919	2923#	2932	2936#	2946	2950#	2960	2970#	2980	2989#	3000	3004	
TSSPRO= 010011	3990#													
TSSRPT= 010010	3970#	3980												
TSSSRV= 010021	3088#	3113	5781#	5880	5918#	5947								
TSSTES= 010017	4271#	4345	6090											
T1 = 023620 G	1406	4270#												
UAM = 000200 G	1522#													
UPTABL 027604	5032#													
UPTA1 027672	5038	5045#												
UPTA3 027670	5042	5044#												
UPTA4 027630	5033	5037#												
UPTEX 027742	5044	5056#												
VECTOR 034021	6131	6143#												
X\$ = 000163	1294#	2092#	2093#	2094#	2095#	2096#	2097#	2098#	2099#	2100#	2101#	2102#	2103#	
	2104#	2105#	2106#	2107#	2108#	2109#	2110#	2114#	2115#	2116#	2117#	2118#	2119#	
	2120#	2121#	2122#	2123#	2124#	2125#	2126#	2127#	2128#	2129#	2130#	2131#	2135#	
	2136#	2137#	2138#	2139#	2144#	2145#	2146#	2147#	2148#	2151#	2152#	2153#	2154#	
	2155#	2156#	2157#	2158#	2161#	2162#	2163#	2164#	2165#	2166#	2169#	2170#	2171#	
	2172#	2176#	2177#	2180#	2181#	2183#	2184#	2187#	2190#	2191#	2192#	2193#	2194#	
	2195#	2196#	2197#	2198#	2199#	2200#	2201#	2202#	2203#	2206#	2207#	2208#	2209#	
	2210#	2211#	2212#	2215#	2216#	2217#	2218#	2219#	2220#	2223#	2224#	2225#	2228#	
	2231#	2232#	2233#	2234#	2235#	2236#	2237#	2238#	2241#	2242#	2243#	2248#		
X\$ALWA= 000000	1291#													
X\$FALS= 000040	1291#													
X\$OFFS= 000400	1291#													
X\$TRUE= 000020	1291#													
\$PATCH 034100	6170#													
. = 034164	1291#	1799#	1805#	1813#	1825#	1834#	1838#	1862#	1891#	1892#	1893#	1894#	1895#	
	1896#	1899#	1999#	2000#	2095#	2101#	2108#	2115#	2117#	2123#	2125#	2136#	2138#	
	2163#	2165#	2169#	2171#	2176#	2183#	2190#	2192#	2194#	2196#	2198#	2217#	2219#	
	2235#	2311#	2780#	3005	4162	4205	4226	4248	4346	6171#				

