

DZV11

CABLE/ECHO TST
CVDZCBO

AH-A942B-MC
FICHE 1 OF 1

JUL 1982
COPYRIGHT © 77-82
MADE IN USA



The main body of the document is a large, dark blue area containing a grid of small, illegible data tables. The grid is organized into approximately 10 columns and 15 rows. Each cell in the grid appears to contain a small table or set of data, but the text is too faint and blurry to be transcribed. The overall appearance is that of a microfiche or a high-density data storage format.

.REM 8

IDENTIFICATION

PRODUCT CODE: AC-A941B-MC
PRODUCT NAME: CVDZCBO DZV11 CABLE/ECHO TST
DATE RELEASED: 17-FEB-82
MAINTAINER: DIAGNOSTIC ENGINEERING

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.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1977, 1982 DIGITAL EQUIPMENT CORPORATION

1. ABSTRACT

THE FUNCTION OF THE DZV11 DIAGNOSTICS IS TO VERIFY THE OPTION OPERATES ACCORDING TO SPECIFICATIONS. THE DIAGNOSTICS ALSO VERIFY THAT THE DZV11 OPERATES IN ITS ENVIRONMENT SUCH AS THE SYSTEM IN WHICH IT IS INSTALLED.

CURRENTLY THERE ARE THREE STANDALONE DIAGNOSTICS (DVDZA, DVDZB, AND DVDZC) ONE SYSTEM MODULE FOR DEC X/11 (DZBA), AND AN OVERLAY FOR ITEX (DVDZD).

DVDZA TOGETHER WITH DVDZB WILL TEST ALL LOGICAL FUNCTIONS OF THE DZV11 INTERFACE MODULE.

DVDZL IS DESIGNED AS A NON-CHAINABLE STANDALONE DIAGNOSTIC PROVIDING THE OPERATOR WITH DIRECT CONTROL OVER THE TESTING OF ALL DZV11 EIA CABLES.

```

*****
*
* NOTE: THIS DIAGNOSTIC HAS BEEN MODIFIED TO RUN IN KXT11 (SBC 11/21)
* BASED SYSTEMS. THE PROGRAM WILL AUTOMATICALLY ADJUST ITSELF TO RUN
* IN THE APPROPRIATE ENVIRONMENT AS FOLLOWS:
*
*
*          LSI-11, 11/2, AND 11/23          SBC 11/21
*          -----
* CSR RANGE:          160010 TO 163770          174000 TO 177770
* VECTOR RANGE:          300 TO 770          300 TO 370
* AUTO-SIZING FOR...
* ...CSR AND VECTOR:   ENABLED          DISABLED
*
*
*****

```

2. REQUIREMENTS

2.1 EQUIPMENT

AN LSI11 CPU WITH MINIMUM 4K OF MEMORY.
 ASR 33 (OR EQUIVALENT FOR CONSOLE)
 ASR 33 (OR EQUIVALENT) TO RUN DZV11 ECHO TEST
 DZV11 INTERFACE MODULE
 H325 CABLE TURNAROUND CONNECTOR.

2.2 STORAGE

PROGRAM WILL USE ALL 4K OF MEMORY EXCEPT WHERE ABL AND BOOTSTRAP LOADER RESIDE. LOCATION 1500 THRU 1740 ARE ESPECIALLY TO BE NOTED AND TO BE UNTOUCHED BY THE OPERATOR IF THE PARAMETERS HAVE BEEN ALREADY BUILT BY RUNNING EITHER THE DVDZA OR DVDZB DIAGNOSTICS. LOADING THIS DIAGNOSTIC WILL PRESERVE THESE LOCATIONS.

3. LOADING PROCEEDURE

3.1 METHOD

ALL PROGRAMS ARE IN ABSOLUTE FORMAT AND ARE LOADED USING THE ABSOLJTE LOADER. NOTE: IF THE DIAGNOSTICS ARE ON A MEDIA SUCH AS DISK ,MAGTAPE,DECTAPE, OR CASSETTE; FOLLOW INSTRUCTIONS FOR THE MONITOR WHICH HAS BEEN PROVIDED ON THAT SPECIFIC MEDIA.

