

DPV-11

DPV-11 DCLT
CVCLHBO

AH-F584B-MC
FICHE 1 OF 1

OCT 1981
COPYRIGHT © 80-81
MADE IN USA



The main body of the document is a large, dense grid of data. Each cell in the grid contains a small, structured table or form. The text within these cells is extremely faint and difficult to read, but the overall layout suggests a comprehensive data set organized in a regular, repeating pattern. The grid covers most of the page area below the header.



CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 2

B 1

SEQ 0001

2222

.TITLE CVCLHB DPV-11 DATA COMM. LINK TEST

.REM 8

IDENTIFICATION

PRODUCT CODE: AC-F582B-MC
PRODUCT NAME: CVCLHB0 DPV-11 DATA COMM. LINK TEST
PRODUCT DATE: 26-OCT-81
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) 1980,1981 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
B	21-SEPT-81	ERNIE COOPER	ADD "SET EXPECT=TRANSMIT COMMAND ADD "EXIT" COMMAND ADD "RPT>" LEVEL ADD CHECK TO INSURE TRANSMIT LIST TOTAL = EXPECT LIST TOTAL UPDATE DOCUMENTATION

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 STATISTICAL COMMANDS
 - 2.5.3 RUN COMMANDS
 - 2.5.4 DEFAULTS
 - 2.5.5 PRINT COMMANDS
 - 2.5.6 MISC COMMANDS
 - 2.6 QUICK STARTUP PROCEDURE
- 3.0 ERROR INFORMATION
 - 3.1 TYPES OF ERROR MESSAGES
 - 3.2 SPECIFIC ERROR MESSAGES
 - 3.2.1 COMMAND LINE INTERPRETER ERRORS
 - 3.2.2 DCLT ERRORS
 - 3.2.3 DEVICE ERRORS
- 4.0 PERFORMANCE AND PROGRESS REPORTS
 - 4.1 PRINTING EVENT LOG
 - 4.2 OPERATOR STATUS MESSAGES
- 5.0 DEVICE INFORMATION TABLES

6.0 MODE AND MESSAGE DESCRIPTIONS

6.1 MODE DESCRIPTIONS

- 6.1.1 TRANSMIT MODE
- 6.1.2 RECEIVE MODE
- 6.1.3 PASSIVE MODE
- 6.1.4 ACTIVE MODE
- 6.1.5 DOWN-LINE LOAD MODE
- 6.1.6 TALK MODE
- 6.1.7 LISTEN MODE
- 6.1.8 MAINTENANCE MODE

6.2 MESSAGE DESCRIPTIONS

7.0 OTHER INFORMATION

- 7.1 INTERFACING TO AN "ITEP" NODE
- 7.2 TROUBLESHOOTING HINTS

- 7.2.1 INTERNAL LOOP AT EACH NODE
- 7.2.2 TRANSMIT ON ONE NODE-RECEIVE ON THE OTHER
- 7.2.3 ONE NODE ACTIVE-THE OTHER NODE PASSIVE
- 7.2.4 BOTH NODES ACTIVE
- 7.2.5 TALK AND LISTEN NODES FOR COMMUNICATIONS

7.3 EXAMPLE OF COMMANDS

- 7.3.1 MESSAGES COMMAND
- 7.3.2 STATISTICAL COMMANDS
- 7.3.3 RUN COMMANDS
- 7.3.4 PRINT COMMANDS
- 7.3.5 EXIT COMMAND

7.4 THINGS TO WATCH OUT FOR

1.0 GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

THIS DCLT (DATA COMMUNICATION LINK TEST) PROGRAM IS MEANT TO PROVIDE FIELD SERVICE WITH A TOOL TO MAINTAIN 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 LOCK BAD:

CVDPVXX DPV-11 FCTNL DIAG
CXDPVXX DPV-11 DECX MODULE
XX= LATEST REVISION

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.

THE DPV11 IS NOT A DMA DEVICE AND THUS MUST RELY ON THE SOFTWARE FOR SERVICE. THEREFORE THIS DCLT WILL NOT RUN WITH THE DPV AT ITS HIGH CLOCK SPEED OF 50KHZ.

2.0 OPERATING INSTRUCTIONS

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

2.1 COMMANDS

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

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

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

2.2 SWITCHES

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

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

EXAMPLE OF SWITCH USAGE:

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

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

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

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

2.3 FLAGS

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

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

IDR	APPLY TO DIAGNOSTICS WHICH DO NOT SUPPORT STATISTICAL REPORTING)
ADR	INHIBIT PROGRAM DROPPING OF UNITS
LOT	EXECUTE AUTODROP CODE
EVL	LOOP ON TEST
	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) N ?
```

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
SET EXPECTMSG-TYPE/QUAL	DEFINE A MESSAGE TO BE PUT ON THE EXPECTED LIST

WHERE: 'TYPE' IS:
-ONES
=ZEROS
1ALT
=OALT
=ITEP
-CCITT
=ALPHA
='A-Z,0-9,SPACES OR TABS IN QUOTES'

WHERE THE OPTIONAL 'QUAL' IS:

/SIZE=NNN MAKE THE MESSAGE 'NNN' BYTES
LONG. (DEFAULT VALUE IS
SIZE OF MESSAGE SPEC'D BY
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)
SET	EXPECT=TRANSMIT	MAKES A COPY OF THE TRANSMIT LIST IN THE EXPECT LIST.
SHOW	EXPECTLIST	LISTS THE MESSAGE SIZE AND TYPE FOR THE MESSAGES IN THE EXPECT LIST
SHOW	TRANSMITLIST	LISTS THE MESSAGE SIZE AND TYPE FOR THE MESSAGES IN THE TRANSMIT LIST

2.5.2 STATISTICAL COMMANDS

COMMAND

DESCRIPTION

PRINT

TAKES THE OPERATOR TO THE
REPORT LEVEL 'RPT>'. FROM
HERE YOU CAN EXAMINE 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.

WHERE '/B' IS OPTIONAL:
DEFAULT IS PRINT WORDS
'/B' CAUSES PRINT BYTES

IS PRINTED IN WORD FORMAT.

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

2.5.3 RUN 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 MODES 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:

=INTERNALITL LOOPS DATA INTERNAL TO USYNRT
=CABLE USE THIS FOR TESTING WITH H3260
OR H3259 TURNAROUND 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'

2.5.4 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
ZEROS - /SIZE=64/COPY=0
OAL - /SIZE=64/COPY=0
TALI - /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/NO*MODEM/CHECK/PASS=1

OTHER NOTES:

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

2.5.5 PRINT COMMAND

THE PRINT COMMAND TAKES YOU TO THE REPORT LEVEL 'RPT>'.
THE COMMANDS AVAILABLE IN RPT> ARE ...

COMMAND	DESCRIPTION
HELP OR ?	PRINTS HELP INFORMATION FOR RPT>
EXIT	RETURNS YOU TO THE LEVEL THAT YOU ENTERED FROM. (DCLT> OR DR>)
LOG	PRINTS THE DCLT EVENT LOG

NOTE: AT SOME FUTURE DATE WHEN DDCMP PROTOCOL IS
SUPPORTED BY THIS DCLT PROGRAM, A COMMAND
WILL BE ADDED TO EXAMINE ERROR COUNTERS.

2.5.6 MISC COMMANDS

COMMANDS	DESCRIPTION
EXIT	FROM THE DCLT> LEVEL RETURNS YOU TO DR>
HELP OR ?	PRINTS HELP INFORMATION

2.6 QUICK START-UP PROCEDURE (XXDP+)

TO START-UP THIS PROGRAM:

1. BOOT XXD +
2. GIVE THE DATE AND ANSWER THE LSI AND 50HZ (IF THERE IS A CLOCK) QUESTIONS
3. TYPE 'R NAME', WHERE NAME IS THE NAME OF THE BIN OR BIC FILE FOR THIS PROGRAM
4. TYPE 'START'
5. ANSWER THE 'CHANGE HW' QUESTION WITH 'Y'
6. ANSWER ALL THE HARDWARE QUESTIONS. THE NUMBER OF UNITS THAT CAN DCLT CAN USE IS ALWAYS '1'.

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

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

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

3.0 ERROR INFORMATION

3.1 TYPES OF ERROR MESSAGES

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

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

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

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

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

3.2 SPECIFIC ERROR MESSAGES

3.2.1 COMMAND LINE INTERPRETER ERRORS

ERROR MESSAGE:	MEANING
?ILL CMD-BAD SYNTAX?	A COMMAND WITH AN ILLEGAL CHAR WAS TYPED - RETYPE THE COMMAND. THE VALID COMMANDS AND THEIR SYNTAX ARE SHOWN IN SECTION 2.5.
?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.
- ? TRANSMIT AND EXPECT LIST MUST BE IDENTICAL FOR LOOP?
IF RUN COMMAND WITH '/LOOP/CH' IS TYPED THE TRANSMIT LIST AND EXPECT LIST MUST BE EQUAL. IF THEY ARE NOT THIS ERROR WILL BE DISPLAYED. USE 'SE E=T' COMMAND.

3.2.2 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.

3.2.3 DEVICE ERROR MESSAGE

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

RECVD=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.

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
XXXXXX 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
XXXXXX XXXXXXXX

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

MODEM DID NOT RETURN MODEM READY
RXCSR TXCSR
XXXXXX XXXXXXXX

MODEM DID NOT RESPOND WITH MR.

CRC IN ERROR
RDSR RXCSR
XXXXXX XXXXXXXX

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

RECEIVER OVERRUN
RDSR RXCSR
XXXXXX XXXXXXXX

RECIEVER DETECTED AN OVERRUN

TIMED OUT IN START,STACK ACK SEQ

THIS ERROR OCCURS WHEN IN THE

RDATA
XXXXXXX

SDATA
XXXXXXX

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
RECEIVED(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 DCL
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. THIS WILL TAKE YOU TO THE RPT> LEVEL. NOW
INPUT THE 'LOG' COMMAND. THE EVENTS ARE PRINTED IN A 'LAST-IN
FIRST-OUT' ORDER.

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

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

MESSAC	MEANING
TXQ	DEVICE IS ABOUT START TRANSMITTING A MESSAGE
TXC	TRANSMISSION OF MESSAGE COMPLETED
RXQ	DEVICE HAS QUEUED SPACE TO RECEIVE/ COMPLETED RECEIVE
ERR	DEVICE ERROR HAS OCCURRED
INI	DEVICE ABOUT TO BE INITIALIZED
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 (BIT 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; 2 BYTES 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, 1 BYTE ENQ(005), START STACK OR ACK, 3 BYTES OF 001, AND 2 BYTES OF CRC.

6.1.1 TRANSMIT MODE

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

6.1.2 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.

6.1.3 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.

6.1.4 ACTIVE MODE -----

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

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

6.1.5 DOWN-LINE-LOAD -----

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

6.1.6 TALK MODE -----

THE 'TALK' END OF THE LINK TRANSMITS OPERATOR-TYPED MESSAGES UNTIL A 'EXIT' MESSAGE IS TYPED. AT THAT POINT, THE NODE GOES INTO 'LISTEN' MODE. AN 'EXIT MESSAGE' IS A MESSAGE WHOSE FIRST FOUR CHARACTERS ARE 'EXIT'. SINCE ONLY THE FIRST FOUR CHARACTERS NEED TO BE 'EXIT', MORE CHARACTERS CAN BE ADDED SO THAT A MESSAGE MAY BE SENT AND THE MODE SWITCHED ALL AT ONCE. FOR EXAMPLE:

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

6.1.7 LISTEN MODE -----

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

6.1.8 MAINTENANCE 'LOOP' MODES

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

INTERNAL TTL LOOPS DATA 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-	DOWNLINE LOAD	** 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
ZEROES	MESSAGE OF ALL 0'S (00000000,00000000,00000000,...)
ONES	MESSAGE OF ALL 1'S (11111111,11111111,11111111,...)
1ALT	MESSAGE OF ALTERNATING 1'S (10101010,10101010,...)
0ALT	MESSAGE OF ALTERNATING 0'S (01010101,01010101,...)
CCITT	'CCITT' 512-BIT (VS. 511 BITS) TEST PATTERN
ITEP	'INTERPROCESSOR TEST PROGRAM'S (ITEP)' MESSAGE 1(DP1:) (<177><177>/SA THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.<15><12><001><177><177><177><177>)
ALPHA	ALPHA-NUMERICS (OR FUTURE COMM TURNAROUND MSG) (#\$!' (AMPERSAND) '()*+,-.0123456789:;<=>?@ABCDEFGHIJK LMNOPQRSTUVWXYZ/[\]^_`%)

OPERATOR-SPECIFIED

'A-Z,0-9,SPACES,TABS'
THESE ARE THAT THE CHARACTERS THAT CAN
BE TYPED BETWEEN QUOTATION MARKS ('..')
TO SPECIFIY A UNIQUE MESSAGE.

7.0 OTHER INFORMATION

7.1 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 F-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.

7.2 TROUBLESHOOTING HINTS

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

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

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

7.2.1 INTERNAL LOOP AT EACH NODE

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

A POSSIBLE COMMAND SEQUENCE IS:

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

WHAT THE ABOVE COMMAND SEQUENCE MEANS:

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

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

```
INI RXQ TXQ RXQ TXC TXQ RXQ TXC  
TXQ RXQ TXC EOP  
MODE-ACTIVE/LOOP=INTERNAL/PASS=00000  
/STATUS/NOCHECK/NOECHO/NO MODEM  
DCLT> (A) ?
```

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 PREREQUISITE DIAGNOSTICS FOR THE COMM. DEVICE.

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

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

WHAT THIS SEQUENCE MEANS:

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

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

```
INI RXQ TXQ TXC RXQ TXQ TXC RXQ
TXQ TXC CMP CMP CMP EOP RXQ TXQ
TXC RXQ TXQ TXC RXQ TXQ TXC CMP
CMP CMP EOP RXQ TXQ TXC RXQ TXQ
TXC RXQ TXQ TXC CMP CMP CMP EOP
MODE=ACTIVE/LOOP=INTERNAL/PASS=00000
/STATUS/CHECK/NOECHO/NOMODEM
```

IF A CABLE TURNAROUND CONNECTOR IS AVAILABLE, PUT IT ON THE END OF THE CABLE JUST BEFORE THE MODEM OR IF A H3260(RS-423) ON BOARD CONNECTOR IS AVAILABLE INSTALL IT AND RUN IN ACTIVE MODE WITH THE '/LOOP-CABLE' SWITCH.
POSSIBLE COMMAND SEQUENCE IS:

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

WHAT THIS SEQUENCE MEANS:

THIS SEQUENCE HAS THE '/LO=C'. THIS INFORMS THE SOFTWARE NOT TO CHECK FOR DATA SET READY SIGNAL FROM THE MODEM.

WHAT YOU SHOULD SEE AFTER ENTERING THE RUN COMMAND

IF THINGS ARE RUNNING CORRECTLY :

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

7.2.2 TRANSMIT ON ONE NODE RECEIVE ON THE OTHER

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

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

WHAT THIS SEQUENCE MEANS:

THE "C E " AND "C T" INITIALIZE BOTH THE TRANSMIT AND EXPECT LISTS. THE "SE T=1ALT/S=250" SETS THE TRANSMIT LIST ON NODE A TO BE 1 MESSAGE WITH A LENGTH OF 250 BYTES AND DATA OF ALTERNATING ONES AND ZEROS. THE "R M=TR/PAS=3" SETS THE RUN MODE OF NODE A TO BE TRANSMIT AND THE PASS COUNT IS SET TO 3. THE "R M=R/NOCH/PAS=3" SETS THE RUN MODE OF NODE B TO BE RECEIVE, NO DATA CHECKING IS TO BE DONE, AND THE PASS COUNT IS SET TO THREE.

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

FOR NODE A:

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

FOR NODE B:

```
INI RXQ EOP RXQ EOP RXQ EOP
MODE-RECEIVE/PASS=00000
/STATUS/NOCHECK/NOECHO/NOMODEM
DCLT> (A) ?
```

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

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

WHAT THIS SEQUENCE MEANS:

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

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

NODE A: IS THE SAME AS ABOVE.

NODE B:

INI RXQ CMP EOP RXQ CMP EOP RXQ CMP EOP
MODE RECEIVE/PASS-00000
/STATUS/CHECK/NOECHO/NOMODEM
DCLT> (A)?

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

7.2.3 ONE NODE ACTIVE THE OTHER NODE PASSIVE

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

NODE A	NODE B
-----	-----
C E	C E
C T	C T
SE T=CCITT/S=10/C=2	SE T=1ALT/S=20/C=2
R M=ACT/NOCH/PAS=3	R M=P/NOCH/PAS=3

WHAT THIS SEQUENCE MEANS:

THE EXECUTION OF THIS SEQUENCE CAUSES THE FOLLOWING THINGS TO HAPPEN ON NODE A. THE TRANSMIT AND EXPECT LISTS ARE INITIALIZED THEN THE TRANSMIT LIST IS SET TO 3 MESSAGES OF 10 BYTES EACH. THE DATA USED IN THE TRANSMIT MESSAGES IS THE CCITT PATTERN. THEN NODE A IS RUN IN ACTIVE MODE WITH DATA CHECKING DISABLED AND THE PASS COUNT SET TO THREE. NOTE STATUS WOULD STILL BE PRINTED IF THE PREVIOUS SEQUENCES HAD BEEN RUN. IF YOU ARE RUNNING FROM LOAD TIME YOU WOULD HAVE TO ADD A "/STA TO THE RUN COMMAND LINE.

NODE B: THE TRANSMIT AND EXPECT LISTS ARE INTIALIZED THEN THE TRANSMIT LIST IS SET TO 3 MESSAGES OF 20 BYTES EACH. THE DATA FOR EACH MESSAGE IS ALTERNATING 1'S AND 0'S. THE NODE IS THEN RUN IN PASSIVE MODE WITH DATA CHECKING DISABLED AND THE PASS COUNT SET TO 3.

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

FOR NODE A:

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

FOR NODE B:

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

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

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

7.2.4 BOTH NODES ACTIVE

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

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

WHAT THIS SEQUENCE MEANS:

NODE A SETS UP IS TRANSMIT LIST TO BE
3 MESSAGES. MESSAGE 1 IS 10 BYTES LONG AND
CONTAINS DATA OF ALTERNATING 0'S AND 1'S
MESSAGE 2 IS 20 BYTES LONG AND CONTAINS
DATA OF THE CCITT PATTERN. MESSAGE THREE
IS 30 BYTES LONG AND CONTAINS ALPHANUMERICS
FOR DATA. THE EXPECT LIST ALSO CONTAINS
3 MESSAGES. MESSAGE 1 IS 11 BYTES LONG AND
CONTAINS 0'S FOR DATA. MESSAGE TWO IS 21
BYTES LONG AND CONTAINS 1'S FOR DATA. MESSAGE
3 IS 31 BYTES LONG AND CONTAINS THE ITEP DATA.
NODE B HAS THE SAME MESSAGES EXCEPT THAT THE
TRANSMIT MESSAGE LIST IS THE EXPECT MESSAGE LIST
AND VICE VERSA.
BOTH NODES ARE RUN IN THE ACTIVE MODE WITH
DATA CHECKING AND -ASS COUNT EQUAL TO THREE.

WHAT YOU SHOULD SEE AFTER ENTERING THE RUN COMMAND
IF THINGS ARE RUNNING CORRECTLY :
ON BOTH NODES A AND B:

```
MODE-ACTIVE/PASS=00000  
/NOSTATUS/CHECK/NOECHO/NOMODEM  
DCLT> (A) ?
```

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

7.2.5 TALK AND LISTEN MODES FOR COMMUNICATING

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

```
R M LIS/NOST  
LIS>
```

```
R M TA/NOST  
TLK>
```

7.3 EXAMPLES OF COMMANDS

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

7.3.1 EXAMPLES OF MESSAGES COMMANDS

THE CLEAR COMMANDS .

```
C E  
C T
```

THIS WILL INITIALIZE THE TRANSMIT AND EXPECT LIST

TO 1 MESSAGE OF 58 BYTES. THE DATA OF THE MESSAGE WILL
BE THE ITEP MESSAGE.

IF THESE COMMANDS ARE FOLLOWED BY A SHOW COMMAND

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

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

DCLT> (A) ?

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

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

AND FOLLOWED IT WITH A SHOW EXPECT LIST COMMAND

SH E

WHAT YOU WOULD SEE IS

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

DCLT> (A) ?

7.3.2 EXAMPLES STATISTICAL COMMANDS

IF YOU TYPE A HELP COMMAND

HELP

WHAT YOU WILL SEE IS

DCLT CMDS:

CLEAR OR SHOW EXPECTLIST OR TRANSMITLIST
PRINT
EXIT
DUMP START-END/B
SET EXPECTMSG OR TRANSMITMSG=TYPE/SIZE=N OR /COPY-N
SET EXPECT=TRANSMIT
TYPE=ONES,ZEROES,1ALT,0ALT,ITEP,CC!TT,ALPHA
OR 'DPR SPCD=A-Z,SP,TAB,0-9' IN QUOTES'
RUN MODE=MTYP/LOOP-LTYP/CHECK,STATUS,ECHO,MODEM,PASS=N
MTYP TRAN,REC,ACT,PAS,TAL,LIS,DOWN
LTYP INT,CAB,LOC,REM/

DCLT> (A) ?

THE SAME WILL HAPPEN IF YOU USE THE ?

THE DUMP COMMAND WORKS LIKE THIS

DUM 41260-41300

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

41260 104423 000167 177772 021122 012112 006312 006312 006312
41300 006312

IF YOU HAD USED THE /B SWITCH

DUM 41260-41300/B

WHAT YOU WOULD SEE IS

```
41260 023 211 167 000 372 377 122 024
41270 112 024 312 014 312 014 312 014
41300 312
```

7.3.3 EXAMPLES RUN COMMANDS

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

IF YOU WERE TO EXECUTE THE RUN COMMAND

```
R M=TR/NOST/CH/PAS=4
```

WHAT WOULD HAPPEN IS AFTER 4 PASSES THE PROGRAM WOULD RETURN TO THE DCLT PROMPT AND PRINT

```
MODE=TRANSMIT/PASS=00000
/NOSTATUS/CHECK/NOECHO/NOMODEM
```

DCLT> (A) ?

IF YOU WERE TO EXECUTE THE RUN COMMAND

```
C E
C T
```

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

WHAT YOU WOULD SEE (IF USING DEFAULT TRANSMIT AND EXPECT MESSAGES) IS

```
INI RXQ TXQ TXC CMP EOP RXQ TXQ
TXC CMP FOP RXQ TXQ TXC CMP EOP
MODE-ACTIVE/LOOP-INTERNAL/PASS 0000
/STATUS/CHECK/NOECHO/NOMODEM
```

DCLT> (A) ?

IF YOU USE THE EXIT COMMAND

```
EXIT
```

WHAT YOU WOULD SEE IS

```
CZCLK EOP
0 CUMULATIVE ERRORS
```

DR>

7.3.4 EXAMPLES PRINT COMMANDS

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

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

HERE ARE SOME EXAMPLES OF RPT> LEVEL COMMANDS:

THE HELP OR ? COMMAND

```
HELP
```

OR

```
?
```

PRODUCES THE FOLLOWING:

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

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

BEWARE THAT THIS DCLT WILL NOT RUN THE DPV11 AT ITS HIGH CLOCK SPFED OF 50KHZ SINCE THE SOFTWARE IS NOT ABLE TO KEEP UP WITH THIS SPEED.

4016
4017
4018
4019
4020
4050
4051 002000
4052
4057
4061
4062
4082
4083
4084
4085
4086
4087
4088 002000
4089
4106
4107
4114
4115
4116 002000
(4) 002000
(4) 002000 103
(4) 002001 126
(4) 002002 103
(4) 002003 114
(4) 002004 110
(6) 002005 000
(6) 002006 000
(5) 002007 000
(5) 002010
(4) 002010 102
(5) 002011
(4) 002011 060
(5) 002012
(4) 002012 000000
(5) 002014
(4) 002014 003410
(5) 002016
(4) 002016 035346
(5) 002020
(4) 002020 000000
(5) 002022
(4) 002022 002130
(5) 002024
(4) 002024 000000
(5) 002026
(4) 002026 035624
(5) 002030
(4) 002030 000000
(5) 002032
(4) 002032 000000
(5) 002034

.SBTTL PROGRAM HEADER
BGNMOD

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

POINTER BGNRPT,BGNAU,BGNDU

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

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

(4) 002034 000000
(5) 002036
(4) 002036 000000
(5) 002040
(4) 002040 002124
(5) 002042
(4) 002042 000340
(5) 002044
(4) 002044 000000
(5) 002046
(4) 002046 000000
(5) 002050
(4) 002050 003
(3) 002051 003
(5) 002052
(4) 002052 000000
(5) 002054 000000
(5) 002056
(4) 002056 000000
(5) 002060
(4) 002060 011474
(5) 002062
(4) 002062 024156
(5) 002064
(4) 002064 000000
(5) 002066
(4) 002066 000000
(5) 002070
(4) 002070 025064
(5) 002072
(4) 002072 025056
(5) 002074
(4) 002074 000000
(5) 002076
(4) 002076 011504
(5) 002100
(4) 002100 104035
(5) 002102
(4) 002102 000000
(5) 002104
(4) 002104 024172
(5) 002106
(4) 002106 025034
(5) 002110
(4) 002110 025032
(5) 002112
(4) 002112 024164
(5) 002114
(4) 002114 000000
(5) 002116
(4) 002116 000000
(5) 002120
(4) 002120 000000

LSAPT:: .WORD 0
LSDTP:: .WORD 0
LSPRIO:: .WORD LSDISPATCH
LSENV1:: .WORD #PRI07
LSEXP1:: .WORD 0
LSMREV:: .WORD 0
 .BYTE C\$REVISION
 .BYTE C\$EDIT
LSEF:: .WORD 0
 .WORD 0
LSSPC:: .WORD 0
L\$DEVP:: .WORD 0
L\$REPP:: .WORD L\$DVTYP
L\$EXP4:: .WORD L\$RPT
L\$EXP5:: .WORD 0
L\$AUT:: .WORD 0
L\$DUT:: .WORD L\$AU
L\$LUN:: .WORD L\$DU
L\$DESP:: .WORD 0
L\$LOAD:: .WORD L\$DESC
 EMT E\$LOAD
L\$ETP:: .WORD 0
L\$ICP:: .WORD L\$INIT
L\$CCP:: .WORD L\$CLEAN
L\$ACP:: .WORD L\$AUTO
L\$PRT:: .WORD L\$PROT
L\$TEST:: .WORD 0
L\$DLY:: .WORD 0
L\$HIME:: .WORD 0

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

M 3
MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-16
DISPATCH TABLE

SEQ 0038

4129
4130
4131
4132
4133
4134
4135
4136 002122
(4) 002122 000001
(3) 002124
(6) 002124 025072
4137

.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

4145
4146
4147
4148
4149
4150
4151
4152
4153
4154 002126
(3) 002126 000010
(3) 002130
(3) 002130
4155
4165
4166
4167
4168
4169
4170
4171 002130 000001
4172
4173
4186
4187
4188
4189
4190
4191 002132 160170
4192 002134 000300
4193 002136 000240
4194 002140 000000
4195 002142 000000
4196 002144 000000
4197 002146 000000
4198
4199
4200 002150
(3) 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:


```
(1)
(1)
(1)
(1) 000340
(1) 000300
(1) 000240
(1) 000200
(1) 000140
(1) 000100
(1) 000040
(1) 000000
(1)
(1)
(1)
(1) 000004
(1) 000010
(1) 000020
(1) 000040
(1) 000100
(1) 000200
(1) 000400
(1) 001000
(1) 002000
(1) 004000
(1) 010000
(1) 020000
(1) 040000
(1) 100000
```

```

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

```
4281
4282
4283      001000      BUFLIM=512.      ;MAX BUFFER SIZE IN BYTES
4284
4285      000017      MSGLIM=15.      ; APPLIES TO TX,RX AND CMP BUFFS
4286
4287
4288
4289
4290
4291      ;***** INDEPENDENT EQUATES
4292
4293
4294
4295
4296
4297
4298
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318
4319
4320
4321
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346
4347
```

```

      000000      ;MCDE OF OPERATION EQUATES
      000001      REC=0      ;RECEIVE MODE
      000002      TRA=1      ;TRANSMIT MODE
      000003      PAS=2      ;PASSIVE MODE
      000004      ACT=3      ;ACTIVE MODE
      000005      DOW=4      ;DOWN-LINE-LOAD MODE
      000006      TAL=5      ;TALK MODE
      LIS=6      ;LISTEN MODE

      ;MAINT LOOP TYPE EQUATES
      000000      NONE= 0      ;NO LOOP
      000001      TTL= 1      ;INTERNAL TTL
      000002      CABLE= 2      ;CABLE LOOP
      000003      MODLOC= 3      ;MODEM LOCAL
      000004      MODREM= 4      ;MODEM REMOTE
      000005      MOP= 5      ;MOP

      ;CLOCK ENABLE VALUES TO BE LOADED IN CLK'S CSR
      000100      LCLKEN= 100      ;L-CLOCK CSR VALUE TO ENABLE THE CLOCK
      000111      PCLKEN= 111      ;P-CLOCK CSR VALUE TO ENABLE THE CLOCK
      001600      PCLKCT= 1600      ;P-CLOCK COUNT SET REGISTER FOR COUNTER

      ;PARAM WORD EQUATES
      000001      STATB= BIT0      ;OPERATOR AWAKE ASKED FOR
      000002      DATCKB= BIT1      ;DATA CHECK BIT
      000004      ECHOB= BIT2      ;ECHO BIT
      000010      MOCHK= BIT3      ;MODEM STATUS CHECK BIT
      000020      CRCB= BIT4      ;CRC CALCUALTE ASKED FOR
      000040      PROTOB= BITS      ;PROTOCOL PROCESSING ASKED FOR

      ;OPTION TYPE EQUATES
      000000      DPV= 0      ;CODE FOR DPV CHAR MODE

      ;EVENT LOG MESSAGE TYPES (USED TO LOCATE EVENT DESCRIPTION IN EVENT TABLE
      ; AND DISPATCHING TO SEPERATE SECTIONS OF THE EVENT REPORTING SECTION)
      000000      TXQ= 0      ;TRANSMIT MESSAGE QUEUED
      000002      TXC= 2      ;TRANSMIT COMPLETE
      000004      RXQ= 4      ;RECEIVE BUFFER QUEUED
      000006      RXC= 6      ;RECEIVE COMPLETE
      000010      DER= 10      ;DEVICE INFORMATION
      000012      DVI= 12      ;DEVICE ABOUT TO INIT
      000014      DCK= 14      ;DATA COMPARISON RESULTS
```

```
4356      000016      MSC=   16      ;MODEM STATUS CHANGE
4357
4358      000020      DLE=   20      ;DATA COMPARISON LENGH ERROR
4359      000022      DDE=   22      ;DATA COMPARISON DATA ERROR
4360      000024      EOP=   24      ;END OF PASS
4361
4362      ;EQUATES FOR FLAG WORD
4363
4364      000001      ININT=  BIT0      ;INPUT INT. REC.
4365      000002      OTINT=  BIT1      ;OUTPUT INT REC
4366      000004      QRX=   BIT2      ;RX QUED /COMPL
4367      000010      QTX=   BIT3      ;TX QUED/COMPL
4368      000100      ERX=   BIT6      ;EXPECT TO GET A RX COMPLETED
4369      000200      ETX=   BIT7      ;EXPECT TO GET A TX COMPLETED
4370
4381
4382      000020      TXM=   BIT4      ;INDICATES TO TX INTERRUPT ROUTINE
4383      ;THAT IT IS TIME TO TRANSMIT BODY OF MSG.
4384      000040      RXM=   BIT5      ;INDICATES TO RX INTERUPPT ROUTINE
4385      ;THAT IT IS TIME TO REC MSG BODY
4386      000400      BCC=   BIT8      ;TIME FOR CRC CHECK.
4387
4388      001000      PAD=   BIT9      ;INDICATES THAT PAD MUST BE SENT
4389
4390      002000      INOVR= BIT10     ;INIT OVER
4391
4392      004000      FIRST= BIT11     ;FIRST TIME FOR CTS
4393
4394      ; SPECIAL CLI CODES FOR 'CHAR' ARGUMENT IN CLI CALLS
4395      ; (COMMAND LINE INTERPRETER DEFINITIONS)
4396      000000      CLIERR= 0
4397      000001      CLIEXI= 1
4398      000002      CLIBR=  2
4399      000003      CLIBIF= 3
4400      000004      CLISPA= 4
4401      000005      CLINUM= 5
4402      000006      CLIALP= 6
4403      000007      CLIALN= 7
4404      000010      CLIOCT= 8.
4405      000011      CLIDEC= 9.
4406      000012      CLISTR= 10.
4407
4408      ; DEFS FOR COMMAND LINE INTERPRETATION ACTION VALUES
4409      000000      NULL=0
4410      000001      CLEAR=1
4411      000002      SHOW=2
4412      000003      CHECK=3
4413      000004      RUN=4
4414      000005      HLP=5
4415      000006      CSHEXP=6
4416      000007      CSHTRN=7
4417      000010      SETEXP=10
4418      000011      SETTRN=11
4419      000012      SIZE=12
4420      000013      QCOPY=13
4421      000014      NUM=14
```

4422 000015
4423 000016
4424 000017
4425 000020
4426 000021
4427 000022
4428 000023
4429 000024
4430 000025
4431 000026
4432 000027
4433 000030
4434 000031
4435 000032
4436 000033
4437 000034
4438 000035
4439 000036
4440 000037
4441 000040
4442 000041
4443 000042
4444 000043
4445 000044
4446 000045
4447 000046
4448 000047
4449 000050
4450 000051
4451 000052
4452 000053
4453 000054
4454 000055
4455 000056
4456 000057
4457 000060
4458
4459 000001
4460 000002
4461 000003
4473
4474
4475
4476
4477
4478
4479 020000
4480 001000
4481 010000
4482 000004
4483 040000
4484 000040
4485 000040
4486
4494
4495

OPRMSG=15
STATUS=16
ENDQ0=17
CMG0=20
CMG1=21
CMG2=22
CMG3=23
CMG4=24
CMG5=25
CMG6=26
ATVMOD=27
PASM0D=30
RECM0D=31
LISM0D=32
DLLMOD=33
TRAM0D=34
TALMOD=35
NO-36
ECHO=37
CRC=40
PROTO=41
PASC=42
MOP=43
TLL0P=44
CBL0P=45
LMD0P=46
RMD0P=47
NOTNUF=50
BADCHR=51
DMPS=52
DMPE=53
DMPQ=54
PRNT=55
MOSC=56
EXIT=57
SETET=60
RPHLP=1
RPEXT=2
RPLOG=3

:REV B BY EC
:REV B BY EC

:FOLLOWING EQUATES USED IN REPORT CLI ; REV B EC

:***** DEVICE DEPENDENT EQUATES
: MODEM SIGNAL BIT DEFINITIONS
: IF SIGNAL AVAILABLE IN DEVICE, EQUATE NAME TO BIT POSITION,
: ELSE EQUATE IT TO = 0

CTS= BIT13
DSR= BIT9
DCD= BIT12
RTS= BIT2
RI= BIT14
SQD= BITS
TM= BITS

:CLEAR TO SEND (CIRCUIT CB)
:DATA SET READY (CIRCUIT CC)
:DATA CARRIER DETECT (CIRCUIT CF)
:REQUEST TO SEND (CIRCUIT CA)
:RING INDICATOR (CIRCUIT CE)
:SIGNAL QUALITY DETECT (CIRCUIT CG)
:MODEM IN TEST MODE (RS 449 ONLY CIRCUIT TM)

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-23
GLOBAL EQUATES SECTION

SEQ 0045

4496
4497 000002
4498 000020
4499 000040
4500 000100
4501 000200
4502 002000
4503 004000
4504 000001
4505 000002
4506 000004
4507 000010
4508 000020
4509 000100
4510 000400
4511 001000
4512 100000
4513 100000
4514 000226
4515

: DEVICE SIGNALS

DTR= BIT1 ;DATA TERMINAL READY
RXENA= BIT4 ;RECEIVER ENABLE
DSITEN= BIT5 ;DATA SET CHANGE ENABLE
RINTEN= BIT6 ;REC INT. ENABLE
RDATRY= BIT7 ;REC DATA READY
RSTARY= BIT10 ;REC STATUS READY
RXACT= BIT11 ;REC ACTIVE
RESET= BIT0 ;MASTER RESET
TXACT= BIT1 ;TX ACTIVE
TBMT= BIT2 ;TX BUFFER EMPTY
TTLL= BIT3 ;TTL LOOP BIT
TXENA= BIT4 ;TX ENABLE
TINTEN= BIT6 ;TX INT ENABLE
TSOM= BIT8 ;TX START OF MSG.
TEOM= BIT9 ;TX END OF MSG.
TERR= BIT15 ;TX ERROR
RERR= BIT15 ;REC OVER RUN
SYN 226 ;SYNC WORD

4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527 002150
4528 002150 000001
4529 002152 000001
4530 002154 000001
4531 002156 000001
4532 002160 000100
4533 002162 000072
4534 002164 000101
4535 002166 000000
4536 002170 000001
4537
4538
4539
4540 002172
4541 002172 002214
4542 002174 002215
4543 002176 002216
4544 002200 002217
4545 002202 002220
4546 002204 002320
4547 002206 002412
4548 002210 002520
4549 002212 002642
4550
4551 002214 000
4552 002215
4553 002215 377
4554 002216
4555 002216 252
4556 002217
4557 002217 125
4558 002220
4559 002220
4560 002220 177603 157427 031011
002226 047321 163715 105221
002234 143325 142304
4561 002240 040041 014116 052606
002246 172334 105025 123754
002254 111337 111523
4562 002260 030030 145064 137642
002266 143531 063617 135075
002274 066730 026575
4563 002300 052012 053627 070071
002306 151172 165044 031605
002314 166632 016741
4564 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
OPBLIF ;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:

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

I 4
MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-25
DEFAULT MESSAGE DEFINITIONS AND TABLES

SEQ 0047

4565 002320
4566
4567 002320 077577 040444 052040
002326 042510 050440 044525
002334 045503 041040 047522
002342 047127 043040 054117
002350 045040 046525 042520
002356 020104 053117 051105
002364 052040 042510 046040
002372 055101 020131 047504
002400 027107
4568 002402 005015 077401 077577
002410 000177
4569 002412
4570 002412
4571 002412 022043 021041 023040
002420 024047 025051 026053
002426 027055 030460 031462
002434 032464 033466 034470
002442 035472 036474 037476
002450 040500 041502 042504
002456 043506 044510 045512
002464 046514 047516 050520
002472 051522 052524 053526
002500 054530 132
4572 002503 057 056133 057135
002510 022537 000
4573 002513
4574 002514
4575
4576
4577
4578
4579 002514 047045 040445
4580 002520 000122
4581 002642
4582
4583
4584
4585
4586 002642 033
4587 002643
4588 002644

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

.ASCII <15><12><001><177><177><177><177>

MSG5:
MSG6: ;ALPHA-NUMERICS (OR FUTURE COMM TURNAROUND MSG)
.ASCII /#S!' &'()*+,-.0123456789:;<->@ABCDEFGHIJKLMN0PQRSTUVWXYZ/

.ASCII ?/[\] ^ _ ` ?