MSEXCP	1#	1291#	4071#	4358#	5315#	6125#	6130#										
MSEXIT	1#	1291#	3004#	4161#	4162	4204#	4205	4225#	4247#	4345#	4346						
MSEXSE	1#	1291#	3004#	4161#	4204#	4225#	4247#	4345#									
MSEXTJ	1#	1291#	3004#	3005	4161#	4204#	4225#	4226	4247#	4248	4345#						
MSGEN	1#	1291#	1310#	1319#	1321#	1323#	1325#	1327#	1329#	1331#	1333#	1335#	1337#	1339#	1341#		
	1343#	1345#	1347#	1349#	1351#	1353#	1356#	1359#	1361#	1363#	1365#	1367#	1369#	1371#	1373#		
	1375#	1377#	1379#	1381#	1383#	1385#	1387#	1389#	1391#	1393#	1405#	1419#	1420#	1446#	2308#		
	2318#	2906#	2919#	2923#	2932#	2936#	2946#	2950#	2960#	2970#	2980#	2989#	3000#	3088#	3113#		
	3970#	3980#	3990#	4006#	4076#	4168#	4180#	4184#	4194#	4211#	4221#	4232#	4243#	4254#	4270#		
	4363#	5320#	5781#	5880#	5918#	5947#	6090#	6109#	6141#	6178#							
MSGENB	1#	1291#	4068#	4069	4355#	4356	5312#	5313									
MSGETS	1#	1291#	1446#	2919#	2932#	2946#	2960#	2980#	3000#	3113#	3980#	3997#	4168#	4184#	4211#		
	4232#	4254#	5880#	5947#	6090#	6140#	6180#										
MSGETT	1#	1291#	3004#	4161#	4204#	4225#	4247#	4345#									
MSGNGB	1#	1291#	1293#	1310#	1319#	1321#	1323#	1325#	1327#	1329#	1331#	1333#	1335#	1337#	1339#		
	1341#	1343#	1345#	1347#	1349#	1351#	1353#	1356#	1359#	1361#	1363#	1365#	1367#	1369#	1371#		
	1373#	1375#	1377#	1379#	1381#	1383#	1385#	1387#	1389#	1391#	1393#	1404#	1405	1418#	1419		
	1420	2308#	2318#	2906#	2923#	2936#	2950#	2970#	2989#	3088#	3970#	3990#	4006#	4180#	4194#		
	4221#	4243#	5781#	5918#	6108#	6109	6175#	6178									
MSGNIN	1#	1291#	1310#	1311	1312	1313	1314	1315	1316#	1317#	1318#	1319#	1320	1321#	1322		
	1323#	1324	1325#	1326	1327#	1328	1329#	1330	1331#	1332	1333#	1334	1335#	1336	1337#		
	1338	1339#	1340	1341#	1342	1343#	1344	1345#	1346	1347#	1348	1349#	1350	1351#	1352		
	1353#	1354	1355	1356#	1357	1358#	1359#	1360	1361#	1362	1363#	1364	1365#	1366	1367#		
	1368	1369#	1370	1371#	1372	1373#	1374	1375#	1376	1377#	1378	1379#	1380	1381#	1382		
	1383#	1384	1385#	1386	1387#	1388	1389#	1390	1391#	1392	1393#	1394	1404#	1406#	1418#		
	2308#	2309	2311	2318#	2319	2324	2908#	2909	2910#	2911	2912#	2913#	2914#	2915	2916#		
	2917	2920#	2925#	2926#	2927#	2928	2929#	2930	2933#	2938#	2939#	2940#	2941#	2942	2943#		
	2944	2947#	2952#	2953#	2954#	2955#	2956	2957#	2958	2961#	2972#	2973#	2974#	2975#	2976		
	2977#	2978	2981#	2991#	2992#	2993#	2994#	2995#	2996	2997#	2998	3001#	3004#	3005#	3113#		