ABSOLUTE LOADER STARTING ADDRESS *500

MEMORY * SIZE

4K	17
8K	37
12K	57
16K	77
20K	117
24K	137
28K	157

3.1.1 STARTING THE PROCESSOR AT THE ABSOLUTE LOADER STARTING ADDRESS WILL LOAD THE DIAGNOSTIC INTO MEMORY.

4. STARTING PROCEEDURE

- A. SET THE SWR TO ALLOW THE DESIRED PROGRAM OPTIONS TO FUNCTION.
NOTE: LOC. 000176 IS USED AS A SOFTWARE SWITCH REGISTER IN ALL OF THE DZV11 DIAGNOSTICS. (SEE SEC. 4.1)
- B. START THE DIAGNOSTIC AT LOC. 200(8). THE PROGRAM WILL TYPE MAINDEC AND PROGRAM NAMES (IF THIS WAS THE FIRST START UP OF THE PROGRAM).
- C. THE PROGRAM WILL THEN ASK FOR THE DEVICE ADDRESS, THE VECTOR AND THE LINE NO. OF THE DZV11 TO BE TESTED. TYPE THESE VALJES ON THE CONSOLE TERMINAL FOLLOWED BY A <CR>. THE PROGRAM WILL THEN ASK FOR WHICH TEST IS DESIRED, ECHO OR CABLE. TYPE EITHER E OR C AND A <CR>. THE DIAGNOSTIC WILL TYPE OUT THE NAME OF THE TEST THAT IS NOW RUNNING (SEE SEC. 5.1).

4.1 CONTROL SWITCH SETTINGS

NOTE: THIS PROGRAM UTILIZES A SOFTWARE SWITCH REGISTER WHICH MAY BE MODIFIED BY CHANGING LOC. 176 OR BY TYPING CONTROL 'G' (^G) ON THE CONSOLE TERMINAL WHILE THE PROGRAM IS RUNNING.

SW 15	SET: HALT ON ERROR
SW 14	SET: RESERVED
SW 13	SET: INHIBIT ERROR PRINT OUT
SW 12	SET: INHIBIT **ALL** TYPE OUT/BELL ON ERROR.
SW 11	SET: RESERVED
SW 10	SET: GO TO END OF PASS AFTER AN ERROR
SW 09	SET: LOOP WITH CURRENT DATA (SEE SEC. 4.1.1)
SW 08	SET: RESTART TEST AFTER AN ERROR
SW 07	SET: RESERVED
SW 06	SET: RESERVED
SW 05	SET: RESERVED
SW 04	SET: RESERVED
SW 03	SET: RESERVED
SW 02	SET: RESERVED
SW 01	SET: RESERVED
SW 00	SET: RESERVED

4.1.1 SWITCH REGISTER RESTRICTIONS

SW 09 LOOP ON CURRENT DATA: THIS SWITCH IS ONLY USED IN THE CABLE TEST TO LOCK ON TESTING IF SETTING THE DTR BIT FOR THE DESIRED LINE IN THE TRANSMIT CONTROL REGISTER OF THE DZV11 WILL CAUSE THE CO AND RING BITS TO SET FOR THAT LINE IN THE MODEM STATUS REGISTER. THIS SWITCH IS DESIGNED TO PROVIDE AN AID FOR A TRAINED TROUBLE-SHOOTER TO SAMPLE VARIOUS SIGNALS ON THE MODULE AND IS NOT MEANT TO BE USED AS A GENERAL USER CONTROL SWITCH.

4.1.2 SWITCH REGISTER PRIORITIES

ERROR SWITCHES

1. SW 12 DELETE PRINT OUT/BELL ON ERROR.
2. SW 13 DELETE ERROR PRINTOUT.
3. SW 15 HALT ON THE ERROR.
4. SW 08 RESTART THE TEST AFTER AN ERROR
5. SW 10 GO TO THE END OF PASS AFTER AN ERROR

SCOPE SWITCHES

1. SW 09 (IF ENABLED BY 'SCOPI'). IF AN '*' IS PRINTED IN FRONT OF THE TEST NO. ON AN ERROR REPORT THEN SW09 IS INCORPORATED IN THAT TEST. THIS SWITCH PROVIDES THE OPERATOR WITH THE ABILITY TO LOCK ON A SPECIFIC TEST OPERATION.
IF THE PROGRAM USER IS TECHNICALLY TRAINED TO ELECTRONICALLY ISOLATE SIGNAL PROBLEMS ON THE DZV11 MODULE, THIS SWITCH MIGHT PROVE TO BE A USEFUL AID.
PRESENTLY THIS SWITCH IS ONLY USED IN THIS DIAGNOSTIC FOR THE CABLE TEST TO LOCK ON CHECKING THAT IF DTR IS SET FOR AN ACTIVE LINE THE CO AND RING WILL BECOME SET FOR THAT LINE.