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
MSG8:
MSG8: .EVEN

4590
4591
4592
4593
4594 002644 000
4595 002645 201
4596 002646 000000
4597 002650 001
4598 002651 001
4599 002652 001
4600 002653
4601 002654 002654
4602 002654 000006
4603
4604 002656 000
4605 002657 201
4606 002660 000000
4607 002662 001
4608 002663 001
4609 002664 001
4610 002666
4611

```
.....  
: 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 ; RESP NUM  
        .BYTE 1 ; MSG NUM  
        .BYTE 1 ; ADDR TO.  
        .EVEN
```

4613
 4614
 4615 002666 000122
 4616 003010 000000
 4617
 4618 003012 000000
 4619 003014 000000
 4620 003016 012246
 4621 003020 012261
 4622 003022 012376
 4623 003024 012463
 4624 003026 012510
 4625 003030 012567
 4626 003032 012645
 4627 003034 012735
 4628 003036
 4629
 4630 003036 013072
 4631 003040 013114
 4632 003042 013147
 4633 003044 013200
 4634 003046
 4635
 4636 003046 013266 013275 013302
 003054 013307 013314 013322
 003062 013327 013335
 4637
 4638
 4639
 4640
 4641 003066 000 377 252
 003071 125 203 177
 003074 043
 4642 003075
 4643 003076
 4644
 4645 003076 013346
 4646 003100 013356
 4647 003102 013367
 4648 003104 013377
 4649 003106 013406
 4650 003110 013423
 4651 003112 013430
 4652
 4653 003114 013437
 4654 003116 013447
 4655 003120 013460
 4656 003122 013466
 4657 003124 013501
 4658
 4659
 4660
 4661 003126 000000
 4662 003130 000000
 4663 003132 000000
 4664 003134 000000

;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 HLP3A
 .WORD HLP4
 .WORD HLP4A
 .WORD HLP5
 .WORD HLP6
 HLPEND:
 ;INDEX TABLE FOR REPORT 'RPT>' HELP MESSAGES REV B EC
 RHLFTB: .WORD RHLFP1
 .WORD RHLFP2
 .WORD RHLFP3
 .WORD RHLFP4
 RHLPEN:
 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

CVCLMB DPV-11 DATA COMM. LINK TEST
CVCLMB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-28
L 4
DEFAULT MESSAGE DEFINITIONS AND TABLES

SEQ 0050

4665 003136 000000
4666 003140 000000
4667 003142 000
4668 003143 000
4669

PSNUM: .WORD 0
PSRADX: .WORD 0
PSNNUF: .BYTE 0
PSGDBD: .BYTE 0

;LOC. TO HOLD NUMERIC VALUE FROM PARSE
;LOC. TO HOLD RADIX USED(LO) AND +/- (HI BYTE)
;RETURN =0 IF ENOUGH OF COMMAND FOUND
;RETURN CODE 0 IF NO ERROR FOUND

```

4671          .SBTTL          MESSAGE BUFFERS AND POINTER TABLES
4672
4673 003144 001000 TXBUF:  .BLKB  BUFLIM  :TRANSMITTER BUFFERS
4674 004144 001000 RXBUF:  .BLKB  BUFLIM  :RECEIVER BUFFERS
4675 005144 001000 CMPBUF: .BLKB  BUFLIM  :COMPARISON BUFFERS
4676 006144 000036 PTRTAB: .BLKW  MSGLIM*2 ;TABLE FOR MESSAGE ADDRS. & BYTE COUNTS
4677 006240 000036 PTR13:  .BLKW  MSGLIM*2
4678 006334 000036 PTR23:  .BLKW  MSGLIM*2
4679 006430          PTREND:          ; END OF MSG. PTR. TABLE
4680
4681 006430 000002          .BLKW  2          ;FILLER FOR OVERFLOW OF RX POINTER TABLE
4682
4683 006434 000000 RXPTR:  .WORD  0          ;RECEIVER MESSAGE POINTER
4684 006436 000000 TXPTR:  .WORD  0          ;TRANSMITTER BUFFER POINTER
4685 006440 000000 CMPPTR: .WORD  0          ;COMPARISON BUFFER POINTER
4686 006442 000000 CMPTOT: .WORD  0          ;CMP MSG TOTAL
4687 006444 000000 CTOTCC: .WORD  0          ;COMPARE BUFFER CHAR. COUNT
4688 006446 000000 CCURAD: .WORD  0          ;CURRENT ADDR OF CMP BUFF TO ADD AT
4689
4690 006450 000000 DVTXA:  .WORD  0          ;DEVICE TX ADDR
4691 006452 000000 DVTCC:  .WORD  0          ;DEVICE TX CHAR COUNT
4692 006454 000000 DVTCT:  .WORD  0          ;DEVICE TX MESSAGE COUNT
4693 006456 000000 TXMTOT: .WORD  0          ;TX MSG TOTAL
4694 006460 000000 TTOTCC: .WORD  0          ;TX BUFFER CHAR. COUNT
4695 006462 000000 TCURAD: .WORD  0          ;CURRENT ADDR. OF TX BUFF TO ADD AT
4696
4697 006464 000000 DVRXA:  .WORD  0          ;DEVICE RX ADDR
4698 006466 000000 DVRCC:  .WORD  0          ;DEVICE RX CHAR COUNT
4699 006470 000000 DVRCT:  .WORD  0          ;DEVICE RX MESSAGE COUNT
4700 006472 000000 RXMTOT: .WORD  0          ;RX MSG TOTAL
4701
4702 006474 000000 LNCNT:  .WORD  0          ;NUMBER OF OPERATOR AWAKE MSGS
4703 006476 000000 OPVAR:  .WORD  0          ;OPTIONAL VARIABLE LOCATION
4704 006500 000000 PSCNT:  .WORD  0          ;PASS COUNTER
4705 006502 000000 ERRCNT: .WORD  0          ;ERROR COUNTER
4706 006504 000000 STADD:  .WORD  0          ;START ADDR.
4707 006506 000000 ENADD:  .WORD  0          ;END ADDR. FOR DUMP
4708 006510 000000 BYTBIT: .WORD  0          ;BYTE BIT FOR DUMP ROUTINE
4709
4710          ;OTHER MESSAGE RELATED STORAGE LOCATIONS
4711
4712 006512 000000 MSGTYP: .WORD  0          ;TYPE OF DATA 0=0'S,1=1'S,2=10'S,3=01'S
4713          ;4=CCITT,5=QUICK FOX,6=ALPHA/NUM,7=OPER
4714 006514 000000 CURCC:  .WORD  0          ;TX/RX/CMP CHAR COUNT
4715 006516 000000 CPTRR:  .WORD  0          ;CURRENT RX POINTER
4716 006520 000000 CPTR:   .WORD  0          ;CURRENT POINTER
4717 006522 000000 CURADD: .WORD  0          ;CURRENT TX/RX/CMP START ADDD
4718 006524 000000 TOTCC:  .WORD  0          ;TOTAL CHAR COUNT NOT MORE THEN 'BUFLIM'
4719 006526 000000 OFFSET: .WORD  0          ;OFFSET COUNT
4720 006530 000000 TEMP:   .WORD  0          ;TEMPORARY LOCATIONS (USED A LOT)
4721 006532 000000 TEMP1:  .WORD  0
4722 006534 000000 TEMP2:  .WORD  0
4723 006536 000000 TEMP3:  .WORD  0
4724 006540 000000 TEMP4:  .WORD  0
4725 006542 000000 TEMP5:  .WORD  0
4726 006544 000000 CONOTM: .WORD  0          ;CONTROL OUT ERROR MSG. ADDRESS

```

CVCLMB DPV-11 DATA COMM. LINK TEST
CVCLMB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-30
MESSAGE BUFFERS AND POINTER TABLES

SEQ 0052

4727 006546 000000
4728 006550 000
4729 006551 000
4730

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

4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780

006552 000000
006554 0000C0
006556 000000
006560 000000
006562 000000

006564 000000

006566 000000
006570 000002

006572 000000
006574 C00000

006576 030704
006600 030736
006602 030776
006604 031032
006606 032216
006610 032242
006612 032462

006614 000000
006616 000000
006620 000000
006622 000074
006624 000000

006626 000000
006630 000000
006632 000000

006634 000000
006636 000000
006640 000000

;MORE INDEPENDENT CODE STORAGE LOCATIONS

LOGUNT: .WORD 0 ;LOC. TO HOLD LOGICAL UNIT NUMBER
PCADD: .WORD 0 ;LOC. HOLD PC OF CALLING ROUTINE
DCLFLG: .WORD 0 ;CLEANUP & EXIT FLAG -1 = EXIT TEST
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)
FHDPLX: .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
;TX ONLY DISPATCH
;PASSIVE LOOP BACK DISP
;ACTIVE LOOP BACK DISP
;DOWN LINE LOAD DISP
;TALK MODE DISPATCH
;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)

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-32
C 5
CLOCK TABLES, EVENT LOG AND POINTERS

SEQ 0054

4782
4783 006642 006644
4784 006644 000341
4785 007546 000001
4786
4787
4788
4789 007550 000000
4790
4791

;EVENT LOG TABLE AND ITS NEXT ENTRY POINTER
EVTPT:R: .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 QUEUE)

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

4793
4794
4795
4796 007552 020000
4797 007554 001000
4798 007556 010000
4799 007560 000004
4800 007562 040000
4801 007564 000040
4802 007566 000040
4803 007570
4804
4805
4806
4807 007570 016171
4808 007572 016175
4809 007574 016201
4810 007576 016205
4811 007600 016211
4812 007602 016215
4813 007604 016221
4814
4815
4816
4817
4818 007606 014604
4819 007610 014630
4820 007612 014657
4821 007614 014704
4822 007616 014732
4823 007620 014777
4824 007622 014747
4825 007624 015131
4826 007626 015025
4827 007630 015062
4828 007632 015115
4829
4830
4831
4832 007634 000000
4833 007636 000000
4834 007640 000000
4835 007642 000000
4836 007644 000000
4837 007646 000000
4838
4839
4840
4841 007650 021150
4842 007652 021150
4843 007654 021150
4844 007656 021150
4845 007660 021222
4846 007662 021316
4847 007664 021512
4848 007666 021566

;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 EVMSDR ;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 ;
EVTTMP: .WORD 0 ; " " BYTE COUNT " " " "

;REPORT CODING DISPATCH TABLE

RPTDSP: .WORD RPTTXQ ;TRANSMIT QUEUED ENTRY DECODING
.WORD RPTTXQ ;TRANSMIT COMPLETE ENTRY DECODING
.WORD RPTTXQ ;RECEIVER QUEUED ENTRY DECODING
.WORD RPTTXQ ;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

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-34
MODEM DATA SECTION

SEQ 0056

4849 007670 021512
4850 007672 021436
4851 007674 021362
4852
4853
4854 007676 000000
4855 007700 000000
4856 007702 000000
4857 007704 000000
4858

.WORD RPTDLE ;DATA COMPARISON LENGH ERROR
.WORD RPTDDE ;DATA COMPARISON DATA ERROR
.WORD RPTTEOP ;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

4860
4861
4862
4863
4864
4865
4866
4867
4868
4869
4870
4871
4872
4873
4874
4875 007706
4876
4877
4878 007706
4879 007712
4880 007716
4881 007720
4882 007734
4883 007736
4884 007752
4885 007754
4886 007770
4887 007772
4888 010004
4889 010010
4890 010024
4891 010030
4892 010044
4893 010050
4894 010054
4895 010066
4896 010072
4897 010104
4898 010110
4899
4900
4901
4902 010112
4903 010116
4904 010132
4905 010136
4906 010154
4907 010160
4908 010176
4909 010202
4910 010220
4911 010224
4912 010242
4913 010246
4914 010272
4915 010276

.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  
-----  
: ! ASCII 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 '?'  
N42$: CLI CLIEXI,0 ; IF YES DO 'HLP' AND EXIT  
CLI CLISTR,HLP,N43$,<'HELP'> ;ELSE, IS FIRST WORD A 'HELP'  
N43$: CLI CLIEXI,0 ; IF YES DO 'HLP' AND EXIT  
CLI CLISTR,PRNT,N44$,<'PRINT'> ;ELSE, IS FIRST WORD A 'PRINT'  
N44$: CLI CLIEXI,0 ; IF YES DO 'PRINT' AND EXIT  
CLI CLISTR,EXIT,N45$,<'EXIT'> ;ELSE, IS FIRST WORD 'EXIT' ;REV B BY EC  
N45$: CLI CLISTR,RUN,N46$,<'RUN'> ; IF YES DO 'EXIT' AND EXIT  
CLI CLIBR,0,N80$ ;ELSE, IS FIRST WORD A 'RUN'  
N46$: CLI CLISTR,NOTNUF,N40$,<'DUMP'> ; IF YES DO 'RUN' & GOTO N80$  
CLI CLIBR,0,N50$ ;ELSE, IS FIRST WORD A 'DUMP'  
N40$: CLI CLISTR,CLEAR,N20$,<'CLEAR'> ; IF YES GOTO N80$  
CLI CLIBR,NOTNUF,N100$ ;ELSE, IS FIRST WORD A 'CLEAR'  
N20$: CLI <'S'>,NOTNUF,N30$ ; IF YES DO 'CLR' & GOTO N100$  
CLI CLISTR,SHOW,N25$,<'HOW'> ;ELSE, IS FIRST CHAR. A 'S'  
N25$: CLI CLIBR,0,N100$ ; IF YES IS REST OF WORD 'HOW'  
CLI CLISTR,0,N30$,<'ET'> ; IF YES, DO 'SHOW',BR N100$  
N30$: CLI CLIBR,0,N110$ ; ELSE, IS REST OF WORD 'ET'  
CLI CLIERR,0 ; IF YES, DO 'SET', BR N110$  
;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  
N82$: CLI CLISTR,ATVMOD,N82$,<'ACTIVE'> ;IS NEXT WORD 'ACTIVE'  
CLI CLIBR,0,N115$ ; IF YES, DO 'ACTIVE',BR N115$  
N83$: CLI CLISTR,PASMOD,N83$,<'PASSIVE'> ;IS NEXT WORD 'PASSIVE'  
CLI CLIBR,0,N115$ ; IF YES, DO 'PASSIVE',BR N115$  
N84$: CLI CLISTR,RECMOD,N84$,<'RECEIVE'> ;IS NEXT WORD 'RECEIVE'  
CLI CLIBR,0,N115$ ; IF YES, DO 'RECVE',BR N115$  
N85$: CLI CLISTR,LISMOD,N85$,<'LISTEN'> ;IS NEXT WORD 'LISTEN'  
CLI CLIBR,0,N115$ ; IF YES, DO 'LISTEN',BR N115$  
N86$: CLI CLISTR,DLLOD,N86$,<'DOWNLINELOAD'> ;IS NEXT WORD 'DOW...'  
CLI CLIBR,0,N115$ ; IF YES, DO 'DWLL',BR N115$  
CLI <'T'>,0,N30$ ;IS NEXT CHAR A 'T'
```

```
4916 010302          CLI      CLISTR,TRAMOD,N87$,<'RANSMIT'>  ; IS REST OF WORD 'RANSMIT'
4917 010320          CLI      CLIBR,0,N115$                    ; IF YES, DO 'TRANSM',BR N115$
4918 010324          N87$:  CLI      CLISTR,TALMOD,N30$,<'ALK'>      ; IS REST OF WORD 'ALK'
4919 010336          CLI      CLIBR,0,N115$                    ; IF YES, DO 'TALK',BR N115$
4920                                     ; IF NO, ERROR - EXIT
4921
4922          ;SECOND KEYWORD (FOR CLEAR OR SHOW)
4923 010342          N100$:  CLI      CLISPA,0,N30$                    ;SKIP LEADING SPACES, NONE=ERR
4924 010346          N102$:  CLI      CLISTR,CSHEXP,N104$,<'EXPECTBUFF'> ;IS NEXT WORD 'EXPE...'
4925 010370          CLI      CLIEXI,0                          ; IF YES, DO CLR-EXP,EXIT
4926 010372          N104$:  CLI      CLISTR,CSHTRN,N30$,<'TRANSMITBUFF'> ;IS NEXT WORD 'TRANS...'
4927 010416          CLI      CLIEXI,0                          ; IF YES, DO CLR-TRN,EXIT
4928                                     ; IF NO - ERROR - EXIT
4929
4930          ;SECOND KEYWORD (FOR SET)
4931
4932 010420          N110$:  CLI      CLISPA,0,N30$
4933 010424          N111$:  CLI      CLISTR,SETEXP,N112$,<'EXPECT'>
4934 010442          CLI      CLIBR,0,N120$
4935 010446          N112$:  CLI      CLISTR,SETTRN,N30$,<'TRANSMIT'>
4936 010466          CLI      CLIBR,0,N120$
4937
4938          ;GET ADDRESSES FOR DUMP COMMAND
4939 010472          N50$:   CLI      CLIALP,0,N51$
4940 010476          N51$:   CLI      CLISPA,0,N52$
4941 010502          N52$:   CLI      CLIOCT,DMP$ ,N30$
4942 010506          CLI      <'>,NOTNUF,N125$
4943 010512          CLI      CLIOCT,DMPE,N30$
4944 010516          CLI      <'>,NOTNUF,N125$
4945 010522          CLI      <'B>,DMPQ,N30$
4946 010526          CLI      CLIBR,0,N125$
4947
4948          ;QUALIFIERS FOR THE RUN COMMAND
4949 010532          N115$:  CLI      CLIALP,0,N114$
4950 010536          N114$:  CLI      <'>,NOTNUF,N125$
4951 010542          CLI      CLISTR,NO,N116$,<'NO'>
4952 010554          N116$:  CLI      <'C>,0,N117$
4953 010560          CLI      CLISTR,CHECK,N117$,<'HECK'>
4954 010574          CLI      CLIBR,0,N115$
4955
4956          N117$:  CLI      CLISTR,STATUS,N118$,<'STATUS'>
4957          CLI      CLIBR,0,N115$
4958          N118$:  CLI      CLISTR,ECHO,N130$,<'ECHO'>
4959          CLI      CLIBR,0,N115$
4960
4961          N130$:  CLI      CLISTR,0,N132$,<'PASS'>
4962          CLI      CLIBR,0,N150$
4963
4964          N132$:  CLI      CLISTR,MOSC,N131$,<'MODEM'>
4965          CLI      CLIBR,0,N115$
4966
4967          N131$:  CLI      CLISTR,0,N30$,<'LOOP'>
4968
4969
4970
```

```
4991 010716          CLI      CLIBR,0,N140$
4992
4993
4994 010722          ;GET MESSAGE TYPE FOR SET MESSAGE COMMANDS
4995          N120$: CLI      <'=>,0,N30$
4996
4997 010726          ; LOOK FOR DEFAULT MESSAGE NAME
4998 010742          N60$:  CLI      CLISTR,MSG1,N61$,<'ONES'>
4999 010746          CLI      CLIBR,0,N121$
5000 010764          N61$:  CLI      CLISTR,MSG0,N62$,<'ZEROES'>
5001 010770          CLI      CLIBR,0,N121$
5002 011004          N62$:  CLI      CLISTR,MSG2,N63$,<'1ALT'>
5003 011010          CLI      CLIBR,0,N121$
5004 011024          N63$:  CLI      CLISTR,MSG3,N64$,<'0ALT'>
5005 011030          CLI      CLIBR,0,N121$
5006 011044          N64$:  CLI      CLISTR,MSG5,N65$,<'ITEP'>
5007 011050          CLI      CLIBR,0,N121$
5008 011064          N65$:  CLI      CLISTR,MSG4,N66$,<'CCITT'>
5009 011070          CLI      CLIBR,0,N121$
5010 011104          N66$:  CLI      CLISTR,MSG6,N67$,<'ALPHA'>
5011 011110          CLI      CLIBR,0,N121$
5012 011130          N67$:  CLI      CLISTR,SETEI,N68$,<'TRANSMIT'> ;REV B BY FC
5013          CLI      CLIBR,0,N125$
5014 011134          ; LOOK FOR QUOTED MESSAGE
5015 011140          N68$:  CLI      <'>,OPRMSG,N30$
5016 011144          N70$:  CLI      <'>,ENDQO,N71$
5017 011150          CLI      CLIBR,0,N121$
5018 011154          N71$:  CLI      CLISPA,0,N72$
5019 011160          N72$:  CLI      CLIALN,0,N73$           ;ONLY A-Z,SP,TAB, OR 0-9 BETWEEN ''S
5020 011164          CLI      CLIBR,0,N70$
5021          N73$:  CLI      CLIERR,BADCHR           ;PRINT ERROR IF NONE LEGAL CHAR FOR ''S
5022
5023 011166          ;GET QUALIFIERS (SIZE OR COPY) FOR SET MESSAGE COMMANDS
5024 011172          N121$:  CLI      CLIALP,0,N123$
5025 011176          N123$:  CLI      <'>,NOINUF,N125$
5026 011212          CLI      CLISTR,SIZE,N122$,<'SIZE'>
5027 011216          CLI      CLIBR,0,N126$
5028 011232          N122$:  CLI      CLISTR,QCOPY,N30$,<'COPY'>
5029          CLI      CLIBR,0,N126$
5030
5031 011236          ;NUMER FOR SIZE OR COPY
5032 011242          N126$:  CLI      <'=>,0,N30$
5033 011246          CLI      CLIDEC,NUM,N30$
5034          CLI      CLIBR,0,N121$
5035
5036 011252          ;GET MAINTENANCE LOOP TYPE FOR RUN 'LOOP' QUALIFIER
5037          N140$:  CLI      <'=>,0,N30$
5046
5047 011256          N141$:  CLI      CLISTR,TTLLOP,N142$,<'INTERNAL TTL'>
5048 011300          CLI      CLIBR,0,N115$
5049 011304          N142$:  CLI      CLISTR,CBLLOP,N143$,<'CABLE'>
5050 011320          CLI      CLIBR,0,N115$
5051 011324          N143$:  CLI      CLISTR,LMDLOP,N144$,<'LOCALMODEM'>
5052 011346          CLI      CLIBR,0,N115$
5053 011352          N144$:  CLI      CLISTR,RMDLOP,N30$,<'REMOTEMODEM'>
5054 011374          CLI      CLIBR,0,N115$
```

CVCLMB DPV-11 DATA COMM. LINK TEST
CVCLMB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-38
COMMAND LINE ACTION TREE

SEQ 0060

5055
5056
5057 011400
5058 011404
5059 011410
5060
5061
5062
5063
5064 011414
5065

:GET LINE NUMBER FOR 'PASS' RUN QUALIFIER
N150\$: CLI <'=>.0,N30\$
CLI CLIDEC,PASC,N30\$
CLI CLIBR,0,N115\$

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

```
5077
5078
5079 ;DEVICE DEPENDENT STORAGE LOCATIONS FOR
5080 ; CURRENT DEVICE PARAMTERS
5081
5082
5083 011416 000000 RXCSR: .WORD 0 ;REC CONTROL AND STATUS
5084 011420 000000 PCSAR: .WORD 0 ;STATUS REGISTIER
5085 011422 000000 RDSR: .WORD 0 ;REC DATA AND STATUS REG
5086 011424 000000 TXCSR: .WORD 0 ;TRANSMIT AND REC. CONTROL
5087 011426 000000 TDSR: .WORD 0 ;TRANSMIT DATA AND STATUS REG
5088
5089
5090 011430 000000 INVEC: .WORD 0 ;INPUT INTERRUPT VECTOR ADDRESS
5091 011432 000000 OUTVEC: .WORD 0 ;OUTPUT INTERRUPT VECTOR ADDRESS
5092 011434 000000 INTPRI: .WORD 0 ;INTERRUPT PRIORITY
5093 011436 000000 OPTYP: .WORD 0 ;DEVICE OPTION TYPE(0=DMC,5=DMR=DMC MODE
5094
5095 011440 065626 DPVP1: .WORD 065626 ;THIS WORD IS BROKEN DOWN AS FOLLOWS
5096 ;BITS 0-7 =SYNC WORD
5097 ;BITS 8-10=ERR DET SELECTED
5098 ;THIS IS SET TO SYNC 262 ;BIT11 = IDLE
5099 ;CRC 16 INIT TO 1 ;BIT12 = SEC ADDR. MODE
5100 ;STRIP SYNC AND BCP MODE ;BIT13 = STRIP SYNC
5101 ;IDLE SET TO MARK ;BIT14 = PORTO TYPE SEL(1=BCP 0=BOP)
5102 ;BIT15 = ALL PARTIES ADDRESS..
5103
5104
5105 011442 000000 CMODS: .WORD 0 ;CURRENT MODEM
5106 011444 000000 IRXCSR: .WORD 0 ;IMAGE OF RXCSR
5107 011446 000000 IRDSR: .WORD 0 ;IMAGE OR RDSR
5108 011450 000000 MSGPTR: .WORD 0 ;MSG PTR.FOR HEADER OR CONTROL
5109 011452 000000 MSGCC: .WORD 0 ;MSG COUNTER OR CC
5110 011454 000000 SYNCC: .WORD 0 ;SYNC CHAR COUNT.
5111 011456 000000 SYNCW: .WORD 0 ;SYNC WORD.PLUS TSOM BIT.
5112 011460 000000 RMSGPT: .WORD 0 ;MSG PTR FOR REC
5113 011462 000000 RMSCCC: .WORD 0 ;CHAR COUNTER FOR REC
5114 011464 000000 BCCW: .WORD 0 ;CRC HOLDING LOC.
5115 011466 000000 MGLCNT: .WORD 0 ;COUNT OF GLITCH ERRORS
5116 011470 000000 MHRCNT: .WORD 0 ;COUNT OF HARD ERRORS
5117 011472 000000 RNODE: .WORD 0 ;1=REMOTE NODE ITEP,0=NON ITEP
5130
5140
5141 ; ERR TBL
```

5143
5144
5145
5146
5147
5148
5149
5150
5151
5152
5153
5154
5155
5164
5165
(4)
(3)
(3)
(2)
5166
5172
5173
5174
5175
5176
(4)
(3)
(3)
(3)
(3)
(3)
(2)
5177
5178
5185
5186
5187

.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>

LSDV TYP::
.ASCIZ /DPV-11/
.EVEN

.SBTTL PROGRAM IDENTIFICATION
: TEST DESCRIPTION
:

DESCRIPT <DPV-11 DATA COMM LINK TEST >

LSDESC::
.ASCIZ /DPV-11 DATA COM
.EVEN

011474				
011474				
011474	050104	026526	030461	
011502	000			
011504				
011504				
011504	050104	026526	030461	
011512	042040	052101	020101	
011520	047503	046515	046040	
011526	047111	020113	042524	
011534	052123	000040		

.EVEN