	3114	3229#	3230#	3231	3232#	3233	3237#	3238#	3239	3240#	3241	3277#	3278#	3279	3280#		
	3281	3300#	3301#	3302	3303#	3304	3310#	3311#	3312#	3313#	3314#	3315#	3316	3317#	3318		
	3325#	3326#	3327#	3328#	3329	3330#	3331	3339#	3340#	3341#	3342	3343#	3344	3346#	3347#		
	3348#	3349#	3350	3351#	3352	3369#	3370#	3371#	3372#	3373#	3374	3375#	3376	3385#	3386#		
	3387#	3388#	3389#	3390	3391#	3392	3400#	3401#	3402#	3403#	3404#	3405	3406#	3407	3417#		
	3418#	3419#	3420	3421#	3422	3426#	3427#	3428	3429#	3430	3446#	3447#	3448	3449#	3450		
	3466#	3467#	3468	3469#	3470	3504#	3505#	3506#	3507	3508#	3509	3514#	3515	3516#	3517#		
	3518	3519#	3520	3523#	3524#	3525#	3526	3527#	3528	3568#	3569#	3570	3571#	3572	3664#		
	3665#	3666#	3667#	3668#	3669#	3670	3671#	3672	3695#	3696#	3697#	3698#	3699#	3700#	3701		
	3702#	3703	3845#	3846#	3847	3848#	3849	3888#	3889#	3890	3891#	3892	3981#	4011#	4012#		
	4014#	4016#	4017#	4019#	4021#	4022#	4024#	4027#	4028#	4030#	4038#	4039#	4040#	4042#	4048#		
	4049#	4050#	4052#	4061#	4063#	4068#	4069#	4070#	4071#	4072	4073	4074	4075	4080#	4081#		
	4082	4083#	4084	4102#	4103#	4104#	4106#	4133#	4134#	4135#	4136#	4137#	4138	4143#	4144#		
	4145#	4146#	4147#	4148	4150#	4151#	4152#	4153#	4154#	4155	4158#	4159#	4161#	4162#	4169#		
	4185#	4198#	4199#	4201#	4204#	4205#	4212#	4225#	4226#	4233#	4247#	4248#	4255#	4285#	4286#		
	4287	4288#	4289	4327#	4328#	4329	4330#	4331	4339#	4341#	4345#	4346#	4355#	4356#	4357#		
	4358#	4359	4360	4361	4362	4372#	4373#	4374	4375#	4376	4381#	4382#	4383	4384#	4385		
	4409#	4410#	4411#	4412	4413#	4414	4424#	4425#	4426#	4427	4428#	4429	4450#	4451#	4452#		
	4453	4454#	4455	4466#	4467#	4468#	4469	4470#	4471	4563#	4564#	4565#	4566	4567#	4568		
	4632#	4633#	4634#	4635#	4636	4637#	4638	4679#	4680#	4681	4682#	4683	4709#	4710#	4711		
	4712#	4713	4772#	4773#	4774	4775#	4776	4821#	4822#	4823	4824#	4825	5154#	5155#	5156#		
	5157#	5172#	5173#	5174#	5175#	5185#	5186#	5187#	5188#	5228#	5229#	5230#	5231#	5279#	5280#		
	5281	5282#	5283	5312#	5313#	5314#	5315#	5316	5317	5318	5319	5370#	5371#	5372	5373#		
	5374	5393#	5394#	5395	5396#	5397	5448#	5459#	5460#	5461#	5462#	5496#	5497#	5498#	5499#		
	5502#	5537#	5560#	5561#	5562#	5563#	5701#	5713#	5714#	5715#	5716#	5834#	5835#	5836#	5837#		
	5859#	5860#	5861#	5862#	5880#	5881	5947#	5948	5998#	6038#	6045#	6059#	6060#	6061#	6062#		
	6091#	6108#	6115#	6116	6117	6125#	6126	6127	6128	6130#	6131	6132	6133	6135#	6136		