4.2 STARTING ADDRESS

SA 200 - THE STARTING ADDRESS FOR ANY DZV11 DIAGNOSTIC IS LOC. 200

NOTE: THIS DIAGNOSTIC IS NOT DESIGNED TO RUN IN AN AUTOMATIC CHAIN MODE BECAUSE OF THE OPERATOR INTERVENTION REQUIRED TO RUN IT.

5. OPERATING PROCEEDURE

WHEN THE PROGRAM IS INITIALLY STARTED, MESSAGES AS DESCRIBED IN SECTION FOUR WILL BE PRINTED AND THE DIAGNOSTIC WILL BEGIN RUNNING.

5.1 HOW TO RUN THE "CABLE/ECHO" TESTS.

NORMAL STARTING PROCEEDURE FOR THE FIRST TIME WOULD BE:
LOAD THE DIAGNOSTIC, SET THE SWR AT LOC. 176 TO WHATEVER SETTINGS ARE
DESIRED, THEN START THE PROGRAM AT LOC. 200.
THE PROGRAM WILL PRINT OUT ON THE CONSOLE TERMINAL:

'VECTOR ADDRESS-''

YOU TYPE A VECTOR FOLLOWED BY A <CR>.

'CONTROL REGISTER ADDRESS-''

YOU TYPE IN THE DZVCSR ADDRESS UNDER TEST FOLLOWED BY A <CR>.

'WHICH TEST ? ECHO OR CABLE (E OR C)''

LETS DO THE CABLE TEST FIRST. TYPE "C" AND A <CR>.

'BAUD RATE- ''

TYPE EITHER 50, 110, 135, 150, 300, 600, 1200 1800, 2000, 2400,
3600, 4800, 7200, 9600 FOLLOWED BY <CR>

'LINE: ''

YOU TYPE THE LINE WHICH HAS THE H325 TEST CONNECTOR. (TYPE
FITHER 0, 1, 2, 3) PROGRAM WILL THEN PRINT:

'CABLE TEST''

AND IF EVERYTHING IS WORKING, THE END OF PASS MESSAGE WILL BE
PRINTED AFTER EACH PA'S.

TO CHANGE LINES, HIT ANY PRINTING KEY ON YOUR CONSOLE TERMINAL
WHILE THE PROGRAM IS RUNNING AND THE FOLLOWING WILL BE PRINTED:

'LINE: ''

NOW CHANGE THE H325 TEST CONNECTOR TO ANOTHER LINE AND TYPE THE
NEW LINE. PROGRAM WILL THEN PRINT:

'CABLE TEST''

AND BEGIN RUNNING THE DIAGNOSTIC.
CONTINUE THIS OPERATION UNTIL ALL LINES ARE TESTED.

5.2 ECHO TEST

START THE PROGRAM AT LOC. 200 AND ENTER THE VALUES FOR THE CSR ADDRESS AND THE DEVICE VECTOR. THE PROGRAM WILL THEN PRINT OUT ON THE CONSOLE:

'WHICH TEST ? ECHO OR CABLE (E OR C)''

NOW TYPE AN 'E' TO DO THE ECHO TEST. PROGRAM WILL PRINT:

'BAUD RATE-''

TYPE THE BAUD RATE. BAUD RATE CHOICES ARE: 50, 75, 110, 135, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600. THE PROGRAM WILL THEN PRINT:

LINE: ''

TYPE THE LINE NUMBER WHICH THE TERMINAL IS CONNECTED TO. THEN THE PROGRAM WILL PRINT:

'TERMINAL ECHO TEST''

*** AT THIS POINT THE MESSAGE:

'THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789''

SHOULD BE PRINTED ON THE TERMINAL CONNECTED TO THE DZV11. IF THIS MESSAGE IS DESIRED TO BE PRINTED CONTINUOUSLY, TYPE A CONTROL G <^G> ON THE CONSOLE TERMINAL WHILE THE MESSAGE IS PRINTING. THE PROGRAM WILL PRINT A PROMPT ON THE CONSOLE ASKING FOR A NEW SWR SETTING. BY SETTING THE SWR TO 377 THE QUICK BROWN FOX MESSAGE WILL BE CONTINUOUSLY PRINTED ON THE DZV TERMINAL. A CONTROL G CAN THEN BE TYPED ON THE CONSOLE TERMINAL AT ANY TIME TO RESET THE SWR AND RETURN TO THE FLOW OF THE DIAGNOSTIC. THE PROGRAM WILL THEN PRINT ON THE CONSOLE TERMINAL:

'TYPE A CHAR. ON DZV11 TERMINAL''

ANY PRINTABLE CHARACTER WHICH IS TYPED ON THE DZV11 TERMINAL WILL BE ECHOED BACK ON THE TERMINAL. IF YOU TYPE CONTROL C <^C> ON THE DZV11 TERMINAL THE PROGRAM WILL PRINT THE END OF PASS MESSAGE ON THE CONSOLE TERMINAL AND THE 'QUICK BROWN FOX' MESSAGE WILL BEGIN PRINTING ON THE DZV11 TERMINAL AGAIN, THE ECHO TEST WILL BE RESUMED.

TO CHANGE LINES:

TYPE ANY PRINTABLE CHARACTER ON THE CONSOLE TERMINAL (NOT THE DZV11 TERMINAL). THE PROGRAM WILL AGAIN TYPE 'LINE: '' AND WAIT FOR A RESPONSE.

5.3 PROGRAM AND/OR OPERATOR ACTION

THE VARIETY OF PROGRAM CONTROL SWITCHES PROVIDED IN THIS DIAGNOSTIC PACKAGE IS DESIGNED TO PROVIDE THE USER WITH A WIDE RANGE OF TROUBLE-SHOOTING TECHNIQUES. BEFORE THE USER ATTEMPTS TO RUN THIS DIAGNOSTIC HE SHOULD BECOME FAMILIAR WITH THE USE OF THESE CONTROL SWITCHES AND THEIR RESTRICTIONS. (SEE SEC. 4.1, 4.1.1, 4.1.2, 4.1.3)

WHEN THE PROGRAM DETECTS AN ERROR THE TEST NUMBER AND PC WILL BE TYPED OUT AND POSSIBLY AN ERROR MESSAGE (DEPENDING ON THE PARTICULAR ERROR). IF IT IS NECESSARY TO KNOW MORE INFORMATION CONCERNING THE ERROR REPORT THEN LOOK IN THE PROGRAM LISTING FOR THAT TEST NUMBER AND THEN NOTE THE PC OF THE ERROR REPORT. THE REASON FOR THE ERROR REPORT WILL BECOME CLEARER WHEN READING THE COMMENTS IN THE PROGRAM LISTING.

6. ERRORS

AS DESCRIBED PREVIOUSLY THERE WILL ALWAYS BE A TEST NUMBER AND PC TYPED OUT AT THE TIME OF AN ERROR (PROVIDING SW 13=0 AND SW 12=0). IN MOST CASES ADDITIONAL INFORMATION WILL BE SUPPLIED TO THE THE ERROR MESSAGE WHICH IS TO GIVE THE OPERATOR AN INDICATION OF THE ERROR.

6.1 ERROR RECOVERY

IF FOR SOME REASON THE DZV11 SHOULD 'HANG THE BUS' (GAIN CONTROL OF BUS SO THAT CONSOLE MANUAL FUNCTIONS ARE INHIBITED) AN INIT OR POWER DOWN/UP IS NECESSARY FOR THE OPERATOR TO REGAIN CONTROL OF THE CPU. IT WILL THEN BE NECESSARY TO CHECK THE PC PROCESSOR REGISTER AND REFER TO THIS LOCATION IN THE PROGRAM LISTING TO FIND OUT WHAT THE PROGRAM WAS DOING AT THE TIME OF THE ERROR.

7. OPERATING RESTRICTIONS

WHEN RUNNING THE CABLE TEST, THE LINE THAT IS DECLARED ACTIVE MUST BE TERMINATED BY AN H325 TEST CONNECTOR WHICH WILL TURN THE TRANSMITTED SIGNAL AROUND TO THE RECEIVER ON THE SAME LINE. THE DIAGNOSTIC IS NOT DESIGNED TO DETERMINE A LOGIC PROBLEM WITH THE DZV INTERFACE. IT IS DESIGNED ONLY TO VERIFY THAT THE INTERFACE CABLE IS PROVIDING A TRUE LINK TO THE TERMINALS WHICH ARE CONNECTED TO THE DZV11.