```
.SBTTL GLOBAL FORMAT STATEMENTS, MESSAGES, AND ASCII INFO
5189
5190
5194 011540 04154 052114 000076 CLISPM: .ASCIZ /DCLT>/
5195 011546 050122 037124 000 CLISRP: .ASCIZ /RPT>/ ;REV B BY EC
5196 011553 045 022516 037501 CLIERM: .ASCIZ /%N%?ILL CMD-BAD SYNTAX?/
5197 011603 045 022516 037501 CLINUF: .ASCIZ /%N%?INCMPLTE CMD?/
5198 011626 047045 040445 047077 CLINBG: .ASCIZ /%N%?NUM TOO BIG?/
5199 011650 047045 040445 041077 CLIBRX: .ASCIZ /%N%?BAD RA'IX?/
5200 011670 047045 040445 021077 CLIBDL: .ASCIZ /%N%?'LOOP' VALID ONLY IN ACTIVE?/
5201 011732 047045 040445 021077 CLINPS: .ASCIZ /%N%?'ECHO' VALID ONLY IN PASSIVE?/
5202 011775 045 022516 037501 CLIBCR: .ASCIZ /%N%?ILL CHR- 'A-Z,0-9,SP,TAB' ONLY?/
5203 012042 047045 040445 021077 CLISEO: .ASCIZ /%N%?'SIZE=0' NOT VALID?/
5204 012073 045 022516 037501 CLIPW: .ASCIZ /%N%?TRANSMIT & EXPECT LIST MUST BE IDENTICAL FOR LOOP?;/REV B EC
5205 012163 045 022516 052101 HLP0: .ASCIZ /%N%?THIS IS DCLT. TYPE 'H' OR '?' FOR DETAILS/
5206 012241 045 022516 000124 HLPF: .ASCIZ /%N%?/
5207 012246 041504 052114 041440 HLP1: .ASCIZ /DCLT CMDS:/
5208 012261 040 046103 040505 HLP2: .ASCII / CLEAR OR SHOW EXPECTLIST OR TRANSMITLIST/<15><12>
5209 012335 040 051120 047111 .ASCII / PRINT/<15><12>
5210 012345 040 054105 052111 .ASCII / EXIT/<15><12> ;REV B EC
5211 012354 042040 046525 020120 .ASCIZ ? DUMP START-END/B?
5212 012376 051440 052105 042440 HLP3: .ASCIZ ? SET EXPECTMSG OR TRANSMITMSG=TYPE/SIZE=N OR /COPY=N?
5213 012463 040 042523 020124 HLP3A: .ASCIZ / SET EXPECT=TRANSMIT/ ;REV B EC
5214 012510 020040 052040 050131 HLP4: .ASCIZ ? TYPE=ONES,ZEROES,1ALT,0ALT,ITEP,CCITT,ALPHA?
5215 012567 040 020040 020040 HLP4A: .ASCIZ / OR 'OPR SPCD=A-Z,SP,TAB,0-9 IN QUOTES'/
5216 012645 040 052522 020116 HLP5: .ASCIZ ? RUN MODE=MTYP/LOOP=L TYP/CHECK,STATUS,ECHO,MODEM,PAS,N?
5217 012735 040 020040 052115 HLP6: .ASCII / MTYP=TRAN,REC,ACT,PAS,TAL,LIS,DOWN/<15><12>
5218 013004 020040 046040 054524 .ASCIZ / L TYP=INT,CAB,LOC,REM/
5219
5220 013034 047045 040445 054524 RHLP0: .ASCIZ /%N%?TYPE 'H' OR '?' FOR HELP!/ ;REV B EC
5221 013072 041504 052114 051040 RHLP1: .ASCIZ /DCLT REPORT CMDS:/ ;REV B EC
5222 013114 047514 020107 020055 RHLP2: .ASCIZ /LOG - PRINT DCLT EVENT LOG/ ;REV B EC
5223 013147 105 044530 020124 RHLP3: .ASCIZ /EXIT - EXIT REPORT LEVEL/ ;REV B EC
5224 013200 042510 050114 026440 RHLP4: .ASCIZ /HELP - PRINT THIS MESSAGE/ ;REV B EC
5225
5226 013232 047045 040445 051515 SHMSG: .ASCIZ ?%N%MSG: TYPE=%T%/SIZE=%D3?
5227 013266 042532 047522 051505 SH TYP0: .ASCIZ /ZEROES/
5228 013275 117 042516 000123 SH TYP1: .ASCIZ /ONES/
5229 013302 040461 052114 000 SH TYP2: .ASCIZ /1ALT/
5230 013307 060 046101 000124 SH TYP3: .ASCIZ /0ALT/
5231 013314 041503 052111 000124 SH TYP4: .ASCIZ /CCITT/
5232 013322 052111 050105 000 SH TYP5: .ASCIZ /ITEP/
5233 013327 101 050114 040510 SH TYP6: .ASCIZ /ALPHA/
5234 013335 117 051120 051440 SH TYP7: .ASCIZ /OPR SPEC/
5235 013346 042522 042503 053111 MO0: .ASCIZ /RECEIVE/
5236 013356 051124 047101 046523 MO1: .ASCIZ /TRANSMIT/
5237 013367 120 051501 044523 MO2: .ASCIZ /PASSIVE/
5238 013377 101 052103 053111 MO3: .ASCIZ /ACTIVE/
5239 013406 047504 047127 044514 MO4: .ASCIZ /DOWNLINELOAD/
5240 013423 124 046101 000113 MO5: .ASCIZ /TALK/
5241 013430 044514 052123 047105 MO6: .ASCIZ /LISTEN/
5242 013437 000 LP0: .ASCIZ //
5243 013440 046057 047517 036520 LP00: .ASCIZ ?/LOOP=?
5244 013447 111 052116 051105 LP1: .ASCIZ ?INTERNAL?
5245 013460 040503 046102 000105 LP2: .ASCIZ ?CABLE?
5246 013466 047514 040503 046514 LP3: .ASCIZ ?LOCALMODEM?
5247 013501 122 046505 052117 LP4: .ASCIZ ?REMODEM?
```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

M 5
MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-42
GLOBAL FORMAT STATEMENTS, MESSAGES, AND ASCII INFO

SEQ 0064

5248 013515 116 117
5249 013517 123 040524 052524
5250 013526 047516
5251 013530 044103 041505 000113
5252 013536 047516
5253 013540 041505 047510 000
5254 013545 116 117
5255 013547 115 042117 046505

PNST: .ASCII /NO/
PST: .ASCIZ /STATUS/
PNCK: .ASCII /NO/
PCK: .ASCIZ /CHECK/
PNEC: .ASCII /NO/
PEC: .ASCIZ /ECHO/
PNMS: .ASCII /NO/
PMS: .ASCIZ /MODEM/

5268
5269 013555 045 022516 046101
5270 013566 046124 037113 000
5271 013573 124 044510 020123
5272 013632

LISP: .ASCIZ /%N%ALIS>/
OPRMM: .ASCIZ /TLK>/
L5060: .ASCIZ /THIS A 50. OR 60. HZ. LSI-11:/
.EVEN

5273
5274
5275
5276
5277
5278
5279
5293

:
: : FORMAT STATEMENTS USED IN PRINT CALLS
:

5294 013632 047045 040445 047504
5295
5296

DLLCM: .ASCIZ /%N%ADOWN LINE LOAD NOT SUPPORTED BY THIS DEVICE/

5297 013712 047045 040445 046103
5298 013736 047045 040445 040502
5299 014017 115 054101 020056
5300 014057 102 043125 042506
5301 014075 045 022516 022524
5302 014126 047045 040445 044103
5303 014211 045 022516 032523
5309
5310 014247 045 022516 032523
5311
5312 014307 045 032523 040445
5313 014352 047045 051445 022463
5314 014404 051445 022465 041501
5315 014461 115 042117 046505
5316 014533 045 032523 040445

BDCLK: .ASCIZ /%N%ACLOCK NOT FOUND/
NOCLK: .ASCIZ /%N%ABAD CLOCK - PROGRAM WILL HANG ON 'TIMEOUT'!!/
TABEX: .ASCIZ /MAX. CHAR. MSG COUNT EXCEEDED -/
BUFEX: .ASCIZ /BUFFER FULL -/
MSGTRN: .ASCIZ /%N%T%A MSG. NOT BUILT !!/
MSGTRU: .ASCIZ /%N%ACHAR. COUNT EXCEEDS BUFF LIMIT - MSG TRUNCATED/
SHFO: .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/

5317
5318
5319
5320

: EVENT DESCRIPTION MESSAGES

5321 014604 051124 047101 046523
5322 014630 051124 047101 046523
5323 014657 122 041505 044505
5324 014704 042522 042503 053111
5325 014732 042504 044526 042503
5326 014747 104 052101 020101
5327 014777 104 053105 041511
5328 015025 104 052101 020101
5329 015062 040504 040524 041440
5330 015115 105 042116 047440
5331 015131 115 042117 046505

EDTXQ: .ASCIZ /TRANSMIT MSG QUEUED/
EDTXC: .ASCIZ /TRANSMIT MSG COMPLETED/
EDRXQ: .ASCIZ /RECEIVE SPACE QUEUED/
EDRXC: .ASCIZ /RECEIVE MSG COMPLETED/
EDDER: .ASCIZ /DEVICE ERROR/
EDDCK: .ASCIZ /DATA COMPARISON STARTED/
EDDVI: .ASCIZ /DEVICE INIT AND SETUP/
EDDLE: .ASCIZ /DATA COMPARISON LENGTH ERROR/
EDDDE: .ASCIZ /DATA COMPARISON DATA ERROR/
EDEOP: .ASCIZ /END OF PASS/
EDMOS: .ASCIZ /MODEM STATUS CHANGE/

```

5332
5333 ;EVENT REPORTING MESSAGES
5334 015155 045 031523 047445 BASM3: .ASCIIZ /%S3%03/
5335 015164 051445 022463 033117 BASM2: .ASCIIZ /%S3%06/
5336 015173 045 022516 033117 BASM1: .ASCIIZ /%N%06/
5337 015201 045 022516 052101 NULEVT: .ASCIIZ /%N%ATHE DCLT EVENT LOG IS EMPTY/
5338 015241 045 022516 037101 EVTF0: .ASCIIZ /%N%A>>> DCLT EVENT LOG ENTRY <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<</
5339 015335 045 022516 032504 EVTF1: .ASCIIZ /%N%D5%A:%Z2%A:%Z2%A:%S3%T/
5340 015364 047045 051445 022463 EVTF2: .ASCIIZ /%N%S3%AADDR OF MSG=%06%S3%ABYTE COUNT=%D5/
5341 015436 047045 051445 022463 EVTF3: .ASCIIZ /%N%S3%T%N/
5342 015450 051445 022463 033117 EVTF3C: .ASCIIZ /%S3%06%S3%06/
5343 015465 045 031523 047445 EVTF3D: .ASCIIZ /%S3%06%S3%06%S3%T/
5344 015507 045 022516 031523 EVTF4: .ASCIIZ /%N%S3%AADDR OF MSG=%06%S3%ABYTE COUNT=%D5%AND. OF CMP ERRS=%D5/
5345 015611 045 022516 031523 EVTF4A: .ASCIIZ /%N%S3%AADDR OF MSG=%06%S3%ARX BYTES=%D5%COMPARE BYTES=%D5/
5346
5356 015707 045 022516 031523 EVTF4B: .ASCIIZ /%N%S3%APASS=%D5%ERRORS=%D5%ASTRT-TO=%D5/
5357
5358 015766 051445 022465 041101 EVTF5A: .ASCIIZ /%S5%ABYTE# IN MSG.=%D5%EXPTD=%03%S3%ARECVD=%03/
5359
5360 016051 045 022516 034523 EVMOCG: .ASCIIZ /%N%S9%ACHANGED TO:/
5361
5362 ; *****
5363 ;DO NOT SEPERATE THE NEXT LIST OF MESSAGES - MODEM SIGNAL HEADER AND REPORT
5364
5365 016074 047045 051445 022470 FVMOH: .ASCIIZ /%N%S8%AMODEM STATUS: CTS DSR DCD RTS RI SQD TM/
5366 016154 047045 051445 022471 EVMOST: .ASCIIZ /%N%S9%S9%S5%A/
5367 016171 130 040 040 EVMCTS: .BYTE 'X,40,40,40
5368 016175 130 040 040 EVMDSR: .BYTE 'X,40,40,40
5369 016201 130 040 040 EVMDCD: .BYTE 'X,40,40,40
5370 016205 130 040 040 EVMRTS: .BYTE 'X,40,40,40
5371 016211 130 040 040 EVMRI: .BYTE 'X,40,40,40
5372 016215 130 040 040 EVMSQD: .BYTE 'X,40,40,40
5373 016221 130 040 040 EVMTM: .BYTE 'X,40,40,40
5374 016225 000 .BYTE 0
5375 .EVEN
5376
5377 ;EXECUTION STATUS MESSAGES TO BE PRINTED TO KEEP OPERATOR AWAKE
5378 016226 047045 000 CR: .ASCIIZ /%N%/ ;CR FOR LINES IN A ROW
5379 016231 045 031523 040445 STXQ: .ASCIIZ /%S3%ATXQ/ ;ABOUT TO TRANSMIT
5380 016242 051445 022463 052101 STXC: .ASCIIZ /%S3%ATXC/ ;TX COMPLETED
5381 016253 045 031523 040445 SRXQ: .ASCIIZ /%S3%ARXQ/ ;ABOUT TO RECEIVE
5382 016264 051445 022463 042501 SDVE: .ASCIIZ /%S3%AERR/ ;DEVICE ERROR
5383 016275 045 031523 040445 SCM: .ASCIIZ /%S3%ACMP/ ;ABOUT TO DO DATA CHECKING OF RECVD VS. EXPTD
5384 016306 051445 022463 044501 SDVI: .ASCIIZ /%S3%AINI/ ;DEVICE ABOUT TO BE INITIALIZED
5385 016317 045 031523 040445 SCML: .ASCIIZ /%S3%ACML/ ;COMPARE LENGTH ERROR
5386 016330 051445 022463 041501 SCMD: .ASCIIZ /%S3%ACMD/ ;COMPARE DATA ERROR
5387 016341 045 031523 040445 SEOP: .ASCIIZ /%S3%AEOP/ ;END OF PASS
5388 .EVEN
5389
5396 016352 051445 022463 046501 SMSC: .ASCIIZ /%S3%AMSC/ ;MODEM STATUS CHANGE.
5397
5398 016363 115 042117 046505 GLMSG: .ASCIIZ /MODEM STATUS GLITCHED/
5399 016411 115 042117 046505 HRDMSG: .ASCIIZ /MODEM STATUS HARD ERROR/
5400

```

```

5402
5421
5422 016441 115 051501 042524 DVEM0: .ASCII /MASTER RESET DID NOT WORK/
5423 016472 005015 020040 051040 .ASCIIZ <15><12>/ RXCSR TXCSR /
5424 016520 047516 041440 042514 DVEM1: .ASCII /NO CLEAR TO SEND FROM MODEM /
5425 016554 005015 020040 051040 .ASCIIZ <15><12>/ RXCSR TXCSR /
5426 016602 044524 042515 047440 DVEM2: .ASCII /TIME OUT WAITING FOR RX OR TX TO COMPLETE/
5427 016653 015 020012 020040 .ASCIIZ <15><12>/ RXCSR TXCSR/
5428 016677 103 041522 044440 DVEM3: .ASCII /CRC IN ERROR/
5429 016713 015 020012 020040 .ASCIIZ <15><12>/ RDSR RXCSR/
5430 016737 122 041505 044505 DVEM4: .ASCII /RECEIVER OVERRUN/
5431 016757 015 020012 020040 .ASCIIZ <15><12>/ RDSR RXCSR/
5432 017003 124 046511 042105 DVEM5: .ASCII /TIMED OUT IN START,STACK,ACK SEQ/
5433 017043 015 020012 020040 .ASCIIZ <15><12>/ RDATA SDATA/
5434 017067 115 042117 046505 DVEM6: .ASCII /MODEM DID NOT RETURN MODEM READY/
5435 017127 015 020012 020040 .ASCIIZ <15><12>/ RXCSR TXCSR/
5436
5437
5438
5449
5450 .EVEN
5451
5462
5463
5471
5472
5473
5474

```

.SBTTL GLOBAL ERROR REPCRT 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.
:--

5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5502
5503 017154
(3) 017154
5504 017154
(10) 017154 005046
(10) 017156 153716 006551
(9) 017162 005046
(9) 017164 153716 006550
(8) 017170 013746 006526
(7) 017174 012746 015766
(6) 017200 012746 000004
(3) 017204 010600
(4) 017206 104414
(4) 017210 062706 000012
5505 017214
(3) 017214
(3) 017214 104423
5506
5507 017216
(3) 017216
5508 017216
(8) 017216 013746 006540
(7) 017222 012746 014307
(6) 017226 012746 000002
(3) 017232 010600
(4) 017234 104414
(4) 017236 062706 000006
5509 017242
(3) 017242
(3) 017242 104423
5510
5511 017244
(3) 017244
5512 017244
(9) 017244 013746 006536
(8) 017250 010446
(7) 017252 012746 014404
(6) 017256 012746 000003
(3) 017262 010600
(4) 017264 104414
(4) 017266 062706 000010
5513 017272
(3) 017272
(3) 017272 104423
5514

BGNMSG ERR1

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

ERR1::

;INDIVIDUAL DATA COMPARE ERROR

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

ENDMSG

L10001:

TRAP C\$MSG

BGNMSG ERR2

PRINTB #EFM2,TEMP4

ERR2::

;TOTAL DATA COMPARE FAILS ERROR

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

ENDMSG

L10002:

TRAP C\$MSG

BGNMSG ERR10

PRINTB #EFM11,R4,TEMP3

ERR10::

;LENGH COMPARISON ERROR

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

ENDMSG

L10003:

TRAP C\$MSG

5515 017274
(3) 017274
5516 017274
(9) 017274 013746 011466
(8) 017300 013746 011470
(7) 017304 012746 014533
(6) 017310 012746 000003
(3) 017314 010600
(4) 017316 104414
(4) 017320 062706 000010
5517 017324
(3) 017324
(3) 017324 104423
5518
5519
5529
5530
5531
5532
5533
5534 017326
(3) 017326
5535 017326
(9) 017326 013746 006540
(8) 017332 013746 006536
(7) 017336 012746 015450
(6) 017342 012746 000003
(3) 017346 010600
(4) 017350 104414
(4) 017352 062706 000010
5536 017356
(3) 017356
(3) 017356 104423
5537
5538
5539
5540
5541
5542
5543 017360
(3) 017360
5544 017360
(10) 017360 013746 006544
(9) 017364 013746 006540
(8) 017370 013746 006536
(7) 017374 012746 015465
(6) 017400 012746 000004
(3) 017404 010600
(4) 017406 104414
(4) 017410 062706 000012
5545 017414
(3) 017414
(3) 017414 104423
5546
5547 017416
(4) 017416 000167

BGNMSG ERR4
PRINTB #EFM13,MHRCNT,MGLCNT

ENDMSG

:PRINT THE 2 OCTAL #'S IN TEMP3/4
:
BGNMSG ERR13
PRINTB #EVTF3C,TEMP3,TEMP4

ENDMSG

:PRINT THE 2 OCTAL #'S IN TEMP3/4
: AND THE MMSG. WHOSE ADDR. IS IN CONOTM
:
BGNMSG ERR14
PRINTB #EVTF3D,TEMP3,TEMP4,CONOTM

ENDMSG

EXIT MSG

ERR4::
;MODEM STATUS CHANGE
MOV MGLCNT,-(SP)
MOV MHRCNT,-(SP)
MOV #EFM13,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

L10004:
TRAP C\$MSG

ERR13::
MOV TEMP4,-(SP)
MOV TEMP3,-(SP)
MOV #EVTF3C,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP

L10005:
TRAP C\$MSG

ERR14::
MOV CONOTM,-(SP)
MOV TEMP4,-(SP)
MOV TEMP3,-(SP)
MOV #EVTF3D,-(SP)
MOV #4,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #12,SP

L10006:
TRAP C\$MSG

.WORD JSJMP

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81^{E 6} 10:00 PAGE 12-47
GLOBAL ERROR REPORT SECTION

SEQ 0069

(3) 017420 177772
5548
5549

.WORD L10006-2-.

5551
 5552
 5553
 5554
 5555
 5556
 5557
 5634
 5635
 5636
 5637
 5638
 5639
 5640
 5641
 5642
 5643
 5644
 5645
 5646
 5647
 5648
 5649
 5650
 5651
 5652
 5653
 5654
 5655
 5656
 5657
 5658
 5659
 5660
 5661
 5662 017422
 5663 017422 012122
 5664 017424 012112
 5665 017426 006312
 5666 017430 006312
 5667 017432 006312
 5668 017434 006312
 5669 017436 006322
 5670 017440 012122
 5671 017442 012122
 5672 017444 000207
 5673

```
.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
```

5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700
5701
5702
5703
5704
5705
5706
5707 017446
(3) 017446
5708
5709 017446 005077 167142
5710 017452 005337 006632
5711 017456 001015
5712 017460 013737 006622 006632
5713 017466 005237 006630
5714 017472 022737 000074 006630
5715 017500 001004
5716 017502 005237 006626
5717 017506 005037 006630
5718
5719 017512 005737 006634 1\$:
5720 017516 001402 2\$:
5721 017520 005337 006634
5722 017524 005737 006636 2\$:
5723 017530 001402 3\$:
5724 017532 005337 006636
5725 017536 005737 006640
5726 017542 001406 4\$:
5727 017544 023737 006622 006632
5728 017552 001002
5729 017554 005337 006640

```

.SBTTL          CLOCK INTERRUPT SERVICE ROUTINE
**
FUNCTIONAL DESCRIPTION:
THIS IS THE CLOCK INTERRUPT SERVICE ROUTINE WHICH TAKES CARE OF
KEEPING THE "TIME-SINCE-START" AND COUNTING DOWN ANY OF THE
"EVENT" TIMERS. THE TIMERS ARE USED TO TIME COMPLETION OF DEVICE
REQUESTS. THE "TIME-SINCE-START" IS USED TO BE LOGGED WITH EACH ENTRY
INTO THE EVENT LOG.

IMPLICIT INPUTS:
TIMTCK: THE CURRENT NO. OF TICKS LEFT TO BE COUNTED UNTIL A SECOND
HAS BEEN COUNTED OFF
CLKHZ: THE NO. OF TICKS IN A SECOND, DETERMINED BY THE SYS. LINE FREQ.
TIMMIN & TIMSEC: CURRENT VALUE OF "TIME-SINCE-START"
IN MINUTES & SECONDS
TIMER 1,2, & S: CURRENT VALUES OF THE "EVENT TIMERS"

IMPLICIT OUTPUTS:
NEW VALUE OF EVENT TIMER "1" DECREMENTED BY 1 TICK IF IT WAS NON-ZERO
NEW VALUE OF EVENT TIMER "2" DECREMENTED BY 1 TICK IF IT WAS NON-ZERO
NEW VALUE OF EVENT TIMER "S" DECREMENTED BY 1 SECOND IF IT WAS NON-ZERO

FUNCTIONAL SIDE EFFECTS:
THE CLOCK IS DISABLED UPON ENTRY AND REENABLED WHEN LEAVING

CALLING SEQUENCE:
THIS ROUTINE IS CALLED WHEN THE CLOCK INTERRUPTS THRU "CLKVEC".
THE ADDRESS OF THIS ROUTINE WAS LOADED INTO THE CLOCK'S INTERRUPT
VECTOR WITH A SUPERVISOR "SETVEC" CALL.
--

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.)

```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-50
CLOCK INTERRUPT SERVICE ROUTINE

SEQ 0072

5730 017560 013777 006624 167026 4\$: MOV CLKEN,@CLKCSR ;REENABLE THE CLOCK TO INTERRUPT
5731 017566 ENDSRV
(3) 017566 L10007:
(2) 017566 000002 RTI

5733
5734
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
5780
5781
5782
5783
5784
5785
5786
5787
5788

017570
017570 012737 016231 006532
017576 012737 000000 006530
017604 000517
017606
017606 012737 016242 006532
017614 012737 000002 006530
017622 000510
017624
017:24 012737 016253 006532
017632 012737 000004 006530
017640 000501
017642
017642 012737 000006 006530
017650 000475
017652

```
.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:

```

```

5789 017652 012737 016264 006532      MOV      #SDVE,TEMP1      ;SET UP MSG. TO PRINT
5790 017660 012737 000010 006530      MOV      #DER,TEMP       ;SET UP EVENT TYPE
5791 017666 000503                BR       LOGS3           ;GO LOG EVENT AND TIME
5792
5793 017670                LOGDVI:
5794 017670 012737 016306 006532      MOV      #SDVI,TEMP1     ;SET UP MSG. TO PRINT
5795 017676 012737 000012 006530      MOV      #DVI,TEMP       ;SET UP EVENT TYPE
5796 017704 113737 006562 006534      MOV      MODTYP,TEMP2    ;
5797 017712 113737 006564 006535      MOV      MLTYP,TEMP2+1   ;
5798 017720 013737 006572 006536      MOV      RPASS,TEMP3     ;
5799 017726 013737 006570 006540      MOV      PARAM,TEMP4    ;SET UP EVNT ENTRIES
5800 017734 000460                BR       LOGS3           ;GO LOG EVENT AND TIME
5801
5802 017736                LOGCMP:
5803 017736 012737 016275 006532      MOV      #SCM,TEMP1     ;SET UP MSG. TO PRINT
5804 017744 012737 000014 006530      MOV      #DCK,TEMP       ;SET UP EVENT TYPE
5805 017752 000451                BR       LOGS3
5806 017754                LOGCML:
5807 017754 012737 016317 006532      MOV      #SCML,TEMP1    ;
5808 017762 012737 000020 006530      MOV      #DLE,TEMP       ;SET UP MSG. AND TYPE
5809 017770 000442                BR       LOGS3           ;GO LOG EVENT AND TIME
5810 017772                LOGCMD:
5811 017772 012737 016330 006532      MOV      #SCMD,TEMP1    ;
5812 020000 012737 000022 006530      MOV      #DDE,TEMP       ;
5813 020006 000433                BR       LOGS3           ;GO LOG MSG TYPE AND TIME
5814 020010                LOGEOP:
5815 020010 012737 016341 006532      MOV      #SEOP,TEMP1    ;
5816 020016 012737 000024 006530      MOV      #EOP,TEMP       ;
5817 020024 000424                BR       LOGS3           ;GO LOG MSG TYPE AND TIME
5818
5819
5833 020026                LOGMSC:
5834 020026 012737 016352 006532      MOV      #SMSC,TEMP1    ;
5835 020034 012737 000016 006530      MOV      #MSC,TEMP       ;
5836 020042 000415                BR       LOGS3
5837
5838
5839 020044 013746 006502 006502      LOGS1: MOV      ERRCNT, -(SP) ;SAVE CURRENT ERROR COUNT
5840 020050 004737 033502                JSR      PC,DVMODS      ; GO GET MODEM STATUS
5841 020054 012604                MOV      (SP)+,R4       ;GET SAVED ERRCNT VALUE
5842 020056 020437 006502                CMP      R4,ERRCNT     ;WHERE ANY ERRORS FOUND
5843 020062 001402                BEQ     1$              ; BR IF NONE
5844 020064 000137 020300                JMP     LOGEX           ; ELSE, LEAVE WITHOUT LOGGING ANYTHING
5845                                ; BUT THE DEVICE ERROR FROM 'DVMODS'
5846 020070 013737 007550 006540 1$: MOV      MODS,TEMP4     ;AND PUT IT IN TEMP4
5847
5848 020076                LOGS3:
5849 020076 022737 000006 006530      CMP      #RXC,TEMP      ;
5850 020104 001434                BEQ     LOGS5           ;IF RXC DONT PRINT
5851 020106 032737 000001 006570      BIT     #STATB,PARAM    ;
5852 020114 001430                BEQ     LOGS5           ;IF NO STATUS SELECTED
5853                                ;GO TO 5
5854
5855 020116 022737 000010 006474      CMP     #10,LNCNT      ;HAVE WE DONE 10?
5856 020124 001012                BNE     LOGS4           ;IF NOT GO TO 4
5857 020126 005037 006474                CLR     LNCNT          ;ESLE CLEAR IT

```

```

5858
5859 020132          PRINTF  #CR          ;ELSE PRINT CR          MOV    #CR,-(SP)
(7) 020132 012746 016226          MOV    #1,-(SP)
(6) 020136 012746 000001          MOV    SP,R0
(3) 020142 010600          TRAP   C$PNTF
(4) 020144 104417          ADD    #4,SP
(4) 020146 062706 000004
5860 020152          LOGS4:
5861 020152 005237 006474          INC    LNCNT          ;INC COUNTER OF # OF AWAKE MSGS
5862 020156          PRINTF  TEMP1          ;PRINT OPERATOR AWAKE MSG.
(7) 020156 013746 006532          MOV    TEMP1,-(SP)
(6) 020162 012746 000001          MOV    #1,-(SP)
(3) 020166 010600          MOV    SP,R0
(4) 020170 104417          TRAP   C$PNTF
(4) 020172 062706 000004          ADD    #4,SP
5863 020176 010346          LOGS5: MOV    R3,-(SP)          ;SAVE R3 ON THE STACK
5864 020200 013703 006642          MOV    EVTPT,R3
5865 020204 113723 006530          MOVB   TEMP,(R3)+          ;LOG EVENT
5866 020210 013737 006622 006530          MOV    CLKHZ,TEMP
5867 020216 163737 006632 006530          SUB    TIMTCK,TEMP
5868 020224 113723 006530          MOVB   TEMP,(R3)+          ;LOG TIME SINCE START
5869 020230 113723 006530          MOVB   TIMSEC,(R3)+
5870 020234 113723 006626          MOVB   TIMMIN,(R3)+          ;TICKS,SECS AND MINS.
5871 020240 013723 006534          MOV    TEMP2,(R3)+          ;LOG EVNT ENTRY 3
5872 020244 013723 006536          MOV    TEMP3,(R3)+          ;LOG EVNT ENTRY 4
5873 020250 013723 006540          MOV    TEMP4,(R3)+          ;LOG EVNT ENTRY 5
5874 020254 020327 007546          CMP    R3,#EVTEND
5875 020260 103404          BLO    LOGS2
5876          ;IF EVENT LOG FULL GO
5877 020262 012713 177777          MOV    #-1,(R3)          ;CONTINUE;ELSE GO TO 2
5878 020266 012703 006644          MOV    #EVTLOG,R3          ;LOG A TABLE END
5879 020272 010337 006642          LOGS2: MOV    R3,EVTPT          ;PUT R3 TO START OF TABLE
5880 020276 012603          MOV    (SP)+,R3          ;RESTORE POINTER
5881 020300 000207          LOGEX: RTS              ;RESTORE R3
5882
  
```

```

5884 .SBTTL REPORT EVENT LOG
5885 ;REV B BY EC
5886 ;THE FOLLOWING COMMANDS ADDED TO REVISION B CVCLHB
5887 ;:DPV DCLT PROGRAM
5888 ;:RPT> LOG
5889 ;: HELP
5890 ;: EXIT
5891 ;:AT SOME FUTURE DATE AFTER DDCMP PROTOCOL IS SUPPORTED BY THIS
5892 ;:PROGRAM, ERROR REPORTING CAN BE ADDED HERE.
5893
5894 REPORT: MOV R2,-(SP) ;SAVE R2,R3,R4 ON THE STACK
5895 MOV R3,-(SP)
5896 MOV R4,-(SP)
5897
5898 ;PRINT HELP MESSAGE
5899 PRINTF #RHLPO ;BASIC HELP MESSAGE
5900
5901 (7) 020310 012746 013034 MOV #RHLPO,-(SP)
5902 (6) 020314 012746 000001 MOV #1,-(SP)
5903 (3) 020320 010600 MOV SP,R0
5904 (4) 020322 104417 TRAP C$PNTF
5905 (4) 020324 062706 000004 ADD #4,SP
5906
5907 GETRCL: CLRB P$GDBD ;INIT GOOD/BAD FLAG -1-BAD INPUT
5908 CLRB P$NNUF ;INIT MORE COMMAND LINE INPUT NEEDED
5909
5910 ;PRINT PROMPT 'RPT>'
5911 GMANID CLISRP,CMDBUF,A,0,1,72..NO
5912
5913 (3) 020340 104443 TRAP C$GMAN
5914 (3) 020342 000406 BR 10000$
5915 (4) 020344 002666 .WORD CMDBUF
5916 (5) 020346 000142 .WORD T$CODE
5917 (5) 020350 011546 .WORD CLISRP
5918 (5) 020352 000000 .WORD 0
5919 (5) 020354 000001 .WORD T$LOLIM
5920 (5) 020356 000110 .WORD T$HILIM
5921 (3) 020360 10000$:
5922 5906 020360 012737 002666 003126 MOV #CMDBUF,P$BUFA ;INFUT BUFFER
5923 5907 020366 012737 020522 003130 MOV #CLIRT,P$TREE ;REPORT CLI TREE
5924 5908 020374 012737 020610 003132 MOV #CLIRAC,P$ACT ;ACTION ROUTINES
5925 5909 020402 005037 003012 CLR QUALFG
5926 5910 020406 004737 023066 JSR PC,P$TRV ;GO PARSE COMMAND LINE
5927 5911 020412 105737 003143 TSTB P$GDBD ;COMMAND OK ?
5928 5912 020416 001412 BEQ 1$ ;YES,BRANCH
5929 5913 020420 PRINTF #CLIERM ;PRINT INVALID INPUT MESSAGE
5930 (7) 020420 012746 011553 MOV #CLIERM,-(SP)
5931 (6) 020424 012746 000001 MOV #1,-(SP)
5932 (3) 020430 010600 MOV SP,R0
5933 (4) 020432 104417 TRAP C$PNTF
5934 (4) 020434 062706 000004 ADD #4,SP
5935 5914 020440 000137 020330 JMP GETRCL ;TRY AGAIN
5936 5915
5937 5916 020444 105737 003142 1$: TSTB P$NNUF ;MORE COMMAND NEEDED ?
5938 5917 020450 001412 BEQ 10$ ;NO,BRANCH
5939 5918 020452 PRINTF #CLINUF ;INCOMPLETE MESSAGE
5940 (7) 020452 012746 011603 MOV #CLINUF,-(SP)
5941 (6) 020456 012746 000001 MOV #1,-(SP)

```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

M 6
MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-55
REPORT EVENT LOG

SEQ 0077

(3)	020462	010600							MOV	SP,R0
(4)	020464	104417							TRAP	C\$PNTF
(4)	020466	062706	000004						ADD	#4,SP
5919	020472	000137	020330			JMP	GETRCL			
5920										
5921	020476	023727	003010	000002	10\$:	CMP	KEYWD1,#RPEXT			:EXIT COMMAND ?
5922	020504	001402				BEQ	20\$:YES,BRANCH
5923	020506	000137	020330			JMP	GETRCL			:GET ANOTHER COMMAND
5924	020512	012604			20\$:	MOV	(SP)+,R4			:RESTORE R4
5925	020514	012603				MOV	(SP)+,R3			:RESTORE R3
5926	020516	012602				MOV	(SP)+,R2			:RESTORE R2
5927	020520	000207				RTS	PC			:RETURN

5929
5930 020522
5931 020526
5932 020532
5933 020534
5934 020550
5935 020552
5936 020566
5937 020570
5938 020602
5939 020604
5940 020606

.SBTTL COMMAND LINE PARSING TREE FOR REPORT
CLIRT: CLI CLISPA,0,R10\$;SKIP SPACES IN COMMAND LINE
R10\$: CLI <'?'>,RPHLP,R11\$;IF INPUT = ? THEN PRINT HELP MESSAGE
;AND EXIT PARSER
R11\$: CLI CLIEXI,0 ;IF INPUT = 'HELP' THEN PRINT HELP
;MESSAGE AND EXIT PARSER
R12\$: CLI CLISTR,RPHLP,R12\$,<'HELP'> ;IF INPUT = 'EXIT' THEN SET KEYWORD =
;RPEXT AND EXIT PARSER
R13\$: CLI CLISTR,RPEXT,R13\$,<'EXIT'> ;IF INPUT = 'LOG' THEN GO PRINT EVENT
;LOG AND EXIT PARSER
R30\$: CLI CLIERR,0
R125\$: CLI CLIEXI,0

5942
5943 020610 006302
5944 020612 016202 020626
5945 020616 062702 020626
5946 020622 004712
5947 020624 000207
5948 020626 000010
5949 020630 000012
5950 020632 000056
5951 020634 000066
5952
5953
5954 020636 000207
5955
5956
5957 020640 012702 003036
5958 020644
(8) 020644 012246
(7) 020646 012746 012241
(6) 020652 012746 000002
(3) 020656 010600
(4) 020660 104417
(4) 020662 062706 000006
5959 020666 020227 003046
5960 020672 001364
5961 020674 012737 000001 003010
5962 020702 000207
5963
5964
5965 020704 012737 000002 003010
5966 020712 000207
5967
5968
5969 020714 004737 020730
5970 020720 012737 000003 003010
5971 020726 000207
5972
5973
5974