MSGNLS	6137	6140#	6175#	6176#	6177#										
MSGNSU	1#	1291#	4068#	4076	4355#	4363	5312#	5320							
MSGNTA	1#	1291#	1446#	2919#	2932#	2946#	2960#	2980#	3000#	3113#	3980#	4168#	4184#	4211#	4232#
MSGNTE	4254#	5880#	5947#	6090#	6140#	6141									
MSHAPT	1#	1291#	4270#												
MSHAP	1#	1291#	1310#												
MSHAP	1#	1291#	1310#	1349											
MSHAP	1#	1291#	1310#	1349											
MSINCR	1#	1291#	1293#	1418#	2906#	2916#	2920#	2923#	2929#	2933#	2936#	2943#	2947#	2950#	2957#
	2961#	2970#	2977#	2981#	2989#	2997#	3001#	3088#	3232#	3240#	3280#	3303#	3317#	3330#	3343#
	3351#	3375#	3391#	3406#	3421#	3429#	3449#	3469#	3508#	3519#	3527#	3571#	3671#	3702#	3848#
	3891#	3970#	3981#	3990#	4006#	4012#	4017#	4022#	4028#	4039#	4049#	4061#	4068#	4077	4083#
	4103#	4137#	4147#	4154#	4159#	4161#	4169#	4180#	4185#	4194#	4199#	4201#	4204#	4212#	4221#
	4233#	4243#	4255#	4270#	4271#	4288#	4330#	4339#	4345#	4355#	4364	4375#	4384#	4413#	4428#
	4454#	4470#	4567#	4637#	4682#	4712#	4775#	4824#	5154#	5172#	5185#	5228#	5282#	5312#	5321
	5373#	5396#	5448#	5459#	5496#	5502#	5537#	5560#	5701#	5713#	5781#	5834#	5859#	5918#	5998#
	6038#	6045#	6059#	6091#	6108#										
MSIOSE	1#	1291#													
MSLDRO	1#	1291#	4011#	4016#	4021#	4027#	4038#	4048#	4102#	4158#	4198#				
MSMASK	1#	1291#													
MSMCHI	1#	1291#													
MSMCLO	1#	1291#													
MSMSK1	1#	1291#													
MSPOP	1#	1291#	1446#	2919#	2932#	2946#	2960#	2980#	3000#	3113#	3980#	3997#	4168#	4184#	4211#
	4232#	4254#	5880#	5947#	6090#	6140#	6180#								
MSPRIN	1#	1291#	2908#	2925#	2938#	2952#	2972#	2991#	3229#	3237#	3277#	3300#	3310#	3325#	3339#
	3346#	3369#	3385#	3400#	3417#	3426#	3446#	3466#	3504#	3514#	3523#	3568#	3664#	3695#	3845#
	3888#	4080#	4285#	4327#	4372#	4381#	4409#	4424#	4450#	4466#	4563#	4632#	4679#	4709#	4772#
	4821#	5279#	5370#	5393#											
MSPUSH	1#	1291#	1293#	1418#	2906#	2923#	2936#	2950#	2970#	2989#	3088#	3970#	3990#	4006#	4180#
	4194#	4221#	4243#	4270#	4271	5781#	5918#	6108#							
MSPUT	1#	1291#	2908#	2925#	2938#	2952#	2972#	2991#	3229#	3237#	3277#	3300#	3310#	3325#	3339#
	3346#	3369#	3385#	3400#	3417#	3426#	3446#	3466#	3504#	3514#	3523#	3568#	3664#	3695#	3845#
	3888#	4080#	4133#	4143#	4150#	4285#	4327#	4372#	4381#	4409#	4424#	4450#	4466#	4563#	4632#
	4679#	4709#	4772#	4821#	5279#	5370#	5393#								
MSPUT1	1#	1291#	2908#	2910	2912	2913	2914	2925#	2926	2927	2938#	2939	2940	2941	2952#
	2953	2954	2955	2972#	2973	2974	2975	2991#	2992	2993	2994	2995	3229#	3230	3237#
	3238	3277#	3278	3300#	3301	3310#	3311	3312	3313	3314	3315	3325#	3326	3327	3328
	3339#	3340	3341	3346#	3347	3348	3349	3369#	3370	3371	3372	3373	3385#	3386	3387
	3388	3389	3400#	3401	3402	3403	3404	3417#	3418	3419	3426#	3427	3446#	3447	3466#
	3467	3504#	3505	3506	3514#	3516	3517	3523#	3524	3525	3568#	3569	3664#	3665	3666
	3667	3668	3669	3695#	3696	3697	3698	3699	3700	3845#	3846	3888#	3889	4080#	4081
	4133#	4134	4135	4136	4143#	4144	4145	4146	4150#	4151	4152	4153	4285#	4286	4327#
	4328	4372#	4373	4381#	4382	4409#	4410	4411	4424#	4425	4426	4450#	4451	4452	4466#
	4467	4468	4563#	4564	4565	4632#	4633	4634	4635	4679#	4680	4709#	4710	4772#	4773
	4821#	4822	5279#	5280	5370#	5371	5393#	5394							
MSRADI	1#	1291#	4071#	4358#	5315#	6115#	6125#	6130#	6135#						
MSRBRO	1#	1291#													
MSRNRO	1#	1291#	4038#	4040	4048#	4050	4102#	4104							
MSSETS	1#	1291#	1293#	1418#	2906#	2923#	2936#	2950#	2970#	2989#	3088#	3970#	3990#	4006#	4180#
	4194#	4221#	4243#	4271#	5781#	5918#	6108#								
MSSTAR	1#	1291#													
MSVC	1#	1291#	2908#	2916	2919#	2920	2925#	2929	2932#	2933	2938#	2943	2946#	2947	2952#
	2957	2960#	2961	2972#	2977	2980#	2981	2991#	2997	3000#	3001	3004#	3229#	3232	3237#
	3240	3277#	3280	3300#	3303	3310#	3317	3325#	3330	3339#	3343	3346#	3351	3369#	3375
	3385#	3391	3400#	3406	3417#	3421	3426#	3429	3446#	3449	3466#	3469	3504#	3508	3514#

CVCLHA DPV-11 DATA COMM. LINK TEST
CVCLHA.P11 25-JUL-80 10:26

MACY11 30A(1052) 25-JUL-80 10:29 PAGE 185
CROSS REFERENCE TABLE -- MACRO NAMES

N 14

SEQ 0182

. ABS. 034164 000

ERRORS DETECTED: 0

CVCLHA/I, CVCLHA.SEQ/CRF/SOL=SVC34R.MLB, CVCLHA.P11

RUN-TIME: 22 27 3 SECONDS

RUN-TIME RATIO: 95/53=1.8

CORE USED: 19K (37 PAGES)