8. MISCELLANEOUS

8.1 EXECUTION TIME

THE EXECUTION TIME FOR THE CABLE TEST DEPENDS UPON THE DESIRED BAUD RATE GIVEN AT START UP TIME. AT 9600. BAUD THE END PASS MESSAGE WILL PRINT OUT BEFORE 10 SECONDS HAVE ELAPSED.
THE EXECUTION TIME FOR THE ECHO TEST IS ENTIRELY DEPENDENT UPON THE NUMBER OF CHARACTERS THE OPERATOR WISHES TO SEND.

8.2 PASS COMPLETE

WHEN THE DIAGNOSTIC HAS COMPLETED A PASS THE FOLLOWING IS AN EXAMPLE OF THE PRINT OUT TO BE EXPECTED.

END PASS DVDZC-A CSR: 160100 VEC: 300 PASSES: 000001 ERRORS: 000000

NOTE: THE NUMBERS FOR CSR AND VEC ARE NOT NECESSARILY THE VALUES FOR THE DEVICE. THEY ARE ONLY FOR THIS EXAMPLE.

8.3 KEY LOCATIONS

AFTER THE BASE DEVICE ADDRESS AND THE BASE VECTOR HAVE BEEN TYPED IN, LOCATIONS 2010 THROUGH 2046 WILL CONTAIN THE VARIOUS DEVICE REGISTER ADDRESSES AND THE DEVICE VECTORS. LOCATION 1374 (SAVLIN) WILL CONTAIN THE LINE NUMBER THAT WAS DECLARED ACTIVE.

9.0 RUNNING THE DZV11 DIAGNOSTIC UNDER APT

9.1.1 THE APT INTERFACE

THE DZV DIAGNOSTICS HAVE BEEN DESIGNED TO BE COMPATIBLE WITH THE APT (AUTOMATED PRODUCT TEST) SYSTEM. THE DZV LOGIC TEST DIAGNOSTICS (DVDZA, AND DVDZB) CAN BE RUN AS STANDALONE DIAGNOSTICS OR IN EITHER OF THE APT MODES. DVDZC, HOWEVER IS DESIGNED AS A STANDALONE DIAGNOSTIC ONLY AND REQUIRES DIRECT OPERATOR PARTICIPATION.

9.1.2 SETTING UP THE DIAGNOSTIC USING APT

ONLY ONE VARIABLE IN THE REGION SUBTITLED 'APT MAILBOX-ETABLE' NEEDS TO BE SET UP BEFORE RUNNING UNDER APT. THIS VARIABLE IS:

SSWREG -(1142) USED AS THE SOFTWARE SWITCH REGISTER WHILE RUNNING UNDER APT.

9.1.3 RUNNING UNDER APT

SSWREG (LOC. 1142) SHOULD BE SET UP PRIOR TO RUNNING THE DIAGNOSTIC.

10.0 PROGRAM DESCRIPTION.

THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.

11

STARTING PROCEDURE

LOAD PROGRAM

START THE PROGRAM AT LOC. 000200

PROGRAM WILL TYPE DZV11 ECHO/CABLE TEST

PROGRAM WILL TYPE WHICH TEST- ECHO OR CABLE

TYPE IN E OR C RESPECTIVELY

PROGRAM WILL TYPE "VECTOR ADDRESS-"

TYPE IN THE ADDRESS OF THE RECEIVER INTERRUPT VECTOR

FOR THE DZV11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>

PROGRAM WILL TYPE "CONTROL REGISTER ADDRESS-"

TYPE IN THE ADDRESS OF THE SYSTEM CONTROL REGISTER

FOR THE DZV11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>

PROGRAM WILL TYPE "LINE NUMBER-"

TYPE IN THE LINE NUMBER TO BE TESTED (IN OCTAL)

, FOLLOWED BY <CARRIAGE RETURN>

PROGRAM WILL TYPE "BAUD RATE-"

TYPE IN THE BAUD RATE OF THE DZV11 TERMINAL

, FOLLOWED BY <CARRIAGE RETURN>

THE FOLLOWING BAUD RATES ARE ACCEPTED IN DECIMAL

50

75

110

135

(ROUNDED OFF 134.5)