```

.SBTTL CLI ACTION DISPATCHER AND ROUTINES
CLIRAC: ASL R2 ;SET UP INDEX
MOV 10$(R2),R2 ;
ADD #10$,R2 ;
JSR PC,(R2) ;GO DO ACTION
RTS PC ;RETURN
10$: .WORD ACTRNL-10$ ;NULL
.WORD ACTRHL-10$ ;HELP ROUTINE
.WORD ACTREX-10$ ;EXIT ROUTINE
.WORD ACTRLG-10$ ;REPORT EVENT LOG ROUTINE

:::::ACTION ROUTINES FOR REPORT:::::
ACTRNL: RTS PC ;NULL

;PRINT HELP MESSAGE
ACTRHL: MOV #RHLPTB,R2 ;INDEX FOR HELP MESSAGES
1$: PRINTF #HLPF,(R2)+ ;PRINT IT

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

CMP R2,#RHLPEN ;LAST MESSAGE ?
BNE 1$ ;NO BRANCH
MOV #RPHLP,KEYWD1 ;SET KEYWORD
RTS PC ;RETURN

;EXIT REPORT LEVEL
ACTREX: MOV #RPEXT,KEYWD1 ;SET KEYWORD AND RETURN
RTS PC

;PRINT ERROR LOG
ACTRLG: JSR PC,REPLG ;GO PRINT EVENT LOG
MOV #RPLOG,KEYWD1 ;SET KEYWORD
RTS PC ;RETURN

```

```

5976
5977
5978
5979 020730 010246
5980 020732 010346
5981 020734 010446
5982
5990
5991
5992
5993 020736 013702 006642
5994 020742 023727 006644 177777
5995 020750 001034
5996 020752
(7) 020752 012746 015201
(6) 020756 012746 000001
(3) 020762 010600
(4) 020764 104416
(4) 020766 062706 000004
5997 020772 000137 021656
5998
5999 020776 162702 000012
6000
6001
6002 021002 020227 006644
6003 021006 001010
6004 021010 012702 007546
6005 021014 026227 177776 177777
6006 021022 001007
6007 021024 000137 021656
6008
6009 021030 020237 006642
6010 021034 001002
6011 021036 000137 021656
6012
6013 021042 162702 000012
6014 021046
(7) 021046 012746 015241
(6) 021052 012746 000001
(3) 021056 010600
(4) 021060 104416
(4) 021062 062706 000004
6015 021066 112203
6016 021070 112237 007640
6017 021074 112237 007634
6018 021100 112237 007636
6019 021104
(11) 021104 016346 007606
(10) 021110 013746 007640
(9) 021114 013746 007634
(8) 021120 013746 007636
(7) 021124 012746 015335
(6) 021130 012746 000005
(3) 021134 010600
(4) 021136 104416
(4) 021140 062706 000014

.SBTTL          DUMP EVENT LOG

REPLOG: 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      EVTICK,-(SP)
        MOV      EVTSEC,-(SP)
        MOV      EVTMIN,-(SP)
        MOV      #EVTF1,-(SP)
        MOV      #5,-(SP)
        MOV      SP,R0
        TRAP     C$PNTS
        ADD     #14,SP
  
```

```

6020 021144 000173 007650          JMP      @RPTDSP(R3)      ;DISPATCH TO DECODING SECTION FOR SPECIFIC TYPE
6021
6022 021150 012237 007642          RPTTXQ: MOV      (R2)+,EVTADD ;STORE MESSAGE ADDRESS FOR PRINTING
6023 021154 012237 007644          MOV      (R2)+,EVTBCT   ;STORE BYTE COUNT FOR PRINTING
6024 021160 012203                MOV      (R2)+,R3       ;STORE MODEM STATUS FOR PRINTING
6025 021162                PRINTS   #EVTF2,EVTADD,EVTBCT ;PRINT ADDR,BYTE CNT
(9) 021162 013746 007644                MOV      EVTBCT,-(SP)
(8) 021166 013746 007642                MOV      EVTADD,-(SP)
(7) 021172 012746 015364                MOV      #EVTF2,-(SP)
(6) 021176 012746 000003                MOV      #3,-(SP)
(3) 021202 010600                MOV      SP,R0
(4) 021204 104416                TRAP    C$PNTS
(4) 021206 062706 000010                ADD     #10,SP
6026 021212 004737 021666          JSR      PC,RPTMSB      ;GO PRINT MODEM STATUS
6027 021216 000137 020776          JMP      RPT            ;GO BACK FOR NEXT EVENT ENTRY
6028
6029 021222 012237 007646          RPTDER: MOV      (R2)+,EVTTMP ;GET ADDRESS OF DEVICE INFO MESSAGE
6030 021226 012237 007676          MOV      (R2)+,DEV1     ;STORE DEVICE REG CONTENTS FOR PRINTING
6031 021232 012237 007700          MOV      (R2)+,DEV2
6032 021236                PRINTS   #EVTF3,EVTTMP ;PRINT DEVICE REG CONTENTS.
(8) 021236 013746 007646                MOV      EVTTMP,-(SP)
(7) 021242 012746 015436                MOV      #EVTF3,-(SP)
(6) 021246 012746 000002                MOV      #2,-(SP)
(3) 021252 010600                MOV      SP,R0
(4) 021254 104416                TRAP    C$PNTS
(4) 021256 062706 000006                ADD     #6,SP
6033 021262                PRINTS   #EVTF3C,DEV1,DEV2
(9) 021262 013746 007700                MOV      DEV2,-(SP)
(8) 021266 013746 007676                MOV      DEV1,-(SP)
(7) 021272 012746 015450                MOV      #EVTF3C,-(SP)
(6) 021276 012746 000003                MOV      #3,-(SP)
(3) 021302 010600                MOV      SP,R0
(4) 021304 104416                TRAP    C$PNTS
(4) 021306 062706 000010                ADD     #10,SP
6034 021312 000137 020776          JMP      RPT            ;GO BACK FOR NEXT EVENT ENTRY
6035
6036 021316 005037 007676          RPTDVI: CLR      DEV1
6037 021322 005037 007700          CLR      DEV2          ;CLEAR UPPER BYTES OF DEV1 & DEV2 BEFORE USE
6038 021326 112237 007676          MOV      (R2)+,DEV1     ;STORE SETUP OPERATION PARAMETERS FOR PRINTING
6039 021332 112237 007700          MOV      (R2)+,DEV2
6040 021336 012237 007702          MOV      (R2)+,DEV3
6041 021342 012237 007704          MOV      (R2)+,DEV4
6042 021346 010246                MOV      R2,-(SP)      ;SAVE R2 ON THE STACK
6043 021350 004737 022564          JSR      PC,SHWOP       ;GO PRINT MODE, MAINT-LOOP TYPE, PARAMTERS.
6044 021354 012602                MOV      (SP)+,R2      ;RESTORE R2
6045 021356 000137 020776          JMP      RPT            ;GO BACK FOR NEXT EVENT ENTRY
6046 021362 012237 007642          RPTTEOP: MOV      (R2)+,EVTADD
6047 021366 012237 007644          MOV      (R2)+,EVTBCT
6048 021372 012237 007646          MOV      (R2)+,EVTTMP
6049 021376                PRINTS   #EVTF4B,EVTADD,EVTBCT,EVTTMP ;PRINT ADDR,RXBYTES,COMPBYTES.
(10) 021376 013746 007646                MOV      EVTTMP,-(SP)
(9) 021402 013746 007644                MOV      EVTBCT,-(SP)
(8) 021406 013746 007642                MOV      EVTADD,-(SP)
(7) 021412 012746 015707                MOV      #EVTF4B,-(SP)
(6) 021416 012746 000004                MOV      #4,-(SP)
(3) 021422 010600                MOV      SP,R0

```

```
(4) 021424 104416 TRAP C$PNTS
(4) 021426 062706 000012 ADD #12,SP
6050
6051 021432 000137 020776 JMP RPT ;THEN GO GET NEXT EVENT ENTRY
6052
6053
6054 021436 012237 007642 RPTDDE: MOV (R2)+,EVTADD ;STORE MESSAGE ADDRESS FOR PRINTING
6055 021442 012237 007644 MOV (R2)+,EVTBCT ;STORE BYTE COUNT FOR PRINTING
6056 021446 012237 007646 MOV (R2)+,EVTTMP ;STORE TOTAL # OF CMP ERRORS
6057 021452 PRINTS #EVT4,EVTADD,EVTBCT,EVTTMP ;PRINT ADDR, BYTE CNT, # CMP ERRS
(10) 021452 013746 007646 MOV EVTTMP,-(SP)
(9) 021456 013746 007644 MOV EVTBCT,-(SP)
(8) 021462 013746 007642 MOV EVTADD,-(SP)
(7) 021466 012746 015507 MOV #EVT4,-(SP)
(6) 021472 012746 000004 MOV #4,-(SP)
(3) 021476 010600 MOV SP,R0
(4) 021500 104416 TRAP C$PNTS
(4) 021502 062706 000012 ADD #12,SP
6058 021506 000137 020776 JMP RPT ;THEN GO GET NEXT EVENT ENTRY
6059
6060 021512 RPTDLE:
6061 021512 012237 007642 RPTDCK: MOV (R2)+,EVTADD ;STORE MSG ADDR FOR PRINT
6062 021516 012237 007644 MOV (R2)+,EVTBCT ;STORE BYTE COUNT
6063 021522 012237 007646 MOV (R2)+,EVTTMP ;STORE BYTE COUNT COMP
6064 021526 PRINTS #EVT4A,EVTADD,EVTBCT,EVTTMP ;PRINT ADDR,RXBYTES,CMPBYTES.
(10) 021526 013746 007646 MOV EVTTMP,-(SP)
(9) 021532 013746 007644 MOV EVTBCT,-(SP)
(8) 021536 013746 007642 MOV EVTADD,-(SP)
(7) 021542 012746 015611 MOV #EVT4A,-(SP)
(6) 021546 012746 000004 MOV #4,-(SP)
(3) 021552 010600 MOV SP,R0
(4) 021554 104416 TRAP C$PNTS
(4) 021556 062706 000012 ADD #12,SP
6065
6066 021562 000137 020776 JMP RPT ;THEN GO GET NEXT EVENT ENTRY
6067
6068
6069
6080
6081
6082 021566 012237 007646 RPTMSC: MOV (R2)+,EVTTMP
6083 021572 PRINTS #EVT3,EVTTMP ;PRINT CHANGE TYPE
(8) 021572 013746 007646 MOV EVTTMP,-(SP)
(7) 021576 012746 015436 MOV #EVT3,-(SP)
(6) 021602 012746 000002 MOV #2,-(SP)
(3) 021606 010600 MOV SP,R0
(4) 021610 104416 TRAP C$PNTS
(4) 021612 062706 000006 ADD #6,SP
6084 021616 012203 MOV (R2)+,R3 ;PUT OLD MODEM STATUS IN R3 FOR PRINTING
6085 021620 004737 021666 JSR PC,RPTMSB ;GO PRINT OLD MODEM STATUS
6086 021624 PRINTS #EVMOCG ;GO PRINT "CHANGED TO:"
(7) 021624 012746 016051 MOV #EVMOCG,-(SP)
(6) 021630 012746 000001 MOV #1,-(SP)
(3) 021634 010600 MOV SP,R0
(4) 021636 104416 TRAP C$PNTS
(4) 021640 062706 000004 ADD #4,SP
```

6087 021644 012203
6088 021646 004737 021666
6089 021652 000137 020776
6090
6091
6092 021656 012604
6093 021660 012603
6094 021662 012602
6095 021664 000207
6096
6097
6098
6099
6100
6101 021666
(7) 021666 012746 016074
(6) 021672 012746 000001
(3) 021676 010600
(4) 021700 104416
(4) 021702 062706 000004
6102 021706 012704 007552
6103 021712 012705 007570
6104 021716 005714
6105 021720 001004
6106 021722 112735 000130
6107 021726 005724
6108 021730 000407
6109 021732 032403
6110 021734 001403
6111 021736 112735 000061
6112 021742 000402
6113 021744 112735 000060
6114 021750 020427 007570
6115 021754 002760
6116 021756
(7) 021756 012746 016154
(6) 021762 012746 000001
(3) 021766 010600
(4) 021770 104416
(4) 021772 062706 000004
6117 021776 000207
6118
6119

```

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  #EVMOHD      ;PRINT MODEM STATUS HEADER
MOV      #EVMOHD,-(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$:      TST      (R4)      ;SEE IF BIT AVAIABLE FROM DEVICE
BNE      7$              ;BR IF THAT MODEM SIG. AVAIABLE
MOVB     #'X,@(R5)+     ;ELSE PUT 'X' IN REPORT IF SIGNAL NOT AVAILBLE
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

```

6121
6122
6123
6124
6125
6126
6127
6128
6129
6130
6131
6132
6133
6134
6135
6136
6137
6138
6139
6140
6141
6142
6143
6144
6145
6146
6147
6148 022000 013702 006504
6149 022004 005003
6150 022006
(8) 022006 010246
(7) 022010 012746 015173
(6) 022014 012746 000002
(3) 022020 010600
(4) 022022 104417
(4) 022024 062706 000006
6151 022030 005737 006510
6152 022034 001416
6153 022036 112237 006530
6154 022042
(8) 022042 005046
(8) 022044 153716 006530
(7) 022050 012746 015155
(6) 022054 012746 000002
(3) 022060 010600
(4) 022062 104417
(4) 022064 062706 000006
6155 022070 000411
6156 022072
(8) 022072 012246
(7) 022074 012746 015164
(6) 022100 012746 000002
(3) 022104 010600
(4) 022106 104417
(4) 022110 062706 000006
6157 022114 020237 006506

.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

CVCLMB DPV-11 DATA COMM. LINK TEST
CVCLMB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81^{H 7} 10:00 PAGE 12-63
DUMP BYTES OR WORDS

SEQ 0085

6158 022120 003005
6159 022122 005203
6160 022124 022703 000010
6161 022130 001725
6162 022132 000736
6163
6164 022134 000207
6165

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

6167
6168
6169
6170
6171
6172
6173
6174
6175
6176
6177
6178
6179
6180
6181
6182
6183
6184
6185
6186
6187
6188
6189
6190 022136 063737 006514 006524
6191 022144 022737 001000 006524
6192 022152 103027
6193
6194
6195
6196 022154
(7) 022154 012746 014126
(6) 022160 012746 000001
(3) 022164 010600
(4) 022166 104417
(4) 022170 062706 000004
6197 022174 163737 006514 006524
6198 022202 012737 001000 006530
6199 022210 163737 006524 006530
6200 022216 013737 006530 006514
6201 022224 013737 006514 006524
6202 022232 000207
6203

.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\$PRINTF
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.
ADDCC: RTS PC ;RETURN TO CALLER

6205
6206
6207
6208
6209
6210
6211
6212
6213
6214
6215
6216
6217
6218
6219
6220
6221
6222
6223
6224
6225
6226
6227
6228
6229
6230
6231
6232
6233
6234
6235
6236
6237
6238
6239
6240
6241
6242
6243
6244
6245
6246
6247
6248
6249
6250
6251
6252
6253
6254
6255

022234
022234 010246
022236 010346
022240 013702 006520
022244 013722 006522
022250 013722 006514
022254 010237 006520
022260 013702 006512
022264 006302
022266 013737 006522 006530
022274 063737 006514 006530
022302 013703 006522
022306 016237 002150 006534
022314 016204 002172
022320 060437 006534
022324 112423
022326 020337 006530
022332 001404
022334 020437 006534
022340 001762
022342 000770
022344 063737 006514 006522
022352 012603
022354 012602
022356 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.
:--

RLDBUF:
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
```

6257
6258
6259
6260
6261
6262
6263
6264
6265
6266
6267
6268
6269
6270
6271
6272
6273
6274
6275
6276
6277
6278
6279
6280
6281
6282
6283
6284
6285
6286
6287
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299
6300
6301
6302
6303
6304
6305
6306
6307
6308
6309
6310
6311
6312

.SBTTL CREATE FACSIMILE OF TX BUFFER AND MESSAGE LIST

..**

THIS ROUTINE ADDED FOR REV B BY EC

FUNCTIONAL DESCRIPTION:

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

CALLING SEQUENCE: JSR PC,FACSIMILE

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

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

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

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

```

6313
6314 022360
6315
6316 022360 010146
6317 022362 005001
6318 022364 116161 003144 005144 10$:
6319 022372 005201
6320 022374 020127 001000
6321 022400 001371
6322
6323 022402 012701 000017 20$:
6324 022406 006301
6325 022410 006301
6326 022412 012737 006144 006440
6327 022420 060137 006440
6328 022424 005001
6329
6330
6331 022426 012737 006144 006436
6332 022434 012737 005144 006446
6333 022442 062737 000002 006436
6334 022450 017737 163762 006444
6335 022456 005037 006442
6336
6337
6338 022462 023737 006456 006442 30$:
6339 022470 001430
6340 022472 013777 006446 163740
6341 022500 062737 000002 006440
6342 022506 013777 006444 163724
6343 022514 062737 000004 006436
6344 022522 063737 006444 006446
6345 022530 017737 163702 006444
6346 022536 062737 000002 006440
6347 022544 005237 006442
6348 022550 000744
6349
6350 022552 013737 006460 006444 40$:
6351
6352
6353 022560 012601
6354 022562 000207
6355
6356

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

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

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

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

40$:
;END WHILE - DO
MOV TTOTCC, CTOTCC ;COPY TOTAL CHARACTER COUNT

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

6358
 6359
 6360
 6361
 6362
 6363
 6364
 6365
 6366
 6367
 6368
 6369
 6370
 6371
 6372
 6373
 6374
 6375
 6376
 6377
 6378
 6379 022564 013702 007676
 6380 022570 006302
 6381 022572 016237 003076 006530
 6382 022600 013702 007700
 6383 022604 006302
 6384 022606 012737 013440 006536
 6385 022614 005702
 6386 022616 001003
 6387 022620 012737 013437 006536
 6388 022626 016237 003114 006532
 6389 022634 013737 007702 006534
 6390 022642
 (11) 022642 013746 006534
 (10) 022646 013746 006532
 (9) 022652 013746 006536
 (8) 022656 013746 006530
 (7) 022662 012746 014211
 (6) 022666 012746 000005
 (3) 022672 010600
 (4) 022674 104416
 (4) 022676 062706 000014
 6391
 6392 022702 005002
 6393 022704 012737 013517 006530
 6394 022712 032737 000001 007704
 6395 022720 001003
 6396 022722 012737 013515 006530
 6397 022730 012737 013530 006532
 6398 022736 032737 000002 007704
 6399 022744 001003
 6400 022746 012737 013526 006532
 6401 022754 012737 013540 006534
 6402 022762 032737 000004 007704
 6403 022770 001003
 6404 022772 012737 013536 006534

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

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-69
SHGW MODE OF OPERATION, LOOP TYPE AND QUALIFIERS

N 7

SEQ 0091

```
6405
6422 023000 012737 013547 006542 3$: MOV #PMS,TEMP5
6423 023006 032737 000010 007704 BIT #MOCHK,DEV4 ;SEE IF /MODEM OR /NOMODEM
6424 023014 001003 BNE 5$ ;BR IF MODEM
6425 023016 012737 013545 006542 MOV #PNMS,TEMP5
6426
6427
6428 023024 5$: PRINTS #SHF1,TEMP,TEMP1,TEMP2,TEMP5 ;,TEMP3,TEMP4 **RFU**
(11) 023024 013746 006542 MOV TEMP5,-(SP)
(10) 023030 013746 006534 MOV TEMP2,-(SP)
(9) 023034 013746 006532 MOV TEMP1,-(SP)
(8) 023040 013746 006530 MOV TEMP,-(SP)
(7) 023044 012746 014247 MOV #SHF1,-(SP)
(6) 023050 012746 000005 MOV #5,-(SP)
(3) 023054 010600 MOV SP,R0
(4) 023056 104416 TRAP ($PNTS
(4) 023060 062706 000014 ADD #14,SP
6429 023064 000207 RTS PC ;RETURN
6430
6431
```

6433
 6434
 6435
 6436
 6437
 6438
 6439
 6440
 6441
 6442
 6443
 6444
 6445
 6446
 6447
 6448
 6449
 6450
 6451 023066
 6452 023066 013704 003126
 6453 023072 013703 003130
 6454 023076 105714
 6455 023100 001441
 6456 023102 121327 000013
 6457 023106 003023
 6458 023110 111305
 6459 023112 006305
 6460 023114 016505 023130
 6461 023120 062705 023130
 6462 023124 004715
 6463 023126 000763
 6464
 6465
 6466 023130 000114
 6467 023132 000134
 6468 023134 000152
 6469 023136 000162
 6470 023140 000204
 6471 023142 000270
 6472 023144 000604
 6473 023146 000650
 6474 023150 000270
 6475 023152 000256
 6476 023154 000736
 6477
 6478
 6479
 6480 023156 121314
 6481 023160 001403
 6482 023162 004737 023226
 6483 023166 000743
 6484 023170 004737 023206
 6485 023174 062703 000004
 6486
 6487 023200 005204
 6488 023202 000735

```
.SBTTL          TRAVERSE COMMAND LINE SUBROUTINES

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

P$TRV:          MOV          P$BUFA,R4
                MOV          P$TREE,R3
P$TR5:          TSTB         (R4)              ;SEE IF ANY CHARS LEFT IN INPUT STRING
                BEQ          P$EXIT           ;BR IF NO
                CMPB         (R3),#11.       ;SEE IF SPECIAL CLI CHAR CODE OR ASCII
                BGT          20$             ;BR IF REGULAR ASCII CHAR.
                MOVB         (R3),R5        ;GET SPECIAL CHAR CODE INTO R5
                ASL          R5
                MOV          10$(R5),R5     ;BUILD TRAVERSE ROUTINE ADDRESS
                ADD          #10$,R5
                JSR          PC,(R5)        ;JSR TO SPECIAL CLI TRAVERSE ROUTINE
                BR           P$TR5         ;GO SEE IF MORE OF STRING LEFT

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

;NOT A SPECIAL CODE
20$:            CMPB         (R3),(R4)       ;SEE IF FIRST CHAR OF STRING IS A MATCH
                BEQ          22$           ;BR IF A MATCH
                JSR          PC,TRVBRC     ;IF NOT A MATCH, GO TAKE MISS BRANCH
                BR           P$TR5        ; THEN GO BACK PT'G TO MISS NODE
22$:            JSR          PC,TRVACT     ;IF A MATCH, GO DO ACTION DEFINED BY
                ADD          #4,R3        ; ACTION CODE IN CLI NODE, THEN
                INC          R4           ; ADJUST PTR TO NEXT CLI NODE
                BR           P$TR5        ;ADJUST BUF PTR TO NEXT CHAR IF MATCH
```

```

6489
6490 023204 000207 P$EXIT: RTS PC ;RETURN FROM PARSER
6491
6492
6493
6494
6495 023206 116302 000001 ;GOTO USER ACTION ROUTINE
6496 023212 042702 177400 TRVACT: MOV 1(R3),R2 ;GET ACTION CODE FROM CLI NODE
6497 023216 013705 003132 BIC #177400,R2 ;CLEAR ANY SIGN EXTENSION
6498 023222 004715 MOV P$ACT,R5 ;GET ADDRESS OF CLI ACTION ROUTINE
6499 023224 000207 JSR PC,(R5) ;GO DO ACTION DEFINED BY CODE
6500 RTS PC ;RETURN TO CALLING CODE
6501
6502 023226 016305 000002 ;TAKE BRANCH IN TREE
6503 023232 060503 TRVBRC: MOV 2(R3),R5 ;GET BRANCH DISPLACEMENT FROM TREE
6504 023234 000207 ADD R5,R3 ;AND POINT R3 TO THE 'MISS' NODE
6505 RTS PC ;RETURN TO P$TRV
6506
6507 023236 062703 000004 ;NO BRANCH TAKEN
6508 023242 000207 TRVNOB: ADD #4,R3 ;THINGS OK, UPDATE R3 TO POINT TO NEXT
6509 RTS PC ;NODE AND RETURN TO P$TRV
6510
6511 023244 004737 023206 TRVERR: JSR PC,TRVACT ;TAKE ERROR ACTION
6512 023250 112737 177777 003143 MOV #1,P$GDBD ;SET ERROR RETURN FLAG
6513 023256 005726 TST (SP)+ ;GET RID OF 'JSR PUSH TO TRVERR'
6514 023260 000137 023204 JMP P$EXIT ;RETURN DIRECT TO EXIT OF P$TRV ROUTINE
6515
6516 023264 004737 023206 TRVEXI: JSR PC,TRVACT ;TAKE EXIT ACTION
6517 023270 105037 003143 CLRB P$GDBD ;SET GOOD/BAD FLAG TO 'SUCCESS (0)'
6518 023274 005726 TST (SP)+ ;GET RID OF 'JSR PUSH TO TRVEXI'
6519 023276 000137 023204 JMP P$EXIT ;RETURN DIRECT TO EXIT OF P$TRV ROUTINE
6520
6521 023302 004737 023206 TRVBR: JSR PC,TRVACT ;GO TAKE BRANCH ACTION
6522 023306 000137 023226 JMP TRVBRC
6523
6524 023312 004737 023206 TRVBIF: JSR PC,TRVACT
6525 023316 105737 003143 TSTB P$GDBD ;SEE IF P$GDBD SET OR CLEARED BY ACTION
6526 023322 001402 BEQ 1$ ;IF CLEAR FALL THRU TO NEXT NODE
6527 023324 000137 023226 JMP TRVBRC ;ELSE TAKE THE 'MISS' BRANCH
6528 023330 000137 023236 1$: JMP TRVNOB ;JUST UPDATE TO NEXT NODE IF THINGS OK
6529
6530 023334 005005 TRVSPA: CLR R5 ;CLEAR 'SPACE OR TAB FOUND' FLAG
6531 023336 121427 000011 1$: CMPB (R4),#11 ;SEE IF CHAR. IN CMD LINE= TAB
6532 023342 001003 BNE 2$ ;BR IF NO, NOT A TAB
6533 023344 005204 INC R4 ;INC INPUT STRING POINTER
6534 023346 005205 INC R5 ;INDICATE A TAB FOUND
6535 023350 000772 BR 1$ ;GO CHECK NEXT CHAR
6536
6537 023352 121427 000040 2$: CMPB (R4),#40 ;SEE IF CHAR. IN CMD LINE= SPACE
6538 023356 001003 BNE 10$ ;BR IF NO, NON-SPACE OR NON-TAB CHAR.
6539 023360 005204 INC R4 ;INC INPUT STRING POINTER
6540 023362 005205 INC R5 ;INDICATE A SPACE FOUND
6541 023364 000764 BR 1$ ;GO CHECK NEXT CHAR
6542 023366 005705 10$: TST R5 ;SEE IF ANY SPACES OR TABS FOUND
6543 023370 001404 BEQ 15$ ;BR IF NO, TAKE NO ACTION
6544 023372 004737 023206 JSR PC,TRVACT ;GO TAKE ACTION IF ANY FOUND

```

```

6545 023376 000137 023236          JMP      TRVNOB          ;JUST GO UPDATE R3 TO NEXT NODE IF OK
6546 023402 000137 023226    15$:    JMP      TRVBRC          ;TAKE BRANCH (MISS) IF NONE FOUND
6547
6548
6549 023406 012737 000012 003140  TRVDEC: MOV      #10.,P$RADX      ;USE DECIMAL AS RADIX AND ASSUME +
6550 023414 000137 023426          JMP      TRVNMA
6551 023420          TRVOCT: ;(SAME AS TRVNUM SINCE DEFAULT RADIX IS OCTAL)
6552 023420 012737 000010 003140  TRVNUM: MOV      #8.,P$RADX      ;USE OCTAL AS RADIX AND ASSUME +
6553 023426 005005          TRVNMA: CLP      R5              ;CLEAR DIGIT COUNTER
6554 023430 121427 000053          CMPB     (R4),#'+'          ;SEE IF THERE'S A + SIGN THERE
6555 023434 001001          BNE      10$              ; BR IF NO
6556 023436 000406          BR       11$              ; ELSE P$RADX ALREADY SAYS +, JUST BR
6557 023440 121427 000055    10$:    CMPB     (R4),#'-'          ;SEE IF THERE'S A - SIGN THERE
6558 023444 001004          BNE      1$              ; BR IF NO
6559 023446 112737 177777 003141  MOVB     #-1,P$RADX+1      ;SET 'MINUS FLAG' (HI BYTE OF P$RADX)
6560 023454 005204    11$:    INC      R4              ;BUMP R4 TO POINT TO FIRST CHAR
6561
6562 023456 121427 000060    1$:    CMPB     (R4),#60          ;SEE IF CHAR. LESS THAN A '0'
6563 023462 002434          BLT      2$              ;BR IF YES (NOT NUMERIC)
6564 023464 121427 000067          CMPB     (R4),#67          ;SEE IF CHAR. GREATER THAN A '7'
6565 023470 003426          BLE      13$             ; BR IF YES
6566 023472 123727 003140 000012  CMPB     P$RADX,#10.       ;SEE IF IN DECIMAL MODE
6567 023500 001417          BEQ      12$             ; BR IF YES (CAN USE HIGHER LIMIT)
6568 023502 121427 000071          CMPB     (R4),#71          ;SEE IF DIGIT WAS A 8 OR 9
6569 023506 003022          BGT      2$              ;BR IF NON-NUMERIC
6570 023510          PRINTF  #CLIBRX          ;ELSE WAS A 8 OR 9 WHEN IN OCTAL RADIX
(7) 023510 012746 011650          MOV      #CLIBRX,-(SP)
(6) 023514 012746 000001          MOV      #1,-(SP)
(3) 023520 010600          MOV      SP,R0
(4) 023522 104417          TRAP     C$PNTF
(4) 023524 062706 000004          ADD      #4,SP
6571 023530 112737 177777 003143  MOVB     #-1,P$GDBD      ;SET ERROR RETURN FLAG
6572 023536 000474          BR       5$              ; PRINT ERROR AND TAKE MISS
6573
6574 023540 121427 000071    12$:    CMPB     (R4),#71          ;SEE IF CHAR. GREATER THAN A '9'
6575 023544 003003          BGT      2$              ;BR IF YES (NOT NUMERIC)
6576 023546 005204    13$:    INC      R4              ;UPDATE CMD LINE PTR TO NEXT CHAR.
6577 023550 005205          INC      R5              ;INDICATE A NUMERIC FOUND
6578 023552 000741          BR       1$              ;GO LOOK AT NEXT CHAR.
6579
6580 023554 005705    2$:    TST      R5              ;SEE IF FOUND ANY NUMERICS
6581 023556 001464          BEQ      5$              ;BR IF NO, TAKE 'MISS' BRANCH
6582 023560 010401          MOV      R4,R1          ;GET POINTER TO START OF NUMERIC STRING
6583 023562 160501          SUB      R5,R1
6584 023564 005037 003136          CLR      P$NUM          ;CLEAR LOC. WHERE VALUE WILL BE STORED
6585 023570 112102    3$:    MOVB     (R1)+,R2        ;GET ASCII CHAR AND CONVERT IT TO A #
6586 023572 162702 000060          SUB      #60,R2
6587 023576 006337 003136          ASL      P$NUM          ;SHIFT CURRENT VALUE TO MAKE ROOM
6588 023602 103437          BCS      7$              ;ERROR IF NUMBER TOO BIG
6589 023604 013737 003136 003134  MOV      P$NUM,P$CNT      ;SAVE FOR LATER IN CASE DECIMAL RADIX
6590 023612 006337 003136          ASL      P$NUM
6591 023616 103431          BCS      7$              ;ERROR IF NUMBER TOO BIG
6592 023620 006337 003136          ASL      P$NUM
6593 023624 103426          BCS      7$              ;ERROR IF NUMBER TOO BIG
6594 023626 123727 003140 000012  CMPB     P$RADX,#10.       ;SEE IF DECIMAL RADIX
6595 023634 001004          BNE      4$              ;BR IF NOT EQUAL

```

```

6596 023636 063737 003134 003136      ADD      P$CNT,P$NUM
6597 023644 103416                    BCS      7$          ;ERROR IF NUMBER TOO BIG
6598 023646 060237 003136      4$:      ADD      R2,P$NUM
6599 023652 103413                    BCS      7$          ;ERROR IF NUMBER TOO BIG
6600 023654 005305                    DEC      R5
6601 023656 001344                    BNE      3$
6602 023660 105737 003141      TSTB    P$RADX+1    ;SEE IF NUM WAS PRECEDED BY A - SIGN
6603 023664 001402                    BEQ      15$        ; BR IF NO
6604 023666 005437 003136      NEG      P$NUM      ; ELSE NEGATE THE NUMBER BEFORE LEAVING
6605 023672 004737 023206      15$:     JSR      PC,TRVACT ;SINCE NUMERIC FOUND, GO TAKE ACTION
6606 023676 000137 023236      JMP      TRVNOB    ;GO POINT R3 TO NEXT NODE
6607
6608 023702      7$:      PRINTF  #CLINBG    ;PRINT NUMBER TOO BIG ERROR
      (7) 023702 012746 011626                    MOV      #CLINBG,-(SP)
      (6) 023706 012746 000001                    MOV      #1,-(SP)
      (3) 023712 010600                    MOV      SP,R0
      (4) 023714 104417                    TRAP    C$PNTF
      (4) 023716 062706 000004                    ADD      #4,SP
6609 023722 112737 177777 003143      MOVB    #-1,P$GDBD ;SET ERROR RETURN FLAG
6610 023730 000137 023226      5$:      JMP      TRVBRC   ;TAKE 'MISS' BRANCH
6611
6612
6613 023734 005005      TRVALP: CLR      R5          ;CLEAR ALPHA FOUND FLAG
6614 023736 121427 000101      1$:      CMPB    (R4),#101    ;SEE IF CHAR. LESS THAN A 'A'
6615 023742 002406                    BLT      2$          ;BR IF YES (NOT ALPHA)
6616 023744 121427 000132      CMPB    (R4),#132    ;SEE IF CHAR. GREATER THAN A 'Z'
6617 023750 003003                    BGT      2$          ;BR IF YES (NOT ALPHA)
6618 023752 005204                    INC      R4          ;UPDATE CMD LINE PTR TO NEXT CHAR
6619 023754 005205                    INC      R5          ;INDICATE AN ALPHA WAS FOUND
6620 023756 000767                    BR       1$          ;GO LOOK AT NEXT CHAR.
6621 023760 005705      2$:      TST      R5          ;SEE IF ANY ALPHA'S WERE FOUND
6622 023762 001404                    BEQ      3$          ;BR IF NO
6623 023764 004737 023206      JSR      PC,TRVACT  ;IF ANY FOUND TAKE ACTION
6624 023770 000137 023236      JMP      TRVNOB    ;THEN UPDATE R3 TO NEXT NODE -NO BRANCH
6625 023774 000137 023226      3$:      JMP      TRVBRC   ;NONE FOUND, TAKE MISS BRANCH
6626
6627 024000 005005      TRVALN: CLR      R5          ;CLEAR ALPHANUM FOUND FLAG
6628 024002 121427 000060      10$:     CMPB    (R4),#60    ;SEE IF CHAR. LESS THAN A '0'
6629 024006 002417                    BLT      2$          ;BR IF YES (NOT NUMERIC OR ALPHA)
6630 024010 121427 000072      CMPB    (R4),#72    ;SEE IF CHAR. GREATER THAN A '9'
6631 024014 003003                    BGT      1$          ;BR IF YES (NOT NUMERIC)
6632 024016 005204                    INC      R4          ;UPDATE CMD LINE PTR TO NEXT CHAR.
6633 024020 005205                    INC      R5          ;INDICATE A NUMERIC FOUND
6634 024022 000767                    BR       10$         ;GO LOOK AT NEXT CHAR.
6635 024024 121427 000101      1$:      CMPB    (R4),#101    ;SEE IF CHAR. LESS THAN A 'A'
6636 024030 002406                    BLT      2$          ;BR IF YES (NOT ALPHA)
6637 024032 121427 000132      CMPB    (R4),#132    ;SEE IF CHAR. GREATER THAN A 'Z'
6638 024036 003003                    BGT      2$          ;BR IF YES (NOT ALPHA)
6639 024040 005204                    INC      R4          ;UPDATE CMD LINE PTR TO NEXT CHAR
6640 024042 005205                    INC      R5          ;INDICATE AN ALPHA FOUND
6641 024044 000756                    BR       10$         ;GO LOOK AT NEXT CHAR.
6642 024046 005705      2$:      TST      R5          ;SEE IF ANY ALPHANUM'S WERE FOUND
6643 024050 001404                    BEQ      3$          ;BR IF NO
6644 024052 004737 023206      JSR      PC,TRVACT  ;IF ANY FOUND TAKE ACTION
6645 024056 000137 023236      JMP      TRVNOB    ;THEN UPDATE R3 TO NEXT NODE -NO BRANCH
6646 024062 000137 023226      3$:      JMP      TRVBRC   ;NONE FOUND, TAKE MISS BRANCH

```

6647					
6648					
6649					
6650	024066	010401		TRVSTR:	MOV R4,R1 ;POINT R1 TO CMD STRING
6651	024070	010305			MOV R3,R5
6652	024072	062705	000006		ADD #6,R5 ;POINT R5 TO MATCH STRING FROM CLI NODE
6653	024076	005037	003134		CLR P\$CNT ;CLEAR CHAR MATCH COUNT
6654	024102	105715		2\$:	TSTB (R5) ;SEE IF END OF MATCH STRING YET
6655	024104	001411			BEQ 10\$;BR IF YES
6656	024106	105711			TSTB (R1) ;SEE IF END OF CMD LINE YET
6657	024110	001407			BEQ 10\$;BR IF YES
6658	024112	121115			CMPB (R1),(R5) ;SEE IF CHARACTERS MATCH
6659	024114	001005			BNE 10\$;BR IF NO
6660	024116	005237	003134		INC P\$CNT ;MATCH -INCREMENT MATCH COUNT
6661	024122	005201			INC R1 ;UPDATE STRING POINTERS
6662	024124	005205			INC R5
6663	024126	000765			BR 2\$;BR TO CONTINUE CHECKING CHARS.
6664					
6665	024130	005737	003134	10\$:	TST P\$CNT ;WHEN DONE SEE IF ANY MATCHES FOUND
6666	024134	001406			BEQ 15\$;BR IF NO, GO TAKE THE MISS BRANCH
6667	024136	010104			MCV R1,R4 ;POINT CMD POINTER TO END OF STRING &
6668	024140	004737	023206		JSR PC,TRVACT ;IF A MATCH FOUND, GO DO MATCH ACTION
6669	024144	066303	000004		ADD 4(R3),R3 ;UPDATE R3 TO NEXT NODE (NO BRANCH)
6670	024150	000207			RTS PC ; (NO RETURN THRU TRVNOB SINCE DIFFERNT
6671					; DISPLACEMENT DUE TO MATCH STRING)
6672	024152	000137	023226	15\$:	JMP TRVBRC ; GO TAKE BRANCH
6673					
6674					; (PARSED OK), -1 IF ILL CMD.....
6675					
6676					

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-75
REPORT CODING SECTION

SEQ 0097

6678
6679
6680
6681
6682
6683
6684
6685
6686 024156
(3) 024156
6687
6699
6700 024156 004737 020302
6701
6702
6709
6710
6717
6718 024162
(3) 024162
(3) 024162 104425

.SBTTL REPORT CODING SECTION

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

BGNRPT

LSRPT::

JSR PC,REPORT

:CALL SUBROUTINE TO DUMP EVENT LOG
: AND BASE TABLE

ENDRPT

L10010:

TRAP CSRPT

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-76
PROTECTION TABLE

SEQ 0098

6720
6721
6722
6723
6724
6725
6726
6727 024164
 (3) 024164
6728
6729 024164 177777
6730 024166 177777
6731 024170 177777
6732
6733 024172
6734

.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

6749
6750
6751
6752
6753
6754
6755
6756
(3)
6757
6781
6782
6783
6784
6785
(3)
6786
6787
6788
(3)
(3)
6789
(2)
6790
(3)
(3)
6791
(2)
6792
(3)
(3)
6793
(2)
6794
6795
(3)
(3)
6796
(2)
6797
6798
6799
6800
6801
6802
6803
(3)
(3)
(3)
6804
(2)
6805
6806
6807
6808
6809

024172
024172
024172 005737 006556
024176 001403
024200 005037 006556
024204
(3) 024204 104444
024206 012737 177777 006560
024214
(3) 024214 012700 000040
(3) 024220 104447
024222
(2) 024222 103417
024224
(3) 024224 012700 000037
(3) 024230 104447
024232
(2) 024232 103513
024234
(3) 024234 012700 000036
(3) 024240 104447
024242
(2) 024242 103002
024244 000137 024714
024250
(3) 024250 012700 000035
(3) 024254 104447
024256
(2) 024256 103521
024260 000523
024262 005037 006560
024266 005037 006620
024272 012702 006614
024276
(3) 024276 012700 000114
(3) 024302 104462
(3) 024304 010001
024306
(2) 024306 103006
024310 004737 017422
024314 012737 000100 006624
024322 000457
024324

.SBTTL INITIALIZE SECTION

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

BGNINIT

LSINIT::

TST DCLFLG
BEQ INIT1
CLR DCLFLG
DOCLN

;CLEANUP AND EXIT? REV B EC
;NO BRANCH REV B EC
;CLEAR FLAG REV B EC
;GO CLEANUP AND EXIT REV B EC
TRAP C\$DCLN

MOV #-1,RESFLG
REDEF #EF.START

;SET RESTART FLAG
;IF HERE CAUSE OF START, DO SOME INIT
MOV #EF.START,RO
TRAP C\$REFG

BCOMPLETE START

REDEF #EF.RESTART

;IF HERE CAUSE OF RESTART, DO SOME INIT
BCS START
MOV #EF.RESTART,RO
TRAP C\$REFG

BCOMPLETE RESTRT

REDEF #EF.CONTINUE

;SEE IF WE'RE HERE CAUSE OF A CONTINUE
BCS #EF.CONTINUE,RO
MOV #EF.CONTINUE,RO
TRAP C\$REFG

BNCOMPLETE S1

;BR IF NOT HERE CAUSE OF CONTINUE
BCC S1

JMP ENDIT
REDEF #EF.NEW

;JMP IF HERE CAUSE OF A CONTINUE
;SEE IF THIS IS A 'NEW PASS'
MOV #EF.NEW,RO
TRAP C\$REFG

BCOMPLETE NEW

;IF YES, BR AROUND LOGUNIT # SETUP
BCS NEW

BR GETPRM

START: CLR RESFLG
CLR CLKVFC

;CLEAR RESTART FLAG SINCE HERE ON START
;CLEAR CLK VECTOR PTR. AS A FLAG IN
; NO CLOCK IS FOUND.
;SETUP R2 AS A PTR. TO CLOCK INFO BLOCK
;LOOK FOR A LINE CLOCK

MOV #CLKCSR,R2
CLOCK L,R1

MOV #L,RO
TRAP C\$CLK
MOV RO,R1

BNCOMPLETE S2

; IF NONE THERE GO LOOK FOR A P-CLOCK
BCC S2
; GO SET UP CLOCK INFO TABLE & CLK VEC.
;SETUP THE ENABLE LINE CLOCK DATA

JSR PC,CLKSET
MOV #LCLKEN,CLKEN
BR RESTRT

S2: CLOCK P,R1

;LOOK FOR A P-CLOCK SINCE NO LINE CLOCK

```

(3) 024324 012700 000120          MOV    #'P,RO
(3) 024330 104462          TRAP   C$CLCK
(3) 024332 010001          MOV    RO,R1
6810 024334          BNCOMPLETE    S3          ; IF NONE THERE GO SEE IF THIS IS LSI
(2) 024334 103017          BCC     S3
6811 024336 004737 017422        JSR    PC,CLKSET          ; ELSE GO SET UP CLOCK INFO & VECTOR
6812 024342 062737 000002 006614    ADD    #2,CLKCSR          ;POINT CLKCSR TO P-CLK COUNT SET REG.
6813 024350 012777 001600 162236    MOV    #PCLKCT,@CLKCSR   ;LOAD CLK SET REG. WITH COUNT VALUE
6814 024356 162737 000002 006614    SUB    #2,CLKCSR          ;POINT CLKCSR BAC TO P-CLK CSR
6815 024364 012737 000111 006624    MOV    #PCLKEN,CLKEN     ;SETUP THE ENABLE THE P-CLK DATA
6816 024372 000433          BR     RESTRT
6817
6818 024374          S3:    READBUS          ;READ BUS TYPE TO SEE IF ON AN LSI
(3) 024374 104407          TRAP   C$RDBU
6819 024376          BNCOMPLETE    S4          ;BR IF NOT, NO CHANCE OF A CLOCK
(2) 024376 103021          BCC     S4
6820 024400 012737 000100 006620    MOV    #100,CLKVEC        ;LOAD 100 AS CLK VECTOR
6821 024406 005037 006616          CLR    CLKBR              ;LOAD 0 AS CLK INT. LEVEL
6822 024412 012737 006624 006614    MOV    #CLKEN,CLKCSR     ;KLUDGE UP THE CSR & ENABLE DATA LOCS
6823 024420          GMANID  L5060,CLKHZ,D,377,50.,60.,YES
(3) 024420 104443          TRAP   C$GMAN
(3) 024422 000406          BR     10000$
(4) 024424 006622          .WORD  CLKHZ
(5) 024426 000052          .WORD  T$CODE
(5) 024430 013573          .WORD  L5060
(5) 024432 000377          .WORD  377
(5) 024434 000062          .WORD  T$LOLIM
(5) 024436 000074          .WORD  T$HILIM
(3) 024440          10000$:
6824 024440 000410          BR     RESTRT
6825
6826 024442          S4:    PRINTF #BDCLK          ;INFORM OPR. NO CLOCK, & EXIT INIT
(7) 024442 012746 013712          MOV    #BDCLK,-(SP)
(6) 024446 012746 000001          MOV    #1,-(SP)
(3) 024452 010600          MOV    SP,RO
(4) 024454 104417          TRAP   C$PNTF
(4) 024456 062706 000004          ADD    #4,SP
6827
6828 024462 005037 006626          RESTRT: CLR    TIMMIN          ;CLEAR TIME SINCE START LOCATIONS
6829 024466 005037 006630          CLR    TIMSEC
6830 024472 013737 006622 006632    MOV    CLKHZ,TIMTCK      ;LOAD TICKS/SEC
6831 024500 012702 006644          MOV    #EVTLOG,R2        ;INIT EVENT TABLE TO ALL 1'S AFTER EACH
6832 024504 10237 006642          MOV    R2,EVTPTR         ; START OR RES AND INIT TABLE POINTER
6833 024510 012722 177777          1$:    MOV    #-1,(R2)+        ;SEE IF REACHED END OF TABLE
6834 024514 020227 007546          CMP    R2,#EVTEND
6835 024520 001373          BNE    1$                ;LOOP UNTIL DONE
6836
6837 024522 012737 177777 006552    NEW:   MOV    #-1,LOGUNT   ;INITIALIZE LOGICAL UNIT #
6838
6839 024530 005237 006552          GETPRM: INC    LOGUNT        ;POINT TO NEXT LOGICAL UNIT
6840 024534 023737 006552 002012    CMP    LOGUNT,L$UNIT     ;SEE IF PAST MAX. LOG. UNIT #
6841 024542 002367          BGE    NEW                ;BR IF YES, AND START OVER
6842
6843 024544          GPWARD LOGUNT,R1        ;GET THE P-TABLE FOR THIS LOG. UNIT
(3) 024544 013700 006552          MOV    LOGUNT,RO
(3) 024550 104442          TRAP   C$GPHRD

```

```

(3) 024552 010001
6844 024554          BNCOMPLETE      GETPRM          ;IF NO P-TABLE AVAIL., GO GET NEXT ONE
(2) 024554 103365          BCC          GETPRM
6845
6846 024556 011137 006566      MOV      (R1),FHDPLX          ;PUT FULL OR HALF DUPLEX ANSWER IN LOC.
6847
6858
6859          ;DEVICE DEPENDENT PART OF GETTING INFO FROM P-TABLE
6860
6861 024562 016137 000002 011416      MOV      2(R1),RXCSR          ;STORE AWAY CSR ADDRESSES
6862
6863
6864 024570 016137 000002 011420      MOV      2(R1),PCSAR
6865 024576 062737 000002 011420      ADD      #2,PCSAR
6866 024604 016137 000002 011422      MOV      2(R1),RDSR
6867 024612 062737 000002 011422      ADD      #2,RDSR
6868 024620 016137 000002 011424      MOV      2(R1),TXCSR
6869 024626 062737 000004 011424      ADD      #4,TXCSR
6870 024634 016137 000002 011426      MOV      2(R1),TDSR
6871 024642 062737 000006 011426      ADD      #6,TDSR
6872
6873 024650 016137 000004 011430      MOV      4(R1),INVEC          ;STORE AWAY INPUT INTERRUPT VECTOR
6874 024656 016137 000004 011432      MOV      4(R1),OUTVEC
6875 024664 062737 000004 011432      ADD      #4,OUTVEC          ;BUILD OUTPUT INTERRUPT VECTOR
6876 024672 016137 000006 011434      MOV      6(R1),INTPRI          ;STORE AWAY INTERRUPT PRIORITY
6877 024700 016137 000012 011436      MOV      12(R1),OPTYP          ;STORE AWAY DEVICE OPTION TYPE
6878 024706 016137 000014 011472      MOV      14(R1),RNODE          ;STORE AWAY THE REMOTE NODE TYPE
6879 024714
6880 024714          ENDIT:      SETVEC  CLKVEC,#CLKINT,#340      ;SETUP CLOCK VECTOR
(7) 024714 012746 000340          MOV      #340,-(SP)
(6) 024720 012746 017446          MOV      #CLKINT,-(SP)
(5) 024724 013746 006620          MOV      CLKVEC,-(SP)
(4) 024730 012746 000003          MOV      #3,-(SP)
(3) 024734 104437          TRAP     C$$SVEC
(2) 024736 062706 000010          ADD      #10,SP
6881
6882          ;DEVICE DEPENDENT VECTOR SETUP
6883
6892 024742          SETVEC  INVEC,#DVRXI,#PRI04      ;SETUP INPUT INTERRUPT VECTOR
(7) 024742 012746 000200          MOV      #PRI04,-(SP)
(6) 024746 012746 034132          MOV      #DVRXI,-(SP)
(5) 024752 013746 011430          MOV      INVEC,-(SP)
(4) 024756 012746 000003          MOV      #3,-(SP)
(3) 024762 104437          TRAP     C$$SVEC
(2) 024764 062706 000010          ADD      #10,SP
6893 024770          SETVEC  OUTVEC,#DVTXI,#PRI04      ;SETUP OUTPUT INTERRUPT VECTOR
(7) 024770 012746 000200          MOV      #PRI04,-(SP)
(6) 024774 012746 034626          MOV      #DVTXI,-(SP)
(5) 025000 013746 011432          MOV      OUTVEC,-(SP)
(4) 025004 012746 000003          MOV      #3,-(SP)
(3) 025010 104437          TRAP     C$$SVEC
(2) 025012 062706 000010          ADD      #10,SP
6894
6895 025016          SETPRI  #PRI00          ;SET THE 'RUN' PRIORITY TO 0
(3) 025016 012700 000000          MOV      #PRI00,R0
(3) 025022 104441          TRAP     C$$SPRI

```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81^{L 8} 10:00 PAGE 12-80
INITIALIZE SECTION

SEQ 0102

6896 025024
(3) 025024 104432
(3) 025026 000002
6897
6909
6910
6911
6912 025030
(3) 025030
(3) 025030 104411

EXIT INIT

TRAP C\$EXIT
.WORD L10012-

.EVEN

ENDINIT

L10012: TRAP C\$INIT

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

M 8
MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-81
AUTODROP SECTION

SEQ 0103

6914
6915
6916
6917
6918
6919
6920
6921
6922
6923 025032
(3) 025032
6924
6931
6932 025032
(3) 025032
(3) 025032 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

6934
6935
6936
6937
6938
6939
6940
6941
(3)
6942
6951
6952
(3)
(3)
6953
(3)
6954
6955
(3)
(3)
6956
6968
6969
6970
6971
(3)
(3)

025034
025034
025034 005077 161554
025040
025040 012700 000340
025044 104441
025046
025046 104433
025050
025050 104432
025052 000002

025054
025054
025054 104412

.SBTTL CLEANUP CODING SECTION

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

BGNCLN

CLR @CLKCSR
SETPRI #PRI07

BRESET

EXIT CLN

.EVEN

ENDCLN

L\$CLEAN::

;DISABLE CLOCK
;SET PROCESSOR PRIORITY BACK TO 7
MOV #PRI07,R0
TRAP C\$SPRI
;CLEAR ALL BEFORE END
TRAP C\$RESET
TRAP C\$EXIT
.WORD L10014-

L10014:
TRAP C\$CLEAN

6973
6974
6975
6976
6977
6978
6979
6980 025056
(3) 025056
6981
6990
6991 025056
(4) 025056 000167
(3) 025060 000000
6992
7004
7005
7006
7007 025062
(3) 025062
(3) 025062 104453

.SBTTL DROP UNIT SECTION

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

BGNDU

L\$DU::

EXIT DU

.WORD JSJMP
.WORD L10015-2-

.EVEN

ENDDU

L10015:
TRAP C\$DU

7009
7010
7011
7012
7013
7014
7015
7016
7017 025064
 (3) 025064
7018
7027
7028 025064
 (4) 025064 000167
 (3) 025066 000000
7029
7041
7042
7043
7044 025070
 (3) 025070
 (3) 025070 104452
7045
7046

.SBTTL ADD UNIT SECTION

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

BGNAU

L\$AU::

EXIT AU

.WORD JSJMP
.WORD L10016-2-

.EVEN

ENDAU

L10016: TRAP C\$AU

7048
 7049
 7050
 7051
 7052
 7053
 7054
 7055
 7056
 7063
 7069
 7070
 (3)
 7071
 7077
 7078
 7079
 7080
 7081
 7082
 7083
 7084
 7085
 7086
 7087
 7088
 7089
 (7)
 (6)
 (3)
 (4)
 (4)
 7090
 7091
 7092
 7093
 7094
 7095
 7096
 7097
 7098
 7099
 7100
 7101
 7102
 7103
 7104
 7105
 7106
 7107
 7108
 7109
 7110
 7111
 7112
 7113

025072
 025072

013777 006624 161514
 025100
 025100 005001
 025102 012737 000001 006634
 025110 005737 006634
 025114 001412
 025116 005301
 025120 001373
 025122
 025122 012746 013736
 025126 012746 000001
 025132 010600
 025134 104417
 025136 062706 000004
 025142 005737 006560
 025146 001112
 025150 005037 006524
 025154 005037 006460
 025160 005037 006444
 025164 012701 006144
 025170 010137 006436
 025174 005037 006434
 025200 012737 006240 006440
 025206 012737 000005 006512
 025214 013737 002162 006514
 025222 012737 003144 006462
 025230 012737 005144 006446
 025236 013737 006462 006522
 025244 013737 006436 006520
 025252 004737 022234
 025256 012737 000001 006456

```

.SBTTL TEST 1: SETUP AND MODES OF OPERATION

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

          BGNTST

          T1::

.SBTTL      PROGRAM SETUP SECTION

MOV      CLKEN,@CLKCSR      ;ENABLE THE CLOCK

GTXRXB:
GTRA2:  CLR      R1
        MOV      #1,TIMER1      ;SET TIMER TO COUNT 1 TICK
        TST      TIMER1        ;CHECK FOR IT TO BE COUNTED OFF
        BEQ      GTRA3        ;BRANCH IF CLOCK EXISTS (COUNTED A TICK)
        DEC      R1
        BNE      1$           ;KEEP CHECKING UNTIL R1 DOES FULL COUNTDOWN
        PRINTF   #NOCLK      ;PRINT BAD CLK MSG AND WARN OF HANG IF TIMEOUT
                                MOV      #NOCLK,-(SP)
                                MOV      #1,-(SP)
                                MOV      SP,R0
                                TRAP    C$PNTF
                                ADD     #4,SP

GTRA3:  TST      RESFLG        ;SEE IF HERE AFTER A RESTART.
        BNE      GTRA5        ;BR IF HERE CAUSE OF A RESTART

; CLEAR COUNTS AND SET UP DEFAULTS

GTRA4:  CLR      TOTCC        ;CLEAR TOTAL CHAR. COUNT TEMP. LOC.
        CLR      TTOTCC       ;CLEAR TOTAL CHAR. COUNT FOR TX BUFF
        CLR      CTOTCC       ;CLEAR TOTAL CHAR. COUNT FOR CMP BUFF
        MOV      #PTRTAB,R1    ;INIT TRANSMIT MESSAGE POINTER
        MOV      R1, TXPTR
        CLR      RXPTR        ; ZERO RX POINTER

MOV      #PTR13,CMPPTR      ;INIT COMPARE MESSAGE POINTER

MOV      #5,MSGTYP          ;SET UP DEFAULT MSG TYPE (QUICK FOX - ITEMP MSG)
MOV      MSG5C,CURCC        ;SET UP DEFAULT CHAR COUNT
MOV      #TXBUF,TCURAD      ;SET UP CURRENT ADD TO START OF TX BUFFER
MOV      #CMPBUF,CCURAD     ;SET UP CURRENT ADD TO START OF CMP BUFFER

MOV      TCURAD,CURADD      ;SETUP CURRENT ADDR TO START OF TXBUF
MOV      TXPTR,CPTR        ;SETUP CURRENT POINTER TABLE POINTER FOR TXBUF
JSR      PC,BLDBUF         ; GO BUILD POINTER TABLE AND BUFFER
MOV      #1,TXMTOT         ;BUMP TOTAL MESSAGE COUNT
  
```

```

7114
7115 025264 013737 006440 006520      MOV      CMPPTR,CPTR      ;SET UP START OF COMPARE POINTER TABLE
7116 025272 013737 006446 006522      MOV      CCURAD,CURADD   ;SET UP CURRENT ADDR. TO START OF CMPBUF
7117 025300 012737 000005 006512      MOV      #5,MSGTYP
7118 025306 013737 002162 006514      MOV      MSG5C,CURCC
7119 025314 004737 022234                JSR      PC,BLDBUF       ;PUT DEFAULT MESSAGE INTO CMPBUF
7120 025320 012737 000001 006442      MOV      #1,CMPTOT       ;BUMP THE COMP MESSG COUNT
7121 025326 012737 000003 006562      MOV      #ACT,MODTYP     ;SET DEFAULT MODE= ACTIVE
7122 025334 005037 006564                CLR      MLTYP           ;SET DEFAULT MAINTENANCE LOOP MODE =NONE
7123 025340 012737 000001 006572      MOV      #1,RPASS        ;SET UP DEFAULT 'RUN PASS' COUNT TO 1
7124 025346 012737 000002 006570      MOV      #2,PARAM        ;SET UP PROG. PARAMETERS - DATACHECKING ENABLED
7125                                     ;
7126 025354                                     PRINTF   #HLP0           ;
(7) 025354 012746 012163                MOV      #HLP0,-(SP)
(6) 025360 012746 000001                MOV      #1,-(SP)
(3) 025364 010600                MOV      SP,RO
(4) 025366 104417                TRAP     C$PNTF
(4) 025370 062706 000004                ADD      #4,SP
7127 025374 013737 006562 007676      GTRA5:  MOV      MODTYP,DEV1
7128 025402 013737 006564 007700      MOV      MLTYP,DEV2
7129 025410 013737 006572 007702      MOV      RPASS,DEV3
7130 025416 013737 006570 007704      MOV      PARAM,DEV4
7131 025424 004737 022564                JSR      PC,SHWOP       ;PRINT TO OPERATOR THE CURRENT MODE.....
7132
7133 025430                MANUAL                 ;SEE IF MANUAL INTERVENTION ALLOWED
(3) 025430 104450                TRAP     C$MANI
7134 025432                BCOMPLETE             GETCL ; BR IF YES (UAM=0 AND NOT CHAINED)
(2) 025432 103412                BCS     GETCL
7135 025434 005737 006572                TST     RPASS          ;SEE IF THIS IS FIRST 'DCLT PASS'
7136 025440 001002                BNF     1$            ; BR IF NOT COMPLETED 1 PASS
7137 025442                EXIT     TST           ; IF DONE 1 PASS IN UNATTENDED MODE - EXIT
(3) 025442 104432                TRAP     C$EXIT
(3) 025444 007676                .WORD   L10017-.
7138 025446 012737 000001 006564      1$:   MOV      #1TL,MLTYP   ;SET UP DEFAULT FOR UNATTENDED MODE
7139 025454 000137 030432                JMP     GTR9           ; 'R M=ACT/LO-I/PAS-1/NOST/CH' AND RUN
7140
7141                .SBTTL              COMMAND LINE FETCH & INTERPRETATION SECTION
7142
7143 025460 105037 003143                GETCL:  CLR     P$GDBD   ;CLEAR CMD LINE PARSING ERROR FLAGS
7144 025464 105037 003142                CLR     P$NNUF
7145 025470                GMANID  CLI$PM,CMDBUF,A,0,1,72.,NO ;GET A COMMAND LINE FROM OPR.
(3) 025470 104443                TRAP     C$GMAN
(3) 025472 000406                BR      10000$
(4) 025474 002666                .WORD   CMDBUF
(5) 025476 000142                .WORD   T$CODE
(5) 025500 011540                .WORD   CLI$PM
(5) 025502 000000                .WORD   0
(5) 025504 000001                .WORD   T$LOLIM
(5) 025506 000110                .WORD   T$HILIM
(3) 025510                10000$:
7146 025510 012737 002666 003126      MOV     #CMDBUF,P$BUFA
7147 025516 012737 007706 003130      MOV     #CLITRE,P$TREE
7148 025524 012737 026434 003132      MOV     #CLIACT,P$ACT
7149 025532 005037 003012                CLR     QUALFG         ;CLEAR QUALIFIER FLAG LOCATION
7150 025536 004737 023066                JSR     PC,P$TRV       ;GO PARSE COMMAND LINE
7151 025542 105737 003143                TSTB    P$GDBD        ;SEE IF PARSED OK OR AN ERROR

```



```
7190 026036 001002      BNE      6$
7191 026040 005037 006456  CLR      TXMTOT
7192 026044 012737 006144 006436 6$:  MOV      #PTRTAB, TXPTR      ;GET POSITION OF END OF TX LIST
7193 026052 013701 006456      MOV      TXMTOT, R1
7194 026056 020127 000017      CMP      R1, #MSG LIM      ;SEE IF MSG COUNT EXCEEDED.
7195 026062 002414      BLT      17$      ; BR IF NO
7196 026064      PRINTF  #MSGTRN, #TABEX      ; ELSE TELL OPR. AND DON'T BUILD MSG.
      (8) 026064 012746 014017      MOV      #TABEX, -(SP)
      (7) 026070 012746 014075      MOV      #MSGTRN, -(SP)
      (6) 026074 012746 000002      MOV      #2, -(SP)
      (3) 026100 010600      MOV      SP, R0
      (4) 026102 104417      TRAP    C$PNTF
      (4) 026104 062706 000006      ADD      #6, SP
7197 026110 000137 025460      JMP      GETCL      ; THEN GO GET A NEW COMMAND.
7198 026114 006301      17$:  ASL      R1      ;# OF MSGS *4 NEXT FREE PTR BLOCK
7199 026116 006301      ASL      R1
7200 026120 060137 006436      ADD      R1, TXPTR
7201 026124 013737 006436 006520  MOV      TXPTR, CPTR      ;SETUP CHAR. COUNT, CURRENT ADDR, & PTR
7202 026132 013737 006462 006522  MOV      TCURAD, CURADD
7203 026140 004737 022136      JSR      PC, ADDCC      ;ADD IN CHAR. COUNT AND CHECK TOTAL
7204 026144 004737 022234      JSR      PC, BLDBUF      ;GO BUILD MESSAGE IN BUFFER AND PTRS.
7205 026150 013737 006520 006436  MOV      CPTR, TXPTR
7206 026156 013737 006524 006460  MOV      TOTCC, TTOTCC      ;UPDATE CHAR. COUNT, CURR ADDR, & PTR
7207 026164 013737 006522 006462  MOV      CURADD, TCURAD
7208 026172 005237 006456      INC      TXMTOT
7209 026176 005337 003014      DEC      QUALVL      ;DEC THE COPY COUNT
7210 026202 001270      BNE      5$
7211 026204 000137 025460      JMP      GETCL
7212
7213 026210 013737 006444 006524 2$:  MOV      CTOTCC, TOTCC      ;SETUP CHAR. COUNT, CURR. ADDR. & PTR
7214 026216 023727 006524 001000  CMP      TOTCC, #BUFLIM      ;SEE IF BUFFER ALREADY FULL
7215 026224 002414      BLT      16$      ; BR IF NOT FULL (BUFLIM # OF CHARS.)
7216 026226      PRINTF  #MSGTRN, #BUFEX      ; ELSE TELL OPR. AND DON'T BUILD MSG.
      (8) 026226 012746 014057      MOV      #BUFEX, -(SP)
      (7) 026232 012746 014075      MOV      #MSGTRN, -(SP)
      (6) 026236 012746 000002      MOV      #2, -(SP)
      (3) 026242 010600      MOV      SP, R0
      (4) 026244 104417      TRAP    C$PNTF
      (4) 026246 062706 000006      ADD      #6, SP
7217 026252 000137 025460      JMP      GETCL      ; THEN GO GET A NEW COMMAND
7218 026256 005737 006444      16$:  TST      CTOTCC      ;IF FIRST "SET" THEN GET RID OF DEFAULT
7219 026262 001002      BNE      7$
7220 026264 005037 006442      CLR      CMPTOT
7221 026270      7$:  MOV      #PTR13, CMPPTR      ;INIT COMPARE MESSAGE POINTER
7222 026270 012737 006240 006440  MOV      CMPTOT, R1
7223 026276 013701 006442      MOV      R1, #MSG LIM      ;SEE IF MSG COUNT EXCEEDED.
7224 026302 020127 000017      CMP      R1, #MSG LIM
7225 026306 002414      BLT      18$      ; BR IF NO
7226 026310      PRINTF  #MSGTRN, #TABEX      ; ELSE TELL OPR. AND DON'T BUILD MSG.
      (8) 026310 012746 014017      MOV      #TABEX, -(SP)
      (7) 026314 012746 014075      MOV      #MSGTRN, -(SP)
      (6) 026320 012746 000002      MOV      #2, -(SP)
      (3) 026324 010600      MOV      SP, R0
      (4) 026326 104417      TRAP    C$PNTF
      (4) 026330 062706 000006      ADD      #6, SP
7227 026334 000137 025460      JMP      GETCL      ; THEN GO GET A NEW COMMAND.
```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-89
COMMAND LINE FETCH & INTERPRETATION SECTION

SEQ 0111

7228	026340	006301		18\$:	ASL	R1		;# OF MSGS *4 = NEXT FREE PTR BLOCK
7229	026342	006301			ASL	R1		
7230	026344	060137	006440		ADD	R1,CMPPTR		
7231	026350	013737	006440	006520	MOV	CMPPTR,CPTR		
7232	026356	013737	006446	006522	MOV	CCURAD,CURADD		
7233	026364	004737	022136		JSR	PC,ADDCC		;ADD IN XHAR. COUNT AND CHECK TOTAL
7234	026370	004737	022234		JSR	PC,BLDBUF		
7235	026374	013737	006520	006440	MOV	CPTR,CMPPTR		
7236	026402	005237	006442		INC	CMPTOT		
7237	026406	013737	006522	006446	MOV	CURADD,CCURAD		;UPDATE CHAR. COUNT, CURR ADDR. & PTR
7238	026414	013737	006524	006444	MOV	TOTCC,CTOTCC		
7239	026422	005337	003014		DEC	QUALVL		;IF COPY WAS GIVEN, PUT MSG IN BUFF
7240	026426	001270			BNE	2\$; AGAIN
7241	026430	000137	025460		JMP	GETCL		;GO BACK UNTIL GET A 'RUN'
7242								
7243								
7244								
7245								
7246								

7248
7249
7250
7251
7252 026434
7253 026434 006302
7254 026436 016202 026452
7255 026442 062702 026452
7256 026446 004712
7257 026450 000207
7258
7259
7260 026452 000150
7261 026454 000152
7262 026456 000162
7263 026460 001550
7264 026462 000262
7265 026464 000172
7266 026466 000306
7267 026470 000400
7268 026472 000722
7269 026474 000732
7270 026476 000750
7271 026500 000760
7272 026502 000770
7273 026504 001062
7274 026506 001556
7275 026510 001102
7276 026512 001162
7277 026514 001170
7278 026516 001200
7279 026520 001210
7280 026522 001220
7281 026524 001230
7282 026526 001246
7283 026530 001334
7284 026532 001344
7285 026534 001364
7286 026536 001372
7287 026540 001402
7288 026542 001412
7289 026544 001422
7290 026546 001450
7291 026550 001460
7292 026552 001564
7293 026554 001600
7294 026556 001632
7295 026560 001642
7296 026562 001652
7297 026564 001662
7298 026566 001672
7299 026570 001702
7300 026572 000142
7301 026574 001140
7302 026576 000656
7303 026600 000706

.SBTTL ACTION TABLE AND ROUTINES
: USER MUST CLEAR/SET P\$GDBD IF USE 'CLIBIF' IN CONNECTION WITH ACTION
: R2 WILL HOLD ACTION CODE FROM PARSING (CLI) NODE
: CLIACT:
ASL R2 ;MULTIPLY ACTION CODE BY 2
MOV 10\$(R2),R2 ;OFFSET VALUE
ADD #10\$,R2 ;ADD BASE VALUE
JSR PC,(R2) ;GO DO ACTION
RTS PC ;RETURN TO TRVACT:
: BRIEF DESCRIPTION OF ACTONS TAKEN
10\$: .WORD ACTNUL-10\$;NULL
.WORD ACTCLR-10\$;CLEAR
.WORD ACTSHO-10\$;SHOW
.WORD ACTCHK-10\$;CHECK
.WORD ACTRUN-10\$;RUN
.WORD ACTHLP-10\$;HELP
.WORD ACTCSE-10\$;CLEAR OR SHOW EXPECT
.WORD ACTCST-10\$;CLEAR OR SHOW TRANSMIT
.WORD ACTSTE-10\$;SET EXPECT
.WORD ACTSTT-10\$;SET TRANSMIT
.WORD ACTSIZE-10\$;SIZE
.WORD ACTCOP-10\$;COPY
.WORD ACTNUM-10\$;NUMERIC VALUE FOR SIZE OR COPY
.WORD ACTOPM-10\$;QUOTED MESSAGE FROM USER
.WORD ACTSTS-10\$;STATUS
.WORD ACTEQO-10\$;END OF QUOTED MESSAGE FROM USER
.WORD ACTMSO-10\$;ONES AS DATA
.WORD ACTMS1-10\$;ZEROS AS DATA
.WORD ACTMS2-10\$;1ALT AS DATA
.WORD ACTMS3-10\$;OACT AS DATA
.WORD ACTMS4-10\$;ITEP AS DATA
.WORD ACTMS5-10\$;CCITT AS DATA
.WORD ACTMS6-10\$;ALPHA AS DATA
.WORD ACTATV-10\$;ACTIVE MODE
.WORD ACTPAS-10\$;PASSIVE MODE
.WORD ACTREC-10\$;RECEIVE MODE
.WORD ACTLIS-10\$;LISTEN MODE
.WORD ACTDLL-10\$;DOWNLINE LOAD
.WORD ACTTRA-10\$;TRANSMIT MODE
.WORD ACTTAL-10\$;TALK MODE
.WORD ACTNO-10\$;NO IE /NOCHECK
.WORD ACTECH-10\$;ECHO
.WORD ACTCRC-10\$;CRC
.WORD ACTPRO-10\$;PROTOCOL
.WORD ACTRPS-10\$;STATUS
.WORD ACTMOP-10\$;SATELLITE IN MAINTFNANCE LOOP MODE
.WORD ACTTLP-10\$;INTERNAL TTL
.WORD ACTCLP-10\$;CABLE LOOP
.WORD ACTLLP-10\$;LOCAL MODEM LOOP
.WORD ACTRLP-10\$;REMOTE MODEM LOOP
.WORD ACTNUF-10\$;MORE COMMAND NEEDED
.WORD ACTBCR-10\$;BAD CHARACTER IN OPERATOR MESSAGE
.WORD ACTDMS-10\$;DUMP MEMORY START ADDRESS
.WORD ACTDME-10\$;DUMP MEMORY END ADDRESS

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

J 9
MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-91
ACTION TABLE AND ROUTINES

SEQ 0113

7304 026602 000700
7305 026604 000246
7306 026606 001572
7307 026610 000236
7308 026612 001272
7309

.WORD ACTDMQ-10\$;DUMP WORD
.WORD ACTPRT-10\$;PRINT
.WORD ACTMOS-10\$;MODEM STATUS
.WORD ACTEXT-10\$;EXIT ROUTINE REV B EC
.WORD ACTSEX-10\$;SET EX=TR REV B EC

7311											
7312	026614	112737	177777	003142	ACTNUF: MOV	#-1,P\$NNUF					;SET FLAG TO SAY NEED MORE OF COMMAND
7313	026622	000207			ACTNUL: RTS	PC					;RETURN TO PARSER
7314											
7315	026624	012737	000001	003010	ACTCLR: MOV	#CLEAR,KEYWD1					;SET LOC TO SAY A CLEAR WAS TYPED
7316	026632	000207			RTS	PC					
7317											
7318	026634	012737	000002	003010	ACTSHD: MOV	#SHOW,KEYWD1					;SET LOC. TO SAY A SHOW WAS TYPED
7319	026642	000207			RTS	PC					
7320											
7321	026644	012702	003016		ACTHLP: MOV	#HLP,KEYWD1					;SETUP R2 AS A POINTER TO HELP MSG TABLE
7322	026650				1\$: PRINTF	#HLPF,(R2)+					;PRINT HELP INFORMATION MESSAGES
(8)	026650	012246								MOV	(R2)+,-(SP)
(7)	026652	012746	012241							MOV	#HLPF,-(SP)
(6)	026656	012746	000002							MOV	#2,-(SP)
(3)	026662	010600								MOV	SP,R0
(4)	026664	104417								TRAP	C\$PNTF
(4)	026666	062706	000006							ADD	#6,SP
7323	026672	020227	003036		CMP	R2,#HLPEND					;SEE IF ALL INFO PRINTED YET
7324	026676	001364			BNE	1\$;IF NO KEEP PRINTING
7325	026700	012737	000005	003010	MOV	#HLP,KEYWD1					;SET LOC. TO SAY A HELP WAS TYPED
7326	026706	000207			RTS	PC					
7327											
7328	026710	012737	000057	003010	ACTEXT: MOV	#EXIT,KEYWD1					;EXIT COMMAND WAS INPUT REV B EC
7329	026716	000207			RTS	PC					;RETURN
7330											
7331	026720	012737	000055	003010	ACTPRT: MOV	#PRNT,KEYWD1					;SET LOC. TO SAY A HELP WAS TYPED
7332	026726	004737	020302		JSR	PC,REPORT					;CALL ROUTINE TO PRINT EVENT LOG AND BASE TABLE
7333	026732	000207			RTS	PC					
7334											
7335	026734	012737	000004	003010	ACTRUN: MOV	#RUN,KEYWD1					;SET RUN FLAG
7336	026742	112737	177777	003142	MOV	#-1,P\$NNUF					;SET FLAG TO SAY NEED MORE OF COMMAND
7337	026750	012737	000001	006572	MOV	#1,RPASS					;SET DEFAULT RUN 'PASS' TO 1
7338	026756	000207			RTS	PC					
7339											
7340	026760	012737	006240	006440	ACTCSE: MOV	#PTR13,CMPPTR					;INIT COMPARE MESSAGE POINTER
7341	026766	013701	006440		MOV	CMPPTR,R1					
7342											
7343	026772	013702	006442		MOV	CMPTOT,R2					
7344	026776	105037	003142		CLRB	P\$NNUF					;FLAG THAT HAVE VALID COMMAND AT THIS PT.
7345	027002	023727	003010	000002	CMP	KEYWD1,#SHOW					;SEE IF A CLEAR OR SHOW WAS TYPED
7346	027010	001471			BEQ	ACTSHW					;BR IF A SHOW WAS TYPED
7347	027012	012737	000001	006442	MOV	#1,CMPTOT					;CLEAR COMPARE MESSAGE COUNT, CHAR. COUNT
7348	027020	005037	006444		CLR	CTOTCC					; AND RESET POINTER
7349											
7350	027024	012737	006240	006440	MOV	#PTR13,CMPPTR					;INIT COMPARE MESSAGE POINTER
7351	027032	013737	006440	006520	MOV	CMPPTR,CPTR					;SET UP TO FILL IN DEFAULT MESSAGE
7352	027040	012701	005144		MOV	#CMPBUF,R1					
7353	027044	010137	006446		MOV	R1,CCURAD					
7354	027050	000431			BR	ACTCLB					
7355											
7356	027052	012701	006144		ACTCST: MOV	#PTRTAB,R1					
7357	027056	013702	006456		MOV	TXMTOT,R2					
7358	027062	105037	003142		CLRB	P\$NNUF					;FLAG THAT HAVE VALID COMMAND AT THIS PT.
7359	027066	023727	003010	000002	CMP	KEYWD1,#SHOW					;SEE IF A CLEAR OR SHOW WAS TYPED
7360	027074	001437			BEQ	ACTSHW					;BR IF A SHOW WAS TYPED

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 12-93
ACTION TABLE AND ROUTINES

SEQ 0115

```

7361 027076 012737 000001 006456      MOV      #1, TXMTOT      ;CLEAR TRANSMIT MESSAGE COUNT, CHAR. COUNT
7362 027104 005037 006460      CLR      TTOTCC        ; AND RESET POINTER
7363 027110 012737 006144 006436      MOV      #PTRTAB, TXPTR
7364 027116 013737 006436 006520      MOV      TXPTR, CPTR
7365 027124 012701 003144      MOV      #TXBUF, R1
7366 027130 010137 006462      MOV      R1, TCURAD
7367
7368 027134 012702 001000      ACTCLB: MOV      #BUFLIM, R2
7369 027140 010137 006522      MOV      R1, CURADD    ;SET UP TO PUT DEFAULT MSG IN LIST AFTER 033'S
7370 027144 012737 000005 006512      MOV      #5, MSGTYP
7371 027152 013737 002162 006514      MOV      MSG5C, CURCC
7372 027160 105021      1$:      CLR      (R1)+        ;FILL EXPT OR TRAN BUFFER WITH 0'S IF A CLEAR
7373 027162 005302      DEC      R2            ;DO 'BUFLIM' NUMBER OF BYTE LOCATIONS
7374 027164 001375      BNE      1$
7375 027166 004737 022234      JSR      PC, BLDBUF    ;'CLEAR' REALLY MEANS TO PUT DEFAULT MSG IN
7376 027172 000207      RTS      PC            ;WHEN DONE, RETURN TO PARSER
7377
7378
7379 027174 012705 003066      ACTSHW: MOV      #SHTAB, R5
7380 027200 122571 000000      5$:      CMP      (R5)+, @ (R1) ;LOOK AT FIRST BYTE OF MSG TO DECIPHER TYPE
7381 027204 001404      BEQ      6$
7382 027206 020527 003075      CMP      R5, #SHTEND  ;SEE IF LOOKED AT ALL OF DEFAULTS YET
7383 027212 001372      BNE      5$
7384 027214 005205      INC      R5            ;MUST BE OPR. SPEC'D THEN
7385 027216 162705 003067      6$:      SUB      #SHTAB+1, R5
7386 027222 006305      ASL      R5
7387 027224 016137 000002 006530      MOV      2(R1), TEMP
7388 027232      PRINTF #SHMSG, SHTYTB(R5), TEMP ;PRINT MSG SIZE & TYPE
(9) 027232 013746 006530      MOV      TEMP, -(SP)
(8) 027236 016546 003046      MOV      SHTYTB(R5), -(SP)
(7) 027242 012746 013232      MOV      #SHMSG, -(SP)
(6) 027246 012746 000003      MOV      #3, -(SP)
(3) 027252 010600      MOV      SP, R0
(4) 027254 104417      TRAP    C$PNIF
(4) 027256 062706 000010      ADD      #10, SP
7389 027262 062701 000004      ADD      #4, R1        ;BUMP R1 TO NEXT SET OF POINTERS
7390 027266 005302      DEC      R2
7391 027270 001341      BNE      ACTSHW
7392 027272 013737 006562 007676      MOV      MODTYP, DEV1
7393 027300 013737 006564 007700      MOV      MLTYP, DEV2
7394 027306 013737 006572 007702      MOV      RPASS, DEV3
7395 027314 013737 006570 007704      MOV      PARAM, DEV4
7396 027322 004737 022564      JSR      PC, SHWOP    ;SHOW THE OPERATOR THE CURRENT MODE..... ALSO
7397 027326 000207      RTS      PC
7398
7399 027330 013737 003136 006504      ACTDMS: MOV      P$NUM, STADD ;SETUP STARTING ADDRESS FOR DUMP
7400 027336 005037 006510      CLR      BYTBIT        ;SET DEFAULT OF WORD DUMP
7401 027342 012737 000052 003010      MOV      #DMPS, KEYWD1 ;FLAG THAT A DUMP WAS TYPED
7402 027350 000403      BR      ACTDME
7403
7404 027352 012737 177777 006510      ACTDMQ: MOV      #-1, BYTBIT ;SET DUMP FLAG TO 'DUMP-WORD'
7405 027360 013737 003136 006506      ACTDME: MOV      P$NUM, ENADD ;SETUP END ADDRESS FOR DUMP (=START IF NO 'EEE'
7406 027366 105037 003142      ACTDMX: CLR      P$NUF ;CLEAR NOT-ENOUGH FLAG, 'DUMP N-N/B' IS VALID
7407 027372 000207      RTS      PC
7408

```

```

7410
7411
7412 027374 012737 000010 003010 ACTSTE: MOV #SETEXP,KEYWD1
7413 027402 000403 BR ACTSTX
7414
7415 027404 012737 000011 003010 ACTSTT: MOV #SETTRN,KEYWD1
7416 027412 012737 000001 003014 ACTSTX: MOV #1,QUALVL ;SET UP DEFAULT COPY TO 1 (/COPY=0)
7417 027420 000207 RTS PC
7418
7419 027422 012737 000012 003012 ACTSIZE: MOV #SIZE,QUALFG
7420 027430 000207 RTS PC
7421
7422 027432 012737 000013 003012 ACTCOP: MOV #QCOPY,QUALFG
7423 027440 000207 RTS PC
7424
7425 027442 023727 003012 000012 ACTNUM: CMP QUALFG,#SIZE ;SEE IF A SIZE OR COPY TYPED
7426 027450 001023 BNE 1$ ;BR IF IT WAS A COPY
7427 027452 005737 003136 TST PSNUM ;CHECK TO BE SURE DIDN'T TRY SIZE=0
7428 027456 001014 BNE 3$ ; BR IF NO
7429 027460 PRINTF #CLISEO
(7) 027460 012746 012042 MOV #CLISEO,-(SP)
(6) 027464 012746 000001 MOV #1,-(SP)
(3) 027470 010600 MOV SP,R0
(4) 027472 104417 TRAP C$PNTF
(4) 027474 062706 000004 ADD #4,SP
7430 027500 112737 177777 003143 MOVB #-1,PSGDBD ;SEE ERROR-IN-CMD FLAG
7431 027506 000411 BR 2$
7432 027510 013737 003136 006514 3$: MOV PSNUM,CURCC ;IF A SIZE LOAD CURCC WITH BYTE COUNT
7433 027516 000405 BR 2$
7434 027520 013737 003136 003014 1$: MOV PSNUM,QUALVL ;IF A COPY, LOAD COPY COUNT
7435 027526 005237 003014 INC QUALVL ;INCREMENT SO FIRST DEC MAKES IT REAL #
7436 027532 000522 2$: BR ACTMEX
7437
7438 027534 012737 000007 006512 ACTOPM: MOV #7,MSGTYP
7439 027542 010437 006530 MOV R4,TEMP ;KEEP TRACK OF START OF QUOTED TEXT
7440 027546 005237 006530 INC TEMP ; SO CAN CALC OPCNT AT END OF QUOTES
7441 027552 000207 RTS PC
7442
7443 027554 010402 ACTEQO: MOV R4,R2
7444 027556 163702 006530 SUB TEMP,R2
7445 027562 010237 006514 MOV R2,CURCC ;CALC BYTE COUNT FOR QUOTED TEXT
7446 027566 010237 002166 MOV R2,OPCNT
7447 027572 013701 006530 MOV TEMP,R1
7448 027576 012705 002520 MOV #OPBUF,R5
7449 027602 112125 1$: MOVB (R1)+,(R5)+ ;COPY QUOTED TEXT TO OPBUF
7450 027604 005302 DEC R2
7451 027606 001375 BNE 1$
7452 027610 000473 BR ACTMEX
7453
7454 027612 ACTBCR: PRINTF #CLIBCR ;BAD CHAR. IN OPR. QUOTED STRING
(7) 027612 012746 011775 MOV #CLIBCR,-(SP)
(6) 027616 012746 000001 MOV #1,-(SP)
(3) 027622 010600 MOV SP,R0
(4) 027624 104417 TRAP C$PNTF
(4) 027626 062706 000004 ADD #4,SP
7455 027632 000207 RTS PC

```


7489	030006	012737	000003	006562	ACTATV: MOV	#ACT,MODTYP	:MODE = ACTIVE
7490	030014	000432			BR	ACTM2X	
7491							
7492	030016	012737	000002	006562	ACTPAS: MOV	#PAS,MODTYP	:MODE = PASSIVE
7493	030024	105037	003142		CLRB	PSNUF	:CLEAR NOT-ENOUGH FLAG
7494	030030	005037	006564		CLR	MLTYP	:CLEAR MAINT LOOP TYPE
7495	030034	000207			RTS	PC	
7496							
7497	030036	005037	006562		ACTREC: CLR	MODTYP	:MODE = RECEIVE
7498	030042	000417			BR	ACTM2X	
7499							
7500	030044	012737	000006	006562	ACTLIS: MOV	#LIS,MODTYP	:MODE = LISTEN
7501	030052	000413			BR	ACTM2X	
7502							
7503	030054	012737	000004	006562	ACTDLL: MOV	#DOW,MODTYP	:MODE = DOWNLINE LOAD
7504	030062	000407			BR	ACTM2X	
7505							
7506	030064	012737	000001	006562	ACTTRA: MOV	#TRA,MODTYP	:MODE = TRANSMIT
7507	030072	000403			BR	ACTM2X	
7508							
7509	030074	012737	000005	006562	ACTTAL: MOV	#TAL,MODTYP	:MODE = TALK
7510							
7511	030102	042737	000004	006570	ACTM2X: BIC	#ECHOB,PARAM	:DISABLE /ECHO (ALL BUT PASSIVE MODE)
7512	030110	105037	003142		CLRB	PSNUF	:CLEAR NOT-ENOUGH FLAG
7513	030114	005037	006564		CLR	MLTYP	:CLEAR MAINT LOOP TYPE
7514	030120	000207			RTS	PC	
7515							

7517	030122	012737	000036	003012	ACTNO:	MOV	#NO,QUALFG		
7518	030130	000207				RTS	PC		
7519									
7520	030132	022737	000036	003012	ACTECH:	CMP	#NO,QUALFG		
7521	030140	001422				BEQ	1\$		
7522	030142	052737	000004	006570		BIS	#ECHOB,PARAM		
7523	030150	022737	000002	006562		CMP	#PAS,MODTYP		;BE SURE IN PASSIVE MODE IF
7524	030156	001416				BEQ	2\$;IF TRYING TO SET /ECHO
7525	030160					PRINTF	#CLINPS		
(7)	030160	012746	011732						MOV #CLINPS,-(SP)
(6)	030164	012746	000001						MOV #1,-(SP)
(3)	030170	010600							MOV SP,RO
(4)	030172	104417							TRAP C\$PNTF
(4)	030174	062706	000004						ADD #4,SP
7526	030200	112737	177777	003143		MOVB	#-1,P\$GDBD		
7527	030206	042737	000004	006570	1\$:	BIC	#ECHOB,PARAM		
7528	030214	005037	003012		2\$:	CLR	QUALFG		;CLEAR 'NO' OUT OF QUALIFIER FLAG
7529	030220	000501				BR	ACTLXX		
7530									
7531	030222	012701	000002		ACTCHK:	MOV	#DATCKB,R1		;SET DATA CHECK BIT
7532	030226	000413				BR	ACTQFG		
7533									
7534	030230	012701	000001		ACTSTS:	MOV	#STATB,R1		;SET THE STATUS BIT
7535	030234	000410				BR	ACTQFG		
7536									
7537	030236	012701	000020		ACTCRC:	MOV	#CRCB,R1		;SET THE CRC BIT
7538	030242	000405				BR	ACTQFG		
7539									
7540	030244	012701	000010		ACTMOS:	MOV	#MOCHK,R1		;SET THE MODEM BIT
7541	030250	000402				BR	ACTQFG		
7542									
7543	030252	012701	000040		ACTPRO:	MOV	#PROTOB,R1		;SET THE PROTOCOL BIT
7544									
7545	030256	050137	006570		ACTQFG:	BIS	R1,PARAM		
7546	030262	022737	000036	003012		CMP	#NO,QUALFG		
7547	030270	001002				BNE	1\$		
7548	030272	040137	006570			BIC	R1,PARAM		
7549	030276	005037	003012		1\$:	CLR	QUALFG		;CLEAR 'NO' OUT OF QUALIFIER FLAG
7550	030302	000450				BR	ACTLXX		
7551									
7552	030304	013737	003136	006572	ACTRPS:	MOV	P\$NUM,RPASS		;GET NUMBER OF 'RUN PASSES'
7553	030312	000444				BR	ACTLXX		
7554									
7555	030314	012737	000005	006564	ACTMOP:	MOV	#5,MLTYP		
7556	030322	000417				BR	ACTLPX		
7557	030324	012737	000001	006564	ACTTLP:	MOV	#1,MLTYP		
7558	030332	000413				BR	ACTLPX		
7559	030334	012737	000002	006564	ACTCLP:	MOV	#2,MLTYP		
7560	030342	000407				BR	ACTLPX		
7561	030344	012737	000003	006564	ACTLLP:	MOV	#3,MLTYP		
7562	030352	000403				BR	ACTLPX		
7563	030354	012737	000004	006564	ACTRLP:	MOV	#4,MLTYP		
7564									
7565	030362	022737	000003	006562	ACTLPX:	CMP	#ACT,MODTYP		;BE SURE IN ACTIVE IF TRYING TO SET LOOP
7566	030370	001415				BEQ	ACTLXX		; BR IF IN ACTIVE
7567	030372	112737	177777	003143		MOVB	#-1,P\$GDBD		

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-4
ACTION TABLE AND ROUTINES

SEQ 0120

7568	030400	005037	006564
7569	030404		
(7)	030404	012746	011670
(6)	030410	012746	000001
(3)	030414	010600	
(4)	030416	104417	
(4)	030420	062706	000004
7570	030424	105037	003142
7571	030430	000207	
7572			

CLR MLTYP
PRINTF #CLIBDL

;CLEAR ANY LOOP TYPE THAT MAY HAVE GOT SET

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

ACTLXX: CLR B P\$NUF
RTS PC

;CLEAR NOT-ENOUGH FLAG

7620
7621
7622
7623
7624
7625
7626
7627
7628
7629
7630
7631
7632
7633
7634
7635
7636
7637
7638
7639
7640
7641
7642

030704
030704 013737 006434 006516
030712 013737 00647? 006470
030720 052737 000104 006574
030726 005037 006520
030732 000137 031074

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,DVRCCT ;SET UP MESSAGE COUNT
BIS #QRX+#ERX,FLAG ;SET UP RX QUE
CLR CPTR ;CLEAR THE TX POINTER
JMP ALLTR ;GO RX.

7644
 7645
 7646
 7647
 7648
 7649
 7650
 7651
 7652
 7653
 7654
 7655
 7656
 7657
 7658
 7659
 7660
 7661
 7662
 7663
 7664
 7665

030736 042737 000002 006570
 030744 013737 006436 006520
 030752 013737 006456 006454
 030760 052737 000210 006574
 030766 005037 006516
 030772 000337 031074

```

.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
TXON2:  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.
  
```

7667
7668
7669
7670
7671
7672
7673
7674
7675
7676
7677
7678
7679
7680
7681
7682
7683
7684
7685
7686
7687
7688
7689
7690
7691

.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

030776
030776 013737 006456 006454
031004 013737 006436 006520
031012 013737 006434 006516
031020 052737 000104 006574
031026 000137 031074

PLCK:
PLCK2: MOV TXMTOT,DVICT ;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.

CVCLMB DPV-11 DATA COMM. LINK TEST
CVCLMB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-9
ACTIVE MODE SECTION

SEQ 0125

7693
7694
7695
7696
7697
7698
7699
7700
7701
7702
7703
7704
7705
7706
7707
7708
7709
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720

031032 013737 006456 006454
031040 013737 006436 006520
031046 013737 006472 006470
031054 013737 006434 006516
031062 052737 000314 006574
031070 000137 031074

..BTTL 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,DVRCT ;SET UP COUNTS
MOV RXPTR,CPTRR
BIS #QRX+#QTX+#ETX+#ERX,FLAG
JMP ALLTR

7722
 7723
 7724
 7725
 7726
 7727
 7728
 7729
 7730
 7731
 7732
 7733
 7734
 7735
 7736
 7737
 7738
 7739
 7740
 7741
 7742
 7743
 7744
 7745
 7746
 7747
 7748
 7749
 7750
 7751
 7752
 7753
 7754
 7755
 7756
 7757
 7758
 7759
 7760
 7761
 7762
 7763
 7764
 7765
 7766
 7767
 7768
 7769
 7770
 7771
 7772
 7773
 7774
 7775
 7776
 7777

.SBTTL TRANSMIT - RECEIVE FOR ALL STANDARD MODES

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

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

USE OF FLAG BITS:
 ORX - 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.
 OTX - 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

031074	032737	000004	006574	ALLTR:	BIT	#ORX, FLAG	
031074	001420			ALCK5:	BEQ	ALCK1	; IF NOT RX GO TO TX'S
031104	013702	006516			MOV	(R2), TEMP2	
031110	011237	006534			MOV	(R2)+, DVRXA	
031114	012237	006464			MOV	(R2), TEMP3	
031120	011237	006536			MOV	(R2), DVRCC	
031124	011237	006466			MOV	R2, CPTRR	
031130	010237	006516			MOV	PC, DVRXQ	; GO QUE DEVICE
031134	004737	033544			JSR	PC, LOGRXQ	; LOG REC QUED
031140	004737	017624			JSR		
031144	032737	000010	006574	ALCK1:	BIT	#OTX, FLAG	

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-11
TRANSMIT - RECEIVE FOR ALL STANDARD MODES

```

7778 031152 001416          BEQ      ALCK2          ;IF NO TX'S GO TO 2
7779 031154 013702 006520    MOV      CPTR,R2
7780 031160 011237 006534    MOV      (R2),TEMP2
7781 031164 012237 006450    MOV      (R2)+,DVTXA
7782 031170 011237 006536    MOV      (R2),TEMP3
7783 031174 012237 006452    MOV      (R2)+,DVTCC
7784 031200 010237 006520    MOV      R2,CPTR
7785 031204 004737 017570    JSR      PC,LOGTXQ
7786
7787 031210 004737 033646    ALCK2:  JSR      PC,DVTXRX ;GO TO TX AND RX SUB ROUT.
7788
7789 031214 032737 000004 006574    BIT      #QRX,FLAG      ;CHECK FOR REC. MSG.
7790 031222 001514          BEQ      ALCK3
7791 031224 013737 006464 006534    MOV      DVRXA,TEMP2
7792 031232 013737 006466 006536    MOV      DVRCC,TEMP3
7793 031240 004737 017642          JSR      PC,LOGRXC      ;LOG REC COMPLETE
7794 031244 032737 000004 006570    UPTABL: BIT      #ECHOB,PARAM ;IS THIS ECHO MODE(PASSIVE)
7795 031252 001406          BEQ      UPTA4          ;IF NOT GO TO 4
7796 031254 013702 006520    MOV      CPTR,R2        ;ELSE SET R2 TO PRESENT TX TABL
7797 031260 013722 006534    MOV      TEMP2,(R2)+    ;STORE OFF RX ADD
7798 031264 013712 006536    MOV      TEMP3,(R2)    ;AND CC
7799 031270 032737 000002 006570    UPTA4:  BIT      #DATCKB,PARAM ;IS DATA CHECKING ASKED FOR
7800 031276 001015          BNE      UPTA1          ;IF SO GO TO 1
7801 031300 012737 000001 006470    MOV      #01,DVRCT      ;ELSE SET DVRCT TO A 1
7802 031306 013737 006434 006516    MOV      RXPTR,CPTRR    ;RESET POINTER
7803 031314 022737 000003 006562    CMP      #ACT,MODTYP    ;IS THIS ACTIVE
7804 031322 001002          BNE      UPTA3
7805 031324 005237 006470    INC      DVRCT          ;IF YES BUMP COUNT
7806 031330 000424          UPTA3:  BR       UPTEX
7807 031332 013702 006516    UPTA1:  MOV      CPTRR,R2
7808 031336 011237 006530    MOV      (R2),TEMP      ;LOAD TEMP WITH PREV. COUNT
7809 031342 163737 006536 006530    SUB      TEMP3,TEMP     ;LOAD TEMP WITH PREV.COUNT-CURRENT
7810 031350 013722 006536    MOV      TEMP3,(R2)+
7811 031354 063737 006536 006534    ADD      TEMP3,TEMP2
7812 031362 013722 006534    MOV      TEMP2,(R2)+    ;STORE OF NEW ADD
7813 031366 013712 006530    MOV      TEMP,(R2)      ;AND NEW CC
7814 031372 162702 000002    SUB      #2,R2          ;PUT POINTER BACK TO ADDR.
7815
7816 031376 010237 006516    MOV      R2,CPTRR      ;AND RESTORE IT.
7817
7818 031402          UPTEX:  CMP      #PAS,MODTYP
7819 031402 022737 000002 006562    BNE      ALCK2A        ;IF NOT PASSIVE LOOP THEN GO TO 2A
7820 031410 001007          BIC      #QRX+#ERX,FLAG ;CLEAR BOTH EXPECTED AND COMPLETED FLAGS
7821 031412 042737 000104 006574    BIS      #QTX+#ETX,FLAG ;SET THE TX FLAGS
7822 031420 052737 000210 006574    BR       ALCK1
7823 031426 000646
7824
7825 031430 005337 006470    ALCK2A: DEC      DVRCT    ;DEC REC COUNT
7826 031434 005737 006470    TST      DVRCT          ;IS IT ALL DONE
7827 031440 001005          BNE      ALCK3          ;NO. GO CHECK TX
7828 031442 042737 000004 006574    BIC      #QRX,FLAG      ;CLEAR THE RX FLAG
7829 031450 005037 006516    CLR      CPTRR          ;YES. CLEAR POINTER
7830 031454 032737 000010 006574    ALCK3:  BIT      #QTX,FLAG ;IS IT TX
7831 031462 001447          BEQ      ALCK4          ;IF NOT TX THEN GO BACK
7832 031464 013737 006450 006534    MOV      DVTXA,TEMP2
7833 031472 013737 006452 006536    MOV      DVTCC,TEMP3   ;LOG TX COMPLETED

```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-12
TRANSMIT - RECEIVE FOR ALL STANDARD MODES

SEQ 0128

7834	031500	004737	017606		JSR	PC,LOGTXC	
7835	031504	005337	006454		DEC	DVTCT	;DEC TX COUNT
7836	031510	022737	000002	006562	CMP	#PAS,MODTYP	
7837	031516	001013			BNE	ALCK3A	;IF NOT PASSIVE MODE GO TO 3A
7838	031520	042737	000210	006574	BIC	#QTX+ETX,FLAG	;CLEAR THE TX FLAGS
7839	031526	052737	000104	006574	BIS	#QRX+ERX,FLAG	;AND SET THE RX FLAGS
7840	031534	005737	006454		TST	DVTCT	
7841	031540	001005			BNE	ALCK3C	;IF MORE RX'S DO IT
7842	031542	000137	031622		JMP	CMPSR	; ELSE COMPARE
7843	031546	005737	006454		ALCK3A:	TST	DVTCT
7844	031552	001402			BEQ	ALCK3B	;IF NOT GO BACK TO 5
7845	031554	000137	031074		ALCK3C:	JMP	ALCK5
7846	031560	005037	006520		ALCK3B:	CLR	CPTR
7847	031564	042737	000010	006574	BIC	#QTX,FLAG	;IF SO CLEAR POINTER
7848	031572	032737	000002	006570	BIT	#DATCKB,PARAM	;CLEAR TX FLAG
7849	031600	001403			BEQ	ALCK4A	;IS IT DAT CK
7850	031602	005737	006516		ALCK4:	TST	CPTRR
7851							
7852	031606	001362			BNE	ALCK3C	;IF SOME RX'S LEFT GO BACK
7853	031610	005737	006520		ALCK4A:	TST	CPTR
7854	031614	001402			BEQ	ALCK4B	;BRANCH IF ANY TX'S LEFT
7855	031616	000137	031210		JMP	ALCK2	
7856	031622				ALCK4B:		
7857							
7858							
7859							

.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

```

7893 031622 032737 000002 006570 CMPSR: BIT #DATCKB,PARAM ;IS DATA CHECKING TO BE DONE
7894 031630 001522 BEQ CMPSEX ;IF NOT THEN EXIT
7895 031632 013737 006434 006520 MOV RXPTR,CPTR ;PUT START OF RX POINTERS TO CPTR
7896 031640 013737 006440 006516 MOV CMPPTR,CPTRR ; AND START OF COMPARE POINTS TO CPTRR
7897 031646 013737 006472 006470 MOV RXMTOT,DVRCT
7899 031654 CMPS3:
7900 031654 013702 006520 MOV CPTR,R2 ;MOVE CURRET RX PT.TO R2
7901 031660 011237 006534 MOV (R2),TEMP2 ;MOVE RX ADD TO EVENT LOG
7902 031664 012201 MOV (R2)+,R1 ;SET R1 TO START ADD OF RX
7903 031666 012237 006536 MOV (R2)+,TEMP3 ;SET CHAR COUNT TO EVENT LOG
7904 031672 010237 006520 MOV R2,CPTR ;RESTORE RX POINT
7906 031676 013702 006516 MOV CPTRR,R2 ;PUT R2 AT COMPARE TABLE
7907 031702 012203 MOV (R2)+,R3 ;SET R3 TO COMPARE ADD
7908 031704 012204 MOV (R2)+,R4 ;SET R4 TO COMP CC
7909 031706 010237 006516 MOV R2,CPTRR ;RESTORE POINTER
7910 031712 010437 006540 MOV R4,TEMP4
7911 031716 004737 017736 JSR PC,LOGCMP ;LOG COMPARE START.
7913 031722 020437 006536 CMP R4,TEMP3 ;IS COMPARE COUNT = TO RX COUNT
7914 031726 001410 BEQ CMPS7 ;IF SO GO TO 7
7915 031730 005237 006502 INC ERRCNT
7916 031734 ERRSOFT 1,EDDLE,ERR10 ;PRINT ERROR
    
```

```

(4) 031734 104457
(5) 031736 000001
(5) 031740 015025
(5) 031742 017244
7917 031744 004737 017754          JSR      PC,LOGCML          ;LOG LENGTH ERROR
7918
7919 031750 005037 006540          CMPS7: CLR      TEMP4          ;CLEAR BAD BYTE COUNTER
7920 031754 012737 000001 006526  MOV      #1,OFFSET        ;SET OFFSET BYTE COUNT TO 1
7921 031762 122123          CMPS1: CMPB     (R1)+,(R3)+    ;COMPARE RX WITH EXPETED
7922 031764 001422          BEQ      CMPS6            ;IF EQUAL THEN GO TO 6
7923
7924 031766 005237 006540          CMPS2: INC      TEMP4          ;INC BAD COUNT
7925 031772 023727 006540 000005  CMP      TEMP4,#5         ;IS IT MORE THEN 5
7926 032000 101014          BHI     CMPS6            ;IF SO GO FOR MORE
7927 032002 114337 006550          MOVB    -(R3),GOOD       ;STORE GOOD BYTE FOR ERROR
7928 032006 114137 006551          MOVB    -(R1),BAD        ;STORE BAD BYTE FOR ERROR
7929 032012 005237 006502          INC     ERRCNT
7930 032016          ERRSOFT 2,EDDDE,ERR1    ;REPORT COMPARISON FAILURE TO OPR.
(4) 032016 104457          TRAP    C$ERSOFT
(5) 032020 000002          .WORD  2
(5) 032022 015062          .WORD  EDDDE
(5) 032024 017154          .WORD  ERR1
7931 032026 005201          INC     R1
7932 032030 005203          INC     R3
7933 032032 005237 006526          CMPS6: INC     OFFSET        ;INC OFFSET
7934 032036 005304          DEC     R4                ;ELSE DEC CHAR COUNT AND SEE IF 0
7935 032040 001350          BNE     CMPS1            ;IF NOT GO BACK
7936 032042 005737 006540          TST    TEMP4            ;SEE IF ANY CMP ERRS FOR THIS MSG
7937 032046 001410          BEQ     CMPS5A          ;BR IF NONE
7938 032050 005237 006502          INC     ERRCNT
7939 032054          ERRSOFT 3,EDDDE,ERR2    ;REPORT # OF MISMATCHES FOR MESSAGE
(4) 032054 104457          TRAP    C$ERSOFT
(5) 032056 000003          .WORD  3
(5) 032060 015062          .WORD  EDDDE
(5) 032062 017216          .WORD  ERR2
7940 032064 004737 017772          CMPS5: JSR      PC,LOGCMD        ;LOG DATA ERROR IN COMPARE
7941 032070          CMPS5A:
7942 032070 005337 006470          DEC     DVRCT
7943 032074 001267          BNE     CMPS3            ;IF NOT ALL DONE GO BACK
7944

```

7946
7947
7948
7949
7950
7951
7952
7953
7954
7955
7956
7957
7958
7959
7960
7961
7962
7963
7964
7965
7966
7967
7968
7969
7970
7971
7972
7973
7974
7975
7976
7977
7978
7979
(4)
(5)
(5)
(5)
7980
7981
7982

032076 005737 011466
032102 001003
032104 005737 011470
032110 001412

032112 005237 006502
032116
032116 104457
032120 000004
032122 014461
032124 017274
032126 005037 011466
032132 005037 011470

.SBTTL MODEM CHANGE REPORTS

..**
FUNCTIONAL DESCRIPTION:
THIS SECTION REPORTS THE NUMBER OF MODEM 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

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

TRAP C\$ERSOFT
.WORD 4
.WORD MSCMS
.WORD ERR4

.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

7984
7985
7986
7987
7988
7989
7990
7991
7992
7993
7994
7995
7996
7997
7998
7999
8000
8001
8002
8003
8004
8005
8006
8007
8008
8009
8010
8011
8012
8013
8014

032136 005237 006500
032142 013737 006476 006540
032150 013737 006500 006534
032156 013737 006502 006536
032164 004737 020010
032170 022737 177777 006572
032176 001403
032200 005337 006572
032204 001402
032206 000137 030636
032212 000137 025374

ENDPS: INC PSCNT ;BUMP PASS COUNT
MOV OPVAR,TEMP4
MOV PSCNT,TEMP2
MOV ERRCNT,TEMP3
JSR PC,LOGEOP ;LOG END OF PASS
CMP #-1,RPASS ;SEE IF RPASS=-1
BEQ 1\$;IF IT IS DON'T DECRMNT, LOOP FOREVER
DEC RPASS ;DEC PASS COUNT
BEQ 2\$;IF DONE EXIT TEST
1\$: JMP GTRX2 ;ELSE GO BACK AND DISPATCH
2\$: JMP GTRAS ;WHEN RPASS=0 GO BACK TO 'DCLT>'

8016
8017
8018
8019
8020
8021
8022
8023
8024
8025
8035
8036 032216
8037 032216
(7) 032216 012746 013632
(6) 032222 012746 000001
(3) 032226 010600
(4) 032230 104417
(4) 032232 062706 000004
8038 032236 000137 025374
8039

.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.
:--

DLL:
PRINTF #DLLCM

JMP GTRAS

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

8041
8042
8043
8044
8045
8046
8047
8048
8049
8050
8051
8052
8053
8054
8055
8056
8057
8058
8059
8060 032242
8061 032242 042737 000002 006570
8062 032250 012702 002520
8063 032254 012722 177777
8064 032260 022702 002642
8065 032264 001373
8066 032266
(3) 032266 104443
(3) 032270 000406
(4) 032272 002520
(5) 032274 000142
(5) 032276 013566
(5) 032300 000000
(5) 032302 000001
(5) 032304 000110
(3) 032306
8067 032306 005002
8068 032310 122762 000377 002520 2\$:
8069 032316 001402
8070 032320 005202
8071 032322 000772
8072 032324 010237 002166 3\$:
8073
8074 032330 012737 002520 006450
8075 032336 012737 002520 006534
8076 032344 013737 002166 006536
8077 032352 013737 002166 006452
8078 032360 004737 017570
8079 032364 052737 000210 006574
8080 032372 005037 006516
8081
8082 032376 004737 033646
8083
8084 032402 013737 006450 006534
8085 032410 013737 006452 006536
8086 032416 004737 017606
8087 032422 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 CS\$GMAN
BR 10001\$
.WORD OPBUF
.WORD T\$CODE
.WORD OPRMM
.WORD 0
.WORD T\$LOLIM
.WORD T\$HILIM

10001\$:

CLR R2 ;NOW GET CHAR COUNT

2\$: 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

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-19
TALK MODE SECTION

SEQ 0135

8088	032430	001304		
8089	032432	022737	052111	002522
8090	032440	001300		
8091	032442	042737	000210	006574
8092	032450	012737	000006	006562
8093	032456	000137	030636	

BNE	TALCK	
CMP	#'IT,OPBUF+2	
BNE	TALCK	
BIC	#QTX+#ETX,FLAG	;CLEAR THE TX BITS
MOV	#LIS,MODTYP	;CHANGE TO LISTEN MODE
JMP	GTRX2	;AND GO BACK TO DISPATCH

8095
8096
8097
8098
8099
8'00
8101
8102
8103
8104
8105
8106
8107
8108
8109
8110
8111
8112
8113
8114
8115
8116
(7)
(6)
(3)
(4)
(4)
8117
8118
8119
8120
8121
8122
8123
8124
8125
8126
8127
8128
8129
8130
8131
8132
8133
8134
(7)
(6)
(3)
(4)
(4)
8135
8136
8137
8138
8139
8140

.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

```

8115 032462 042737 000002 006570 LISCK: BIC #DATCKB,PARAM ;CLEAR CHECK BIT
8116 032470 PRINTF #LISP ;PRINT PROMPT FOR OPR.
      (7) 032470 012746 013555 MOV #LISP,-(SP)
      (6) 032474 012746 000001 MOV #1,-(SP)
      (3) 032500 010600 MOV SP,R0
      (4) 032502 104417 TRAP C$PNTF
      (4) 032504 062706 000004 ADD #4,SP
8117 032510 012737 002520 006464 LISCKA: MOV #OPBUF,DVRXA ;SET DEVICE UP TO REC AT OPBUF
8118 032516 012737 002520 006534 MOV #OPBUF,TEMP2
8119 032524 012737 000122 006466 MOV #82.,DVRCC ;SET UP CHAR COUNT TO 82.
8120 032532 012737 000122 006536 MOV #82.,TEMP3
8121 032540 052737 000104 006574 BIS #QRX+#ERX,FLAG ;SET UP FLAG
8122 032546 005037 006520 CLR CPTR ;CLEAR THE TX.
8124 032552 004737 033544 JSR PC,DVRXQ ;QUE RX
8125 032556 004737 017624 JSR PC,LOGRXQ
8127 032562 004737 033646 JSR PC,DVTXRX ;GO TO DEVICE RX. SUBROUTINE
8129 032566 013737 006464 006534 MOV DVRXA,TEMP2
8130 032574 013737 006466 006536 MOV DVRCC,TEMP3 ;SET UP ADDR.AND CC.
8131 032602 004737 017642 JSR PC,LOGRXC ;LOG COMPLETED
8132 032606 063737 006464 006466 ADD DVRXA,DVRCC
8133 032614 105077 153646 CLRB @DVRCC
8134 032620 PRINTF #OPBFPT
      (7) 032620 012746 002514 MOV #OPBFPT,-(SP)
      (6) 032624 012746 000001 MOV #1,-(SP)
      (3) 032630 010600 MOV SP,R0
      (4) 032632 104417 TRAP C$PNTF
      (4) 032634 062706 000004 ADD #4,SP
8135 032640 022737 054105 002520 CMP #'EX,OPBUF ;COMPARE FOR EX OF 'EXIT'
8136 032646 001320 BNE LISCKA ;IF NOT EXIT THEN GO BACK
8137 032650 022737 052111 002522 CMP #'IT,OPBUF+2 ;IF FIRST HALF OK CHECK NEXT PART
8138 032656 001314 BNE LISCKA ;IF NOT EXIT THE GO BACK
8139 032660 012737 000005 006562 MOV #TAL,MODIYP ;CHANGE MODE TO TALK
8140 032666 000137 030636 JMP GTRX2 ;RETURN TO DISPATCHER
  
```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81^{H 11} 10:00 PAGE 13-21
LISTEN MODE SECTION

SEQ 0137

8141
8142

8144
8145
8163
8164
8165
8166
8167
8198
8199
8200
8201
8202
8203
8204
8205
8206
8207
8208
8209
8210
8211
8212
8213
8214
8215
8216
8217
8218
8219
8220
8221
8222
8223
8224
8225
8226
8227
8228
8229
8230
8231
8232
8233
8234
8235
8236
8237
8238
8239
8240
8241
8242
(4)
(5)
(5)

032672

032672 012777 000001 156524
 032700 105777 156512
 032704 001423
 032706 104422
 032710 012737 016441 006534
 032716 017737 156474 006536
 032724 017737 156474 006540
 032732 004737 017652
 032736 005237 006502
 032742 104457
 032744 000005
 032746 016441

.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     C$BRK

: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     C$ERSOFT
.WORD   5
.WORD   DVEMO

```

```

(5) 032750 017326
8243 032752 000747          BR DVINIT          ;GO BACK AND TRY MSTR CLR AGAIN IF ERROR
8244
8245                          ;SET TTL LOOP IF REQU'D
8246
8247 032754 042737 000003 006574 DVIN1: BIC      #?,FLAG      ;CLEAR INPUT AND OUTPUT INT FLAGS
8248 032762 042777 000010 156434 BIC      #TTL,@TXCSR  ;CLEAR INTERNAL LOOP
8249 032770 022737 000001 006564 CMP      #TTL,MLTYP   ;IS TTL SELECTED
8250 032776 001004 BNE      DVIN3        ; IF NOT GO TO 3
8251 033000 052777 000010 156416 BIS      #TTL,@TXCSR  ;ELSE SET INTERNAL LOOP
8252 033006 000455 BR       DVIN37
8253
8254 033010 022737 000002 006564 DVIN3:  CMP      #CABLE,MLTYP
8255 033016 001451 BEQ      DVIN37      ;IF CABLE LOOP SKIP CHECK
8256                          ;FOR MODEM READY
8257
8258 033020 022737 000004 006562 CMP      #DOW,MODTYP  ;CHECK IF DLL
8259 033026 001002 BNE      DVIN3A       ;BRANCH IF NOT DLL
8260 033030 000137 033460 JMP      DVINEX       ;ELSE EXIT
8261
8262 033034 012777 000002 156354 DVIN3A: MOV     #DTR,@RXCSR ;SET UP DTR.
8263
8264 033042 012737 002000 006634 DVIN38: MOV     #2000,TIMER1
8265 033050 005737 006634 TST     TIMER1
8266 033054 001022 BNE      DVIN39       ;IF TIMER NOT OUT GO TO 39
8267
8268                          ;SET ERROR FOR NO MODEM READY
8269
8270 033056 012737 017067 006534 MOV     #DVEM6,TEMP2
8271 033064 017737 156326 006536 MOV     @RXCSR,TEMP3
8272 033072 017737 156326 006540 MOV     @TXCSR,TEMP4
8273 033100 004737 017652 JSR     PC,LGDVE
8274 033104 005237 006502 INC     ERRCNT
8275 033110 ERRSOFT 11,DVEM6,ERR13
(4) 033110 104457
(5) 033112 000013
(5) 033114 017067
(5) 033116 017326
8276 033120 000745
8277 033122 DVIN39: BR     DVIN3A      ;THEN TRY TO SET DTR AGAIN
(3) 033122 104422
8278 033124 017737 156266 011444 MOV     @RXCSR,IRXCSR ;GET COPY OR RXCSR
8279 033132 032737 001000 011444 BIT     #BIT9,IRXCSR  ;IS MODEM READY SET
8280 033140 001743 BEQ     DVIN38
8281 033142 013777 011440 156250 DVIN37: MOV    DPVF1,@PCASAR ;SET UP PCASAR
8282 033150 005737 011472 TST     RNODE         ;CHECK REMOTE NODE
8283 033154 001141 BNE     DVINEX        ;EXIT IF ITEP
8284 033156 005737 006566 TST     FHDPLX        ;IS THIS FULL DUPLEX
8285 033162 001536 BEQ     DVINEX        ;BANCH IF NOT
8286
8287                          ;SET UP TO SEND STRT
8288 033164 112737 000005 002645 MOV     #5,HDMMSG+1   ;SET UP ENQ
8289 033172 052737 000060 006574 BIS     #RXM!TXM,FLAG ;SET FLAG WORD
8290 033200 012737 000074 006640 MOV     #60.,TIMERS   ;SET TIMER FOR 1 MINUTE
8291 033206 004737 035136 JSR     PC,CTSSR      ;SET CTS IF NESC.
8292 033212 012737 000006 002646 DVIN41: MOV    #6,HDMCC   ;SET UP STRT CODE

```

```

TRAP  CSERSOFT
.WORD 11
.WORD DVEM6
.WORD ERR13

```

```

TRAP  CSBRK

```

```

8293 033220 004737 035006      JSR    PC,DVIN31      ;GO TX STRT AND CHK FOR RX.
8294 033224 005737 006640      TST    TIMERS
8295 033230 001466              BEQ    DVIN81         ;IF TIMER EXPIERED EXIT
8296
8297 033232 022737 000006 002660 DVIN4:  CMP    #6,RHDMCC     ;IS THE RCVD=STRT
8298 033240 001441              BEQ    DVIN8         ;IF SO GO TO ASTRT
8299 033242 022737 000007 002660      CMP    #7,RHDMCC     ;IS IT A STACK
8300 033250 001360              BNE    DVIN41        ;IF NOT STACK ETIHER GO BACK
8301
8302 033252 004737 035136      DVIN9: JSR    PC,CTSSR     ;SET REQUEST TO SEND
8303 033256 042737 001010 006574      BIC    #QTX.PAD,FLAG ;CLEAR TX COMPT FLAG.
8304 033264 012737 000001 002646      MOV    #1,HDMCC      ;SET UP ACK
8305 033272 012737 002645 011450      MOV    #HDMMSG+1,MSGPTR ;SET UP POINTER
8306 033300 013737 002654 011452      MOV    HDMC,MSGCC
8307 033306 012737 000010 011454      MOV    #8.,SYNCC     ;SET UP SYNC COUNT
8308 033314 052777 000120 156102      BIS    #TXENA!TINTEN,@TXCSR
8309 033322 032737 000010 006574 DIVN91: BIT    #QTX,FLAG
8310 033330 001053              BNE    DVINEX        ;EXIT IF ACK SENT
8311 033332
(3) 033332 104422                                TRAP    C$BRK
8312 033334 005737 006640      TST    TIMERS
8313 033340 001370              BNE    DIVN91        ;IF NOT TIMER EXPIRED RECHK TX.
8314 033342 000421              BR     DVIN81        ;IF TIMER OUT REPORT IT
8315
8316 033344 012737 000007 002646 DVIN8:  MOV    #7,HDMCC      ;SET POTINTER TO STACK
8317 033352 004737 035006              JSR    PC,DVIN31     ;AND GO SEND STACK
8318 033356 005737 006640      TST    TIMERS
8319 033362 001411              BEQ    DVIN81        ;REPORT ERROR IF TIME OUT
8320 033364 022737 000001 002660      CMP    #1,RHDMCC     ;IS IT ACK RCVD?
8321 033372 001432              BEQ    DVINEX        ;IF SO EXIT
8322 033374 022737 000007 002660      CMP    #7,RHDMCC     ;IS IT STACK RCVD
8323 033402 001723              BEQ    DVIN9         ;IF SO SEND ACK
8324 033404 000757              BR     DVIN8         ;IF NEITHER SEND ANOTHER ACK
8325
8326                                ;DO ERROR AND REPEAT
8327
8328 033406 012737 017003 006534 DVIN81: MOV    #DVEM5,TEMP2
8329 033414 013737 002660 006536      MOV    RHDMCC,TEMP3
8330 033422 013737 002646 006540      MOV    HDMCC,TEMP4
8331 033430 004737 017652              JSR    PC,LGDVE
8332 033434 005237 006502              INC    ERRCNT
8333 033440              ERRSOFT 10.,DVEM5,ERR13
(4) 033440 104457                                TRAP    C$ERSOFT
(5) 033442 000012              .WORD 10
(5) 033444 017003              .WORD DVEM5
(5) 033446 017326              .WORD ERR13
8334 033450 005237 006476              INC    OPVAR
8335 033454 000137 032672              JMP    DVINIT        ;TRY ALL OVER AGAIN
8336
8337 033460 004737 035310      DVINEX: JSR    PC,CLRRTS   ;CLEAR RTS IF NESC
8338 033464 042737 173777 006574      BIC    #173777,FLAG  ;CLEAR FLAG WORD
8339 033472 052737 002000 006574      BIS    #INOV,FLAG    ;SET THE INITT OVER FLAG
8340 033500 000207              RTS    PC            ;RETURN TO CALLER
8341
8342

```

8344
8345
8355
8356
8357
8358
8359
8360
8361
8362
8363
8364
8365
8366
8367
8368
8369
8370
8371
8372
8373
8374
8375
8376
8377
8378
8379
8380
8381

.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
:--

DVMODS: MOV @RXCSR,MODS
BIC #BIT5,MODS ;CLEAR BIT 5
BIT #BIT5,@TXCSR ;SEE IT TM OR SQ SET
BEQ DVMEX ;IF NOT EXIT
BIS #BIT5,MODS ;IF SET SET BIT 5 IN MODS
DVMFX: BIC #106720,MODS ;CLEAR ALL UNUSED BITS
RTS PC ;RETURN TO CALLER

8383
 8395
 8396
 8397
 8398
 8399
 8400
 8401
 8402
 8403
 8404
 8405
 8406
 8407
 8408
 8409
 8410
 8411
 8412
 8413
 8414
 8415
 8416
 8417
 8418
 8419
 8420
 8421
 8422
 8423
 8424
 8425
 8426
 8427
 8428
 8429
 8430
 8431
 8432
 8433

033544				
033544	032737	000004	006574	
033552	001434			
033554	042737	000444	006574	
033562	005737	011472		
033566	001415			
033570	052737	000440	006574	
033576	013737	006464	011460	
033604	012737	000072	011462	
033612	012737	000070	006466	
033620	000406			
033622	012737	002657	011460	
033630	013737	002654	011462	
033636	052777	000160	155552	
033644	000207			

```
.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
:--

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 ITEP 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 ITEP 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
```

8435
8436
8465
8466
8467
8468
8469
8470
8471
8472
8473
8474
8475
8476
8477
8478
8479
8480
8481
8482
8483
8484
8485
8486
8487
8488
8489
8490
8491
8492
8493
8494
8495
8496
8497
8498
8499
8500
8501
8502
8503
8504
8505
8506
8507
8508
8509
8510
8511
8512
8513
8514
8515
8516
8517
8518

```
.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,HDMCC
        MOV    HDMC,MSGCC        ;SET CC FOR HEADER
DVTR2:  MOV    #177,SYNCC        ;SET UP FOR 177 SYNCs.
        BIS    #TXENA!#TINTEN,@TXCSR
```

```

8519 033766 012737 000074 006640 DVTR3: MOV #60.,TIMERS ;SET TIMER FOR 60 SECS
8520
8521 033774 DVTR8: BREAK
(3) 033774 104422
8522 033776 005737 006640 TST TIMERS ;IS TIMER EXPIRED TRAP C$BRK
8523 034002 001022 BNE TOINOT
8524
8525 ;LOG ERROR TIME OUT RX OR TX NOT COMPLETED
8526
8527 034004 012737 016602 006534 MOV #DVEM2,TEMP2
8528 034012 017737 155400 006536 MOV @RXCSR,TEMP3
8529 034020 017737 155400 006540 MOV @TXCSR,TEMP4
8530 034026 004737 017652 JSR PC, LGDVE
8531 034032 005237 006502 INC ERRCNT
8532 034036 ERRSOF T 7,DVEM2,ERR13
(4) 034036 104457 TRAP C$ERSOF T
(5) 034040 000007 .WORD 7
(5) 034042 016602 .WORD DVEM2
(5) 034044 017326 .WORD ERR13
8533 034046 000747 BR DVTR3 ;RETURN TO CHECK TIMER
8534
8535 034050 032737 000010 006574 TOINOT: BIT #QTX,FLAG ;IS IT TX COMPL?
8536 034056 001406 BEQ DVTR4 ;BRANCH IF TX NOT DONE.
8537 034060 004737 035310 JSR PC, CLRRTS
8538 034064 032737 000100 006574 BIT #ERX,FLAG ;ARE WE EXPECTING TO RX
8539 034072 001416 BEQ DVTREX ;BRANCH IF NOT.
8540
8541 034074 032737 000004 006574 DVTR4: BIT #QRX,FLAG ;IS RX DONE
8542 034102 001734 BEQ DVTR8 ;GO BACK AND TIME IF NOT
8543
8544 034104 032737 000200 006574 BIT #ETX,FLAG ;ARE WE EXPECTG TO TX.
8545 034112 001406 BEQ DVTREX ;BRANCH IF NOT.
8546
8547 034114 032737 000010 006574 BIT #QTX,FLAG ;IS IT TX COMPLETED
8548 034122 001724 BEQ DVTR8 ;GO BACK AND TIME OUT
8549 034124 004737 035310 JSR PC, CLRRTS ;CLEAR RTS IF NESC.
8550 034130 000207 DVTREX: RTS PC ;AND EXIT
8551

```

8553
8554
8563
8564
8565
8566
8567
8574
8575
8576
8577
8578
8579
8580
8581
8582
8583
8584
8585
8586
8587
8588
8589
8590
8591
8592
8593
8594
8595
8596
8597
8598
8599
8600
8601
8602
8603
8604
8605
8606
8607
8608
8609
8610
8611
(3)
8612
8613
8614
8615
8616
8617
8618
8619
8620
8621

034132
034132
034132 017737 155260 011444
034140 032737 000010 006570
034146 001456
034150 032737 002000 006574
034156 001452
034160 005737 011444
034164 100047
034166 013737 011444 011442
034174 042737 106760 011442
034202 032777 000040 155214

: 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
'ORX' 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 'ORX' 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

BGNSRV DVRXI
DVRXI::
MOV @RXCSR,IRXCSR ;MOV RX CSR TO IMAGE
BIT #MOCHK,PARAM ;ANY MODEM CHANGES TO REPORT
BEQ RXIN21 ;IF NO* 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

```

8622 034210 001403          BEQ    RXIN2          ;IF TEST MODE SET
8623 034212 052737 000040 011442    BIS    #TM,CMODS     ;SET IT IN NEW STATUS
8624 034220 013737 011442 006536    RXIN2: MOV    CMODS,TEMP3
8625 034226 013737 007550 006540    MOV    MODS,TEMP4
8626 034234 023737 006540 006536    CMP    TEMP4,TEMP3   ;COMPARE OLD TO CURRENT
8627 034242 001406          BEQ    GLINC          ;INC GLITCH COUNT
8628 034244 005237 011470          INC    MHRCNT         ;INC HARD COUNT
8629 034250 012737 016411 006534    MOV    #HRDMSG,TEMP2 ;SET UP HARD MESG.
8630 034256 000405          BR     RXIN1
8631 034260 005237 011466          GLINC: INC    MGLCNT   ;INC GLITCH COUNT
8632 034264 012737 016363 006534    MOV    #GLMSG,TEMP2  ;SET UP GLITCH
8633 034272 004737 020026          RXIN1: JSR    PC,LOGMSC ;GO LOG MODEM STATUS CHANGE
8634 034276 013737 011442 007550    MOV    CMODS,MODS    ;MOVE CURRENT TO OLD
8635
8636          ;TEST FOR STATUS OR DATA
8637
8638 034304 032737 002200 011444    RXIN21: BIT   #RSTARY!RDATRY,IRXCSR
8639 034312 001544          BEQ    RXINEX         ;IF NEITHER EXIT
8640 034314 017737 155102 011446    MOV    @RDSR,IRDSR
8641 034322 032737 000200 011444    BIT   #RDATRY,IRXCSR ;IS THIS DATA
8642 034330 001455          BEQ    RXIN3          ;IF NOT GO TO 3
8643
8644          ;GET HERE WITH GOOD DATA
8645
8646 034332 013702 011460          RXIN4: MOV    RMSGPT,R2
8647 034336 113722 011446          MOV    IRDSR,(R2)+   ;STORE DATA AWAY
8648 034342 010237 011460          MOV    R2,RMSGPT    ;PUT POINTER BACK
8649
8650
8651 034346 005337 011462          DEC    RMSGCC
8652 034352 001124          BNE   RXINEX         ;GET OUT IF NOT ALL DONE
8653 034354 032737 000400 006574    BIT   #BCC,FLAG     ;IS THE BCC FLAG ALREADY SE
8654 034362 001066          BNE   RXIN5         ;BRANCH IF YES.
8655 034364 032737 100000 011446    BIT   #RERR,IRDSR   ;IS THE ERR CHK BIT SET INDICATING
8656          ;GOOD BCC.
8657          ;BRANCH IF GOO
8658 034374 013737 011446 006536    MOV    IRDSR,TEMP3
8659 034402 013737 011444 006540    MOV    IRXCSR,TEMP4
8660 034410 012737 016677 006534    MOV    #DVEM3,TEMP2
8661 034416 004737 017652          JSR    PC,LGDVE
8662 034422 005237 006502          INC    ERRCNT
8663 034426          ERRSOFT 8,DVEM3,ERR13
(4) 034426 104457
(5) 034430 000010
(5) 034432 016677
(5) 034434 017326
8664
8665 034436 000467          BR     RXIN8         ;DISABLE INTERRUPTS AND EXIT
8666
8667 034440 052737 000400 006574    RXIN6: BIS    #BCC,FLAG ;SET FLAG
8668 034446 012737 000002 011462    MOV    #2.,RMSGCC   ;SET THE COUNT TO 2
8669 034454 012737 011464 011460    MOV    #BCCW,RMSGPT ;SET POINTER TO BCC WORD
8670 034462 000460          BR     RXINEX
8671
8672          ;STATUS CHECK
8673

```

```

TRAP    C$ERSOFT
.WORD   8
.WORD   DVEM3
.WORD   ERR13

```

```

8674 034464 032737 002000 011444 RXIN3: BIT #RSTARY,IRXCSR ;IS THIS A STATUS INT.
8675 034472 001454 BEQ RXINEX ;EXIT IF NOT
8676
8677 ;LOG OVERRUN ERROR
8678
8679 034474 012737 016737 006534 MOV #DVEM4,TEMP2
8680 034502 013737 011446 006536 MOV IRDSR,TEMP3
8681 034510 013737 011444 006540 MOV IRXCSR,TEMP4
8682 034516 004737 017652 JSR PC,LGDVE
8683 034522 005237 006502 INC ERRCNT
8684 034526 ERRSOFT 9,DVEM4,ERR13
(4) 034526 104457 TRAP C$ERSOFT
(5) 034530 000011 .WORD 9
(5) 034532 016737 .WORD DVEM4
(5) 034534 017326 .WORD ERR13
8685 034536 000424 BR RXIN7
8686
8687 034540 032737 000040 006574 RXIN5: BIT #RXM,FLAG ;IS THE RX M BODY BIT SET
8688 034546 001020 BNE RXIN7 ;IF YES THEN ALL DONE
8689 034550 052737 000040 006574 BIS #RXM,FLAG
8690 034556 042737 000400 006574 BIC #BCC,FLAG ;CLEAR BCC AND SET RXM
8691 034564 013737 006464 011460 MOV DVRXA,RMSGPT ;MOVE ADDRESS TO POINTER
8692 034572 013737 002660 011462 MOV RMDMCC,RMSGCC ;MOVE THE CHAR COUNT IN
8693 034600 013737 002660 006466 MOV RMDMCC,DVRCC ;SET THE CC TO AMOUNT IN HEADER
8694 034606 000406 BR RXINEX ;AND FINISH.
8695
8696 034610 052737 000004 006574 RXIN7: BIS #QRX,FLAG ;SET FLAG BIT
8697
8698 034616 042777 000120 154572 RXIN8: BIC #RINTEN+RXENA,@RXCSR ;CLEAR INTAND RX ENABLE
8699
8700 RXINEX: ENDSRV
8701
(3) 034624 L10020:
(2) 034624 000002 RTI

```

8703
8710
8711
8712
8713
8714
8715
8716
8717
8718
8719
8720
8721
8722
8723
8724
8725
8726
8727
8728
8729
8730
8731
8732
8733
8734
8735
8736
8737
8738
8739
8740
8741
8742
8743
8744
8745
8746
8747
8748
8749
8750
8751
8752
8753
8754
8755
8756
8757
8758
8759
8760
8761
8762
8763

```
.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
      (3) 034626
      8745 034626 005737 011454      TST     SYNCC      ;ANY SYNCs TO SEND
      8746 034632 001406              BEQ     TXIN1      ;IF NOT GO TO 1
      8747 034634 013777 011456 154564  MOV     SYNCW,@TDSR ;ELSE SET TSOM AND SYNC WORD
      8748 034642 005337 011454      DEC     SYNCC      ;DEC SYNC COUNT
      8749 034646 001056              BNE     TXINEX     ;IF NOT ZERO EXIT
      8750 034650 042777 001400 154550 TXIN1: BIC     #TEOM,TSOM,@TDSR
      8751 034656 032737 001000 006574  BIT     #PAD,FLAG  ;IS THE PAD BIT SET
      8752 034664 001412              BEQ     TXIN2      ;GO TO 2 IF NOT SET
      8753 034666 012777 000377 154532  MOV     #377,@TDSR ;LOAD FF TO TX DATA REG.
      8754 034674 042777 000100 154522  BIC     #TINTEN,@TXCSR ;CLEAR TX INT ENABLE
      8755 034702 052737 000010 006574  BIS     #QTX,FLAG  ;SET THE TX COMPLETE IN FLAG
      8756 034710 000435              BR      TXINEX     ;AND EXIT
      8757 034712 013702 011450      TXIN2: MOV     MSGPTR,R2 ;LOAD R2 WITH TX ADDR.
      8758 034716 112277 154504      MOVB   (R2)+,@TDSR ;LOAD DATA BYTE
      8759 034722 010237 011450      MOV     R2,MSGPTR ;RESTORE POINTER
      8760 034726 005337 011452      DEC     MSGCC     ;DEC CC
      8761 034732 001024              BNE     TXINEX
      8762 034734 052777 001000 154464  BIS     #TEOM,@TDSR
      8763 034742 032737 000020 006574  BIT     #TXM,FLAG  ;IS THIS THE END OF DATA MSG.
```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-33
DEVICE TRANSMIT INTERRUPT ROUTINE

SEQ 0149

```

8764 034750 001012          BNE    TXIN3      ;IF SO SET THE PAD BIT
8765 034752 052737 000020 006574  BIS    #TXM,FLAG  ;IF NOT MUST BE END OF HEADER
8766 034760 013737 006450 011450  MOV    DVTXA,MSGPTR ;SO SET UP MSGPTR FOR MSG
8767 034766 013737 006452 011452  MOV    DVTCC,MSGCC  ;AND THE CC FOR MSG.
8768 034774 000403          BR     TXINEX
8769 034776 052737 001000 006574  TXIN3: BIS    #PAD,FLAG ;SET THE PAD BIT
8770
8771          035004          TXINEX:
8772          035004          ENDSRV
(3) 035004
(2) 035004 000002
L10021:
RTI

```

8774
8775
8776
8777
8778
8779
8780
8781
8782
8783
8784
8785
8786
8787
8788
8789
8790
8791
8792
8793
8794
8795
8796
8797
8798
8799
8800
8801
8802
8803
8804
8805
8806
8807
8808
8809
8810
8811
8812
8813
8814
8815
8816
8817
8818
8819
8820
8821
8822
8823
8824
8825
8826
8827
8828

```
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.
--

8803 035006 042737 000004 006574 DVIN31: BIC      #QRX,FLAG          ;CLEAR RX COMPLE.
8804                                MOV      #RHDMSG+1,RMSGPT      ;SET UP POINTER
8805                                MOV      HDMC,RMSGCC        ;AND CC
8806                                ;ENABLE RCVR.
8809 035030 052777 000160 154360      BIS      #RINTEN,RXENA,DSITEN,@RXCSR
8810                                ;SET UP TRANSMITTER TO SEND
8813 035036 004737 035136                DVIN32: JSR      PC,CTSSR          ;SET RTS.
8814 035042 042737 001010 006574      BIC      #QTX!PAD,FLAG          ;CLEAR TX COMPT FLAG.
8815 035050 012737 002645 011450      MOV      #HDMSG+1,MSGPTR        ;MOVE THE CURRENT POINTER TO MSGPTR.
8816 035056 013737 002654 011452      MOV      HDMC,MSGCC
8817 035064 012737 000010 011454      MOV      #8,SYNCC              ;SET UP SYNC COUNT
8818 035072 052777 000120 154324      BIS      #TXENA!TINTEN,@TXCSR
8819                                ;NOW WAIT FOR TIME OUT OR TX COMPLETE
8822 035100                DVIN35: BREAK
8823 035100 104422                ;TRAP CSBRK
8823 035102 005737 006640      TST      TIMERS                ;IS IT TIMED OUT
8824 035106 001412                BEQ      DVIN34                ;IF YES EXIT
8825 035110 032737 000010 006574      BIT      #QTX,FLAG            ;IS TX DONE
8826 035116 001770                BEQ      DVIN35                ;IF NOT GO BACK AND CK TIME OUT
8827 035120 004737 035310      JSR      PC,CLRRTS            ;GO CLEAR RTS IF NESC.
8828 035124 032737 000004 006574      BIT      #QRX,FLAG            ;DID WE RX ANYTHING
```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-35
I 12
DEVICE TRANSMIT CONTROL MSG

SEQ 0151

8829 035132 001741
8830 035134 000207
8831

BEQ DVIN32
DVIN34: RTS PC

:IF NOT RETRANSMIT LAST
:RETURN TO CALLER

```

8833 .SBTTL                DEVICE RTS TO CTS DELAY
8834 .**
8835 .FUNCTIONAL DESCRIPTION:
8836 .CTSSR--THIS ROUTINE SETS REQUEST TO SEND TO MODEM
8837 .AND CHECKS FOR CLEAR TO SEND TO COME BACK
8838 .IF CTS DOES NOT COME BACK BEFORE TIMER EXPIRES
8839 .AND ERROR IS REPORTED AND WE TRY AGAIN.
8840 .THE ROUTINE IS SKIPPED IF INTERNAL LOOP IS SET.
8841 .
8842 .
8843 .: OUTPUTS:
8844 .
8845 .SUBORDINATE ROUTINES USED:
8846 .      'LGDVE' - LOG DEVICE ERROR
8847 .CALLING SEQUENCE:
8848 .      JSR      PC,CTSSR
8849 .--
8850 .
8851 .
8852 035136 022737 000001 006564 CTSSR: CMP      #1,MLTYP      ;IS THIS TTL LOOP
8853 .
8854 035144 001460          BEQ      DVTXR9      ;BR IF YES
8855 .
8856 .      ;SET RTS AND WAIT FOR CTS
8857 .
8858 035146 032737 004000 006574 DVTXR3: BIT      #FIRST,FLAG
8859 035154 001014          BNE      CTSS3      ;IF NOT FIRST TIME SKIP DFLY
8860 035156 012737 000000 006530          MOV      #0,TEMP
8861 035164 005237 006530          CTSS4: INC      TEMP
8862 035170          BREAK
8863 035170 104422          (3) TRAP      CSBRK
8864 035172 005737 006530          TST      TEMP
8865 035176 001372          BNE      CTSS4      ;IF NOT ZERO GO BACK
8866 035200 052737 004000 006574          BIS      #FIRST,FLAG ;SET FIRST FLAG.
8867 035206 052777 000004 154202          CTSS3: BIS      #RTS,@RXCSR ;SET REQUEST TO SEND
8868 035214 012737 001750 006634          MOV      #1000.,TIMER1 ;SET UP TIMER
8869 035222          DVTXR2: BREAK
8870 035222 104422          (3) TRAP      CSBRK
8871 035224 032777 020000 154164          BIT      #CTS,@RXCSR ;IS CLEAR TO SEND BACK
8872 035232 001025          BNE      DVTXR1      ;BR. IF CTS IS SET
8873 035234 005737 006634          TST      TIMER1      ;ELSE TEST IF TIME EXPIRED
8874 035240 001370          BNE      DVTXR2      ;BR IF TIME NOT EXPRIED.
8875 .
8876 .      ;SET ERROR FOR NO CTS
8877 .
8878 035242 012737 016520 006534          MOV      #DVEM1,TEMP2
8879 035250 017737 154142 006536          MOV      @RXCSR,TEMP3
8880 035256 017737 154142 006540          MOV      @TXCSR,TEMP4
8881 035264 004737 017652          JSR      PC,LGDVE
8882 035270 005237 006502          INC      ERRCNT
8883 035274          ERRSOFT 6,DVEM1,ERR13
8884 035274 104457          (4) TRAP      CSERSOFT
8885 035276 000006          (5) .WORD      6
8886 035300 016520          (5) .WORD      DVEM1
8887 035302 017326          (5) .WORD      ERR13
8888 035304 000720          BR      DVTXR3      ;THEN TRY TO SET RTS AGAIN

```

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-37
DEVICE RTS TO CTS DELAY

K 12

SEQ 0153

8883 035306
8884 035306 000207

DVTXR1:
DVTXR9: RTS PC ;

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

L 12
MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-38
DEVICE CLEAR REQUEST TO SEND

SEQ 0154

8886
8887
8888
8889
8890
8891
8892
8893
8894
8895 035310
8896
8897 035310 005737 006566
8898 035314 001011
8899 035316 012737 002000 006534
8900 035324 005337 006534
8901 035330 001375
8902
8903 035332 042777 000004 154056
8904 035340 000207
8905

.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:
TST 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

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-39
M 12
DEVICE CLEAR REQUEST TO SEND

SEQ 0155

8918
8919
8920
8921 035342
(3) 035342
(3) 035342 104401
8922

.EVEN
ENDTST

L10017: TRAP C\$ETST

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-40
N 12
DEVICE CLEAR REQUEST TO SEND

SEQ 0156

8929
8930

8932
8933
8934
8935
8936
8937
8938
8939
8940
8941
8942
8943
8944
8945
8955
8956
8957
8958
8959
8960
8971
8972
8973
8974
8975
8976
8977
8978
8979
8980
8981
8982
8983
8984
8985
8986
8994
8995

035344
(3) 035344 000016
(3) 035346
035346
035346 000130
(4) 035346 000130
(4) 035350 035402
(4) 035352 000001
035354
(4) 035354 001031
(4) 035356 035433
(4) 035360 160000
(4) 035362 177776
035364
(4) 035364 002031
(4) 035366 035461
(4) 035370 000300
(4) 035372 000776
035374
(4) 035374 006130
(4) 035376 035514
(4) 035400 000001
035402
(2)
(3) 035402

.SBTTL HARDWARE PARAMETER CODING SECTION

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

BGNHRD

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

.SBTTL DEVICE INDEPENDENT SECTION

GPRML DPLX,0,1,YES

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

.SBTTL DEVICE DEPENDENT SECTION

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

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

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

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

GPRML RNODM,14,1,YES

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

END IRD

.EVEN
L10022:

.NLIST BEX

:DEVICE INDEPENDENT QUESTIONS

DPLX: .ASCIZ /FULL DUPLEX OPERATION : /

:DEVICE DEPENDENT QUESTION

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 13-42
DEVICE DEPENDENT SECTION

SEQ 0158

8996
8997 035433 104 053105 041511
8998 035461 111 052116 051105
8999 035514 042522 047515 042524
9000
9001
9002
9009

CSRADR: .ASCIZ /DEVICE CSR ADDRESS : /
VECTOR: .ASCIZ /INTERRUPT VECTOR ADDRESS: /
RNODM: .ASCIZ /REMOTE NODE 'ITEP':/
.LIST BEX
.EVEN

ACT =	000003	4295#	7121	7489	7565	7803			
ACTATV	030006	7283	7489#						
ACTBCR	027612	7301	7454#						
ACTCHK	030222	7263	7531#						
ACTCLB	027134	7354	7368#						
ACTCLP	030334	7297	7559#						
ACTCLR	026624	7261	7315#						
ACTCOP	027432	7271	7422#						
ACTCRC	030236	7292	7537#						
ACTCSE	026760	7266	7340#						
ACTCST	027052	7267	7356#						
ACTDLL	030054	7287	7503#						
ACTDME	027360	7303	7402	7405#					
ACTDMQ	027352	7304	7404#						
ACTDMS	027330	7302	7399#						
ACTDMX	027366	7406#							
ACTECH	030132	7291	7520#						
ACTEQO	027554	7275	7443#						
ACTEXT	026710	7307	7328#						
ACTHLP	026644	7265	7321#						
ACTLIS	030044	7286	7500#						
ACTLLP	030344	7298	7561#						
ACTLPX	030362	7556	7558	7560	7562	7565#			
ACTLXX	030424	7529	7550	7553	7566	7570#			
ACTMEX	030000	7436	7452	7469	7474	7480	7483	7485#	
ACTME1	027734	7458	7460	7462	7464	7466	7473#		
ACTMOP	030314	7295	7555#						
ACTMOS	030244	7306	7540#						
ACTMS0	027634	7276	7457#						
ACTMS1	027642	7277	7459#						
ACTMS2	027652	7278	7461#						
ACTMS3	027662	7279	7463#						
ACTMS4	027672	7280	7465#						
ACTMS5	027702	7281	7467#						
ACTMS6	027720	7282	7470#						
ACTM2X	030102	7490	7498	7501	7504	7507	7511#		
ACTNO	030122	7290	7517#						
ACTNUF	026614	7300	7312#						
ACTNUL	026622	7260	7313#						
ACTNUM	027442	7272	7425#						
ACTOPM	027534	7273	7438#						
ACTPAS	030016	7284	7492#						
ACTPRO	030252	7293	7543#						
ACTPRT	026720	7305	7331#						
ACTQFG	030256	7532	7535	7538	7541	7545#			
ACTREC	030036	7285	7497#						
ACTREX	020704	5950	5965#						
ACTRHL	020640	5949	5957#						
ACTRLG	020714	5951	5969#						
ACTRLP	030354	7299	7563#						
ACTRNL	020636	5948	5954#						
ACTRPS	030304	7294	7552#						
ACTRUN	026734	7264	7335#						
ACTSEX	027744	7308	7477#						
ACTSHO	026634	7262	7318#						
ACTSHW	027174	7346	7360	7379#	7391				

ACTSTE	027374	7268	7412#							
ACTSTS	030230	7274	7534#							
ACTSTT	027404	7269	7415#							
ACTSTX	027412	7413	7416#							
ACTSZE	027422	7270	7419#							
ACT'AL	030074	7289	7509#							
ACTTLP	030324	7296	7557#							
ACTTRA	030064	7288	7506#							
ADDCC	022136	6190#	7203	7233						
ADDCC1	022232	6192	6202#							
ADR =	000020 G	4278#								
ALCK	031032	4760	7712#							
ALCK1	031144	7768	7777#	7823						
ALCK2	031210	7778	7787#	7855						
ALCK2A	031430	7820	7825#							
ALCK3	031454	7790	7827	7830#						
ALCK3A	031546	7837	7843#							
ALCK3B	031560	7844	7846#							
ALCK3C	031554	7841	7845#	7852						
ALCK4	031602	7831	7850#							
ALCK4A	031610	7849	7853#							
ALCK4B	031622	7854	7856#							
ALCK5	031074	7767#	7845							
ALLTR	031074	7641	7665	7690	7717	7766#				
ASSEMB=	000010	4030								
ATVMOD=	000027	4432#	4905							
BAD	006551	4729#	5504	7928*						
BADCHR=	000051	4450#	5020							
BASM1	015173	5336#	6150							
BASM2	015164	5335#	6156							
BASM3	015155	5334#	6154							
BCC =	000400	4386#	8418	8421	8653	8667	8690			
BCCW	011464	5114#	8669							
BDCLK	013712	5297#	6826							
BIT0 -	000001 G	4278#	4315	4364	4504					
BIT00 -	000001 G	4278#								
BIT01 =	000002 G	4278#								
BIT02 =	000004 G	4278#								
BIT03 =	000010 G	4278#								
BIT04 =	000020 G	4278#								
BIT05 -	000040 G	4278#								
BIT06 =	000100 G	4278#								
BIT07 =	000200 G	4278#								
BIT08 =	000400 G	4278#								
BIT09 =	001000 G	4278#								
BIT1 -	000002 G	4278#	4316	4365	4497	4505				
BIT10 =	002000 G	4278#	4390	4502						
BIT11 -	004000 G	4278#	4392	4503						
BIT12 =	010000 G	4278#	4481							
BIT13 -	020000 G	4278#	4479							
BIT14 -	040000 G	4278#	4483							
BIT15 =	100000 G	4278#	4512	4513						
BIT2 -	000004 G	4278#	4317	4366	4482	4506				
BIT3 -	000010 G	4278#	4318	4367	4507					
BIT4 =	000020 G	4278#	4319	4382	4498	4508				
BIT5 -	000040 G	4278#	4320	4384	4484	4485	4499	8374	8375	8377

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 15-8
CROSS REFERENCE TAB F -- USER SYMBOLS

SEQ 0168

FSDU = 000016	4030#	6980	7007											
FSEND = 000041	4030#	4051	5505	5509	5513	5517	5536	5545	5547	5731	6718	6896	6912	
	6932	6955	6971	6991	7007	7028	7044	7070	7137	7176	8701	8772	8921	
	8979	9060												
F\$HARD= 000004	4030#	8944	8979											
F\$HW = 000013	4030#	4154	4200											
F\$INIT= 000006	4030#	6756	6912											
F\$JMP = 000050	4030#	5547	6896	6955	6991	7028	7137	7176						
F\$MOD = 000000	4030#	4051	9060											
F\$MSG = 000011	4030#	5503	5505	5507	5509	5511	5513	5515	5517	5534	5536	5543	5545	
F\$PROT= 000021	4030#	6727	6733											
F\$PWR = 000017	4030#													
F\$RPT = 000012	4030#	6686	6718											
F\$SEG = 000003	4030#													
F\$SOFT= 000005	4030#													
F\$SRV = 000010	4030#	5707	5731	8611	8701	8744	8772							
F\$SUB = 000002	4030#													
F\$SW = 000014	4030#													
F\$TEST= 000001	4030#	7070	8921											
GETCL 025460	7134	7143#	7154	7158	7161	7163	7165	7172	7179	7181	7188	7197	7211	
	7217	7227	7241	7583										
GETTRM 024530	6797	6839#	6844											
GETRCL 020330	5901#	5914	5919	5923										
GI INC 034260	8627	8631#												
GLMSG 016363	5398#	8632												
GOOD 006550	4728#	5504	7927*											
GTRA2 025100	7083#													
GTRA3 025142	7086	7091#												
GTRA4 025150	7096#													
GTRA5 025374	7092	7127#	8013	8038										
GTRRX 030534	7594#													
GTRX2 030636	7610#	8012	8093	8140										
GTR9 030432	7139	7168	7576#											
GTXRXB 025100	7082#													
G\$CNTO= 000200	4030#													
G\$DELM= 000372	4030#													
G\$DISP= 000003	4030#													
G\$EXCP= 000400	4030#													
G\$HILI= 000002	4030#													
G\$LOLI= 000001	4030#													
G\$NO = 000000	4030#	5905	7145	8066										
G\$OFFS= 000400	4030#	5905	6823	7145	8066	8958	8975	8976	8977					
G\$OF SI= 000376	4030#	5905	6823	7145	8066	8958	8975	8976	8977					
G\$PRMA= 000001	4030#	8975	8976											
G\$PRMD= 000002	4030#	5905	6823	7145	8066									
G\$PRML= 000000	4030#	8958	8977											
G\$RADA= 000140	4030#	5905	7145	8066										
G\$RADB= 000000	4030#													
G\$RADD= 000040	4030#	6823												
G\$RADL= 000120	4030#	8958	8977											
G\$RADO= 000020	4030#	8975	8976											
G\$XFER= 000004	4030#													
G\$YES = 000010	4030#	6823	8958	8975	8976	8977								
HDMC 002654	4602#	8306	8430	8515	8806	8816								
HDMCC 002646	4596#	8292*	8304*	8316*	8330	8514*								
HDMSG 002644	4594#	8288*	8305	8512*	8513	8815								

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 15-10
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0170

LCLKEN=	000100	4309#	6806						
LGDVE	017652	5788#	8240	8273	8331	8530	8661	8682	8879
LIS =	000006	4298#	7500	8092					
LISCK	032462	4763	8115#						
LISCKA	032510	8117#	8136	8138					
LISMOD=	000032	4435#	4911						
LISP	013555	5269#	8116						
LMDLOP=	000046	4447#	5051						
LNCNT	006474	4702#	5855	5857*	5861*	7600*			
LOE =	040000	G 4278#							
LOGCMD	017772	5810#	7940						
LOGCML	017754	5806#	7917						
LOGCMP	017736	5802#	7911						
LOGDVI	017670	5793#	7607						
LOGEOP	020010	5814#	8006						
LOGEX	020300	5844	5881#						
LOGMSC	020026	5833#	8633						
LOGRXC	017642	5785#	7793	8131					
LOGRXQ	017624	5780#	7776	8125					
LOGS1	020044	5773	5778	5783	5787	5839#			
LOGS2	020272	5875	5879#						
LOGS3	020076	5791	5800	5805	5809	5813	5817	5836	5848#
LOGS4	020152	5856	5860#						
LOGS5	020176	5850	5852	5863#					
LOGTXC	017606	5775#	7834	8086					
LOGTXQ	017570	5770#	7785	8078					
LOGUNT	006552	4734#	6837*	6839*	6840	6843			
LOOPS	003114	4653#	6388						
LOT =	000010	G 4278#							
LP0	013437	4653	5242#	6387					
LP00	013440	5243#	6384						
LP1	013447	4654	5244#						
LP2	013460	4655	5245#						
LP3	013466	4656	5246#						
LP4	013501	4657	5247#						
L\$ACP	002110	G 4116#							
L\$APT	002036	G 4116#							
L\$AU	025064	G 4116	7017#						
L\$AUT	002070	G 4116#							
L\$AUTO	025032	G 4116	6923#						
L\$CCP	002106	G 4116#							
L\$CLEA	025034	G 4116	6941#						
L\$CO	002032	G 4116#							
L\$DEPO	002011	G 4116#							
L\$DESC	011504	G 4116	5176#						
L\$DESP	002076	G 4116#							
L\$DEVP	002060	G 4116#							
L\$DISP	002124	G 4116	4136#						
L\$DLY	002116	G 4116#							
L\$DTP	002040	G 4116#							
L\$DTYP	002034	G 4116#							
L\$DU	025056	G 4116	6980#						
L\$DUT	002072	G 4116#							
L\$DVTY	011474	G 4116	5165#						
L\$EF	002052	G 4116#							
L\$ENVI	002044	G 4116#							

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

E 1.
MACY11 30A(1052) 16-JUN-81 10:00 PAGE 15-13
CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0173

NOD107	010702	4990#
NOD11	007772	4887#
NOD110	010716	4991#
NOD111	010722	4994#
NOD112	010726	4997#
NOD113	010742	4998#
NOD114	010746	4999#
NOD115	010764	5000#
NOD116	010770	5001#
NOD117	011004	5002#
NOD12	010004	4888#
NOD120	011010	5003#
NOD121	011024	5004#
NOD122	011030	5005#
NOD123	011044	5006#
NOD124	011050	5007#
NOD125	011064	5008#
NOD126	011070	5009#
NOD127	011104	5010#
NOD13	010010	4889#
NOD130	011110	5011#
NOD131	011130	5012#
NOD132	011134	5014#
NOD133	011140	5015#
NOD134	011144	5016#
NOD135	011150	5017#
NOD136	011154	5018#
NOD137	011160	5019#
NOD14	010024	4890#
NOD140	011164	5020#
NOD141	011166	5023#
NOD142	011172	5024#
NOD143	011176	5025#
NOD144	011212	5026#
NOD145	011216	5027#
NOD146	011232	5028#
NOD147	011236	5031#
NOD15	010030	4891#
NOD150	011242	5032#
NOD151	011246	5033#
NOD152	011252	5036#
NOD153	011256	5047#
NOD154	011300	5048#
NOD155	011304	5049#
NOD156	011320	5050#
NOD157	011324	5051#
NOD16	010044	4892#
NOD160	011346	5052#
NOD161	011352	5053#
NOD162	011374	5054#
NOD163	011400	5057#
NOD164	011404	5058#
NOD165	011410	5059#
NOD166	011414	5064#
NOD167	020522	5930#
NOD17	010050	4893#

NOD170	020526	5931#
NOD171	020532	5932#
NOD172	020534	5933#
NOD173	020550	5934#
NOD174	020552	5935#
NOD175	020566	5936#
NOD176	020570	5937#
NOD177	020602	5938#
NOD2	007716	4880#
NOD20	010054	4894#
NOD200	020604	5939#
NOD201	020606	5940#
NOD21	010066	4895#
NOD22	010072	4896#
NOD23	010104	4897#
NOD24	010110	4898#
NOD25	010112	4902#
NOD26	010116	4903#
NOD27	010132	4904#
NOD3	007720	4881#
NOD30	010136	4905#
NOD31	010154	4906#
NOD32	010160	4907#
NOD33	010176	4908#
NOD34	010202	4909#
NOD35	010220	4910#
NOD36	010224	4911#
NOD37	010242	4912#
NOD4	007734	4882#
NOD40	010246	4913#
NOD41	010272	4914#
NOD42	010276	4915#
NOD43	010302	4916#
NOD44	010320	4917#
NOD45	010324	4918#
NOD46	010336	4919#
NOD47	010342	4923#
NOD5	007736	4883#
NOD50	010346	4924#
NOD51	010370	4925#
NOD52	010372	4926#
NOD53	010416	4927#
NOD54	010420	4932#
NOD55	010424	4933#
NOD56	010442	4934#
NOD57	010446	4935#
NOD6	007752	4884#
NOD60	010466	4936#
NOD61	010472	4939#
NOD62	010476	4940#
NOD63	010502	4941#
NOD64	010506	4942#
NOD65	010512	4943#
NOD66	010516	4944#
NOD67	010522	4945#
NOD7	007754	4885#

SVCSUB= 000001	8958	8975	8976	8977	8979	9059								
SVCTAG= 000001	4030#	4038#												
	4030#	4040#	4200	5505	5509	5513	5517	5536	5545	5731	5905	6718	6823	
	6912	6932	6971	7007	7044	7145	8066	8701	8772	8921	8979			
SVCTST= 000001	4030#	4037#	7070											
SYN = 000226	4514#													
SYNCC 011454	5110#	8307*	8516*	8745	8748*	8817*								
SYNCW 011456	5111#	7601*	7606*	8747										
SLSYM= 010000	4030#	4200#	5505#	5509#	5513#	5517#	5536#	5545#	5731#	5905#	6718#	6823#	6912#	
	6932#	6971#	7007#	7044#	7145#	8066#	8701#	8772#	8921#	8979#				
S1 024250	6793	6795#												
S2 024324	6804	6809#												
S3 024374	6810	6818#												
S4 024442	6819	6826#												
TABEX 014017	5299#	7196	7226											
TAL = 000005	4297#	7509	8139											
TALCK 032242	4762	8060#	8088	8090										
TALMOD= 000035	4438#	4918												
TBMT = 000004	4506#													
TCURAD 006462	4695#	7107*	7110	7202	7207*	7366*								
TDSR 011426	5087#	6870*	6871*	8747*	8750*	8753*	8758*	8762*						
TEMP 006530	4720#	5772*	5777*	5782*	5786*	5790*	5795*	5804*	5808*	5812*	5816*	5835*	5849	
	5865	5866*	5867*	5868	6153*	6154	6198*	6199*	6200	6239*	6240*	6246	6381*	
	6390	6393*	6396*	6428	7387*	7388	7439*	7440*	7444	7447	7808*	7809*	7813	
	8860*	8861*	8863											
TEMP1 006532	4721#	5771*	5776*	5781*	5789*	5794*	5803*	5807*	5811*	5815*	5834*	5862	6388*	
	6390	6397*	6400*	6428										
TEMP2 006534	4722#	5796*	5797*	5871	6242*	6244*	6248	6389*	6390	6401*	6404*	6428	7770*	
	7780*	7791*	7797	7811*	7812	7832*	7901*	8004*	8075*	8084*	8118*	8129*	8237*	
	8270*	8328*	8527*	8629*	8632*	8660*	8679*	8876*	8899*	8900*				
TEMP3 006536	4723#	5512	5535	5544	5798*	5872	6384*	6387*	6390	7772*	7782*	7792*	7798	
	7809	7810	7811	7833*	7903*	7913	8005*	8076*	8085*	8120*	8130*	8238*	8271*	
	8329*	8528*	8624*	8626	8658*	8680*	8877*							
TEMP4 006540	4724#	5508	5535	5544	5799*	5846*	5873	7910*	7919*	7924*	7925	7936	8003*	
	8239*	8272*	8330*	8529*	8625*	8626	8659*	8681*	8878*					
TEMP5 006542	4725#	6422*	6425*	6428										
TEOM = 001000	4511#	8750	8762											
TERR = 100000	4512#													
TIMERS 006640	4779#	5725	5729*	8290*	8294	8312	8318	8519*	8522	8823				
TIMER1 006634	4777#	5719	5721*	7084*	7085	8264*	8265	8867*	8871					
TIMER2 006636	4778#	5722	5724*											
TIMMIN 006626	4773#	5716*	5870	6828*										
TIMSEC 006630	4774#	5713*	5714	5717*	5869	6829*								
TIMTCK 006632	4775#	5710*	5712*	5727	5867	6830*								
TINTEN= 000100	4509#	8308	8517	8754	8818									
TM = 000040	4485#	4802	8621	8623										
TOINOT 034050	8523	8535#												
TOTCC 006524	4718#	6190*	6191	6197*	6199	6201*	7096*	7184*	7185	7206	7213*	7214	7238	
TRA = 000001	4293#	7506												
TRAMOD= 000034	4437#	4916												
TRVACT 023206	6484	6495#	6511	6516	6521	6524	6544	6605	6623	6644	6668			
TRVALN 024000	6473	6627#												
TRVALP 023734	6472	6613#												
TRVBIF 023312	6469	6524#												
TRVBR 023302	6468	6521#												
TRVBRC 023226	6482	6502#	6522	6527	6546	6610	6625	6646	6672					

MSEXIT	2014#	4030#	5547#	6896#	6955#	6991#	7028#	7137#	7176#						
MSEXSE	2022#	4030#	5547#	6896#	6955#	6991#	7028#	7137#	7176#						
MSEX*J	2018#	4030#	5547#	6896#	6955#	6991#	7028#	7137#	7176#						
MSGEN	2038#	4030#	4116#	4136#	4154#	4200#	5165#	5176#	5503#	5505#	5507#	5509#	5511#	5513#	5515#
	5517#	5534#	5536#	5543#	5545#	5707#	5731#	5905#	6686#	6718#	6727#	6756#	6823#	6912#	6923#
	6932#	6941#	6971#	6980#	7007#	7017#	7044#	7070#	7145#	8066#	8611#	8701#	8744#	8772#	8921#
	8944#	8979#	9059#												
MSGENB	1938#	4030#	5905#	6823#	7145#	8066#									
MSGETS	2035#	4030#	4200#	5505#	5509#	5513#	5517#	5536#	5545#	5731#	6718#	6733#	6912#	6932#	6971#
	7007#	7044#	8701#	8772#	8921#	8979#	9060#								
MSGETT	1877#	4030#	5547#	6896#	6955#	6991#	7028#	7137#	7176#						
MSGNGB	1902#	4030#	4051#	4116#	4136#	4154#	5165#	5176#	5503#	5507#	5511#	5515#	5534#	5543#	5707#
	6686#	6727#	6756#	6923#	6941#	6980#	7017#	8611#	8744#	8944#	9059#				
MSGNIN	2049#	4030#	4116#	4136#	4154#	5165#	5176#	5504#	5505#	5508#	5509#	5512#	5513#	5516#	5517#
	5535#	5536#	5544#	5545#	5547#	5731#	5859#	5862#	5899#	5905#	5913#	5918#	5958#	5996#	6014#
	6019#	6025#	6032#	6033#	6049#	6057#	6064#	6083#	6086#	6101#	6116#	6150#	6154#	6156#	6196#
	6390#	6428#	6570#	6608#	6718#	6785#	6788#	6789#	6790#	6791#	6792#	6793#	6795#	6796#	6803#
	6804#	6809#	6810#	6818#	6819#	6823#	6826#	6843#	6844#	6880#	6892#	6893#	6895#	6896#	6912#
	6932#	6952#	6953#	6955#	6971#	6991#	7007#	7028#	7044#	7089#	7126#	7133#	7134#	7137#	7145#
	7153#	7157#	7176#	7187#	7196#	7216#	7226#	7322#	7388#	7429#	7454#	7525#	7569#	7582#	7916#
	7930#	7939#	7979#	8037#	8066#	8116#	8134#	8232#	8242#	8275#	8277#	8311#	8333#	8521#	8532#
	8663#	8684#	8701#	8772#	8822#	8862#	8868#	8881#	8921#	8944#	8958#	8975#	8976#	8977#	8979#
	9059#														
MSGNLS	1913#	4030#	5905#	6823#	7145#	8066#									
MSGNSU	1898#	4030#													
MSGNTA	1890#	4030#	4200#	5505#	5509#	5513#	5517#	5536#	5545#	5731#	6718#	6912#	6932#	6971#	7007#
	7044#	8701#	8772#	8921#	8979#										
MSGNTE	1894#	4030#	7070#												
MSHAPT	1739#	4030#	4116#												
MSHNAP	1824#	4030#	4116#												
MSINCR	2026#	4030#	4051#	4154#	5503#	5504#	5505#	5507#	5508#	5509#	5511#	5512#	5513#	5515#	5516#
	5517#	5534#	5535#	5536#	5543#	5544#	5545#	5707#	5859#	5862#	5899#	5905#	5913#	5918#	5958#
	5996#	6014#	6019#	6025#	6032#	6033#	6049#	6057#	6064#	6083#	6086#	6101#	6116#	6150#	6154#
	6156#	6196#	6390#	6428#	6570#	6608#	6686#	6718#	6727#	6756#	6788#	6790#	6792#	6795#	6796#
	6803#	6809#	6818#	6823#	6826#	6843#	6880#	6892#	6893#	6895#	6896#	6912#	6923#	6932#	6941#
	6952#	6953#	6955#	6971#	6980#	7007#	7017#	7044#	7070#	7089#	7126#	7133#	7137#	7145#	7153#
	7157#	7176#	7187#	7196#	7216#	7226#	7322#	7388#	7429#	7454#	7525#	7569#	7582#	7916#	7930#
	7939#	7979#	8037#	8066#	8116#	8134#	8232#	8242#	8275#	8277#	8311#	8333#	8521#	8532#	8611#
	8663#	8684#	8744#	8822#	8862#	8868#	8881#	8921#	8944#						
MSIOSE	1700#	4030#													
MSLDRO	1942#	4030#	6788#	6790#	6792#	6795#	6803#	6809#	6843#	6895#	6952#				
MSMASK	1671#	4030#													
MSMCHI	4#	4030#													
MSMCLO	1624#	4030#													
MSMSK1	1677#	4030#													
MSPOP	1881#	4030#	4200#	5505#	5509#	5513#	5517#	5536#	5545#	5731#	6718#	6733#	6912#	6932#	6971#
	7007#	7044#	8701#	8772#	8921#	8979#	9060#								
MSPRIN	1636#	4030#	5504#	5508#	5512#	5516#	5535#	5544#	5859#	5862#	5899#	5913#	5918#	5958#	5996#
	6014#	6019#	6025#	6032#	6033#	6049#	6057#	6064#	6083#	6086#	6101#	6116#	6150#	6154#	6156#
	6196#	6390#	6428#	6570#	6608#	6826#	7089#	7126#	7153#	7157#	7187#	7196#	7216#	7226#	7322#
	7388#	7429#	7454#	7525#	7569#	7582#	8037#	8116#	8134#						
MSPUSH	1631#	4030#	4051#	4154#	5503#	5507#	5511#	5515#	5534#	5543#	5707#	6686#	6727#	6756#	6923#
	6941#	6980#	7017#	7070#	8611#	8744#	8944#								
MSPUT	1972#	4030#	5504#	5508#	5512#	5516#	5535#	5544#	5859#	5862#	5899#	5913#	5918#	5958#	5996#
	6014#	6019#	6025#	6032#	6033#	6049#	6057#	6064#	6083#	6086#	6101#	6116#	6150#	6154#	6156#
	6196#	6390#	6428#	6570#	6608#	6826#	6880#	6892#	6893#	7089#	7126#	7153#	7157#	7187#	7196#

CVCLHB DPV-11 DATA COMM. LINK TEST
CVCLHB.P11 15-JUN-81 15:48

MACY11 30A(1052) 16-JUN-81 10:00 PAGE 16-4
CROSS REFERFNC TABLE -- MACRO NAMES

SEQ 0188

PRINTX	1359#	4030#								
READBU	1399#	4030#	6818							
REDEF	1403#	4030#	6788	6790	6792	6795				
RFLAGS	1408#	4030#								
SETPRI	1413#	4030#	6895	6952						
SETVEC	1418#	4030#	6880	6892	6893					
SLASH	1424#	4030#								
STARS	1438#	4030#								
SVC	1452#	4029#	4030							
XFER	1612#	4030#	5547#	6896#	6955#	6991#	7028#	7137#	7176#	
XFERF	1616#	4030#								
XFERT	1620#	4030#								

. ABS. 035624 000

ERRORS DETECTED: 0

CVCLHB/I, CVCLHB, SEQ/DOC/CRF=SVC34R.MLB, CVCLHB.P11
RUN-TIME: 22 28 3 SECONDS
RUN-TIME RATIO: 71/54=1.3
CORE USED: 19K (37 PAGES)

DOCUMENT PAGES: 188