150

300

600

1200

1800

2000

2400

3600

4800

7200

9600

ALL OTHERS ARE REJECTED

47

PROGRAM WILL TYPE "ECHO" OR "CABLE TEST" TO INDICATE THAT TESTING HAS STARTED

74

INITIAL ADDRESS OF THE STACK POINTER *** 1120 ***

79 MISCELLANEOUS DEFINITIONS

91 GENERAL PURPOSE REGISTER DEFINITIONS

103 PRIORITY LEVEL DEFINITIONS

113 "SWITCH REGISTER" SWITCH DEFINITIONS

141 DATA BIT DEFINITIONS (BIT00 TO BIT15)

169 BASIC "CPU" TRAP VECTOR ADDRESSES

384 BITS 15-11=CPU TYPE
11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
11/70=06,PDQ=07,Q=10
BIT 10=REAL TIME CLOCK
BIT 9=FLOATING POINT PROCESSOR
BIT 8=MEMORY MANAGEMENT

392 MEM.TYPE BYTE -- (HIGH BYTE)
900 NSEC CORE=001
300 NSEC BIPOLAR=002
500 NSEC MOS=003

397 MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE

436 THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
USED IN THE PROGRAM.

488 THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

494 EM ::POINTS TO THE ERROR MESSAGE
DH ::POINTS TO THE DATA HEADER
DT ::POINTS TO THE DATA
DF ::POINTS TO THE DATA FORMAT

873 INCREMENT THE PASS NUMBER (\$PASS)
IF THERES A MONITOR GO TO IT
IF THERE ISN'T JUMP TO XBEGIN

995 ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYIE.
THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
NOTE1: \$NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
NOTE2: \$FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
NOTE3: \$FILLC CONTAINS THE CHARACTER TO FILL AFTER.

CALL:
1) USING A TRAP INSTRUCTION
TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
OR

TYPE
MESADR

- 1728 ***** ECHO TEST *****
THIS TEST WILL ACCEPT 1 CHARACTER AT A TIME
(IN INTERRUPT MODE) AND TRANSMIT THAT SAME CHARACTER,
ONE LINE AT A TIME, ANY LINE 0 THRU 7 (OCTAL)
- 1799 ***** CABLE TEST *****
THIS TEST TRANSMITS A BINARY COUNT PATTERN
VIA INTERRUPT MODE TO THE RECEIVER
...THE LINE UNDER TEST MUST BE TERMINATED WITH THE TEST CONNECTOR
- 1808 TEST TO VERIFY THAT SETTING DTR FOR A GIVEN LINE
WILL BRING UP 'CO' AND 'RING' FOR THE SAME LINE
JUMPERS W1,W2,W3 AND W4 MUST BE INSTALLED ON THE
INTERFACE MODULE OTHERWISE AN ERROR REPORT WILL RESULT.

&

CVDZCB
CVDZCB.P11 10-AUG-81 10:56

.;GPA MACY11 30G(1063) ^{B 2} 10-AUG-81 11:15 PAGE 21

SEQ 0014

2144
2145
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(2)
(1)
(1)
(1)
(1)
(1)
(1)

000001

```
.;GPA PRGFRT ^?MAINDEC-11-DVDZCA/<200>/DZV11 ECHO AND CABLE TESTS ?.DVDZCA
.;GPA .HEADER <MD-11-DVDZC-A>,1977
.TITLE CVDZC-B
.*COPYRIGHT (C) 1977,1981
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC-C5), JAN, 1981.
.*
$TN=1
.*STARTING PROCEDURE
.*LOAD PROGRAM
.*START THE PROGRAM AT LOC. 000200
.*PROGRAM WILL TYPE DZV11 ECHO/CABLE TEST
.*PROGRAM WILL TYPE WHICH TEST- ECHO OR CABLE
.*TYPE IN E OR C RESPECTIVELY
.*PROGRAM WILL TYPE 'VECTOR ADDRESS-'
.*TYPE IN THE ADDRESS OF THE RECEIVER INTERRUPT VECTOR
.*FOR THE DZV11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
.*PROGRAM WILL TYPE 'CONTROL REGISTER ADDRESS-'
.*TYPE IN THE ADDRESS OF THE SYSTEM CONTROL REGISTER
.*FOR THE DZV11 TO BE TESTED, FOLLOWED BY <CARRIAGE RETURN>
.*PROGRAM WILL TYPE 'LINE NUMBER-'
.*TYPE IN THE LINE NUMBER TO BE TESTED (IN OCTAL)
.* FOLLOWED BY <CARRIAGE RETURN>
.*PROGRAM WILL TYPE 'BAUD RATE-'
.*TYPE IN THE BAUD RATE OF THE DZV11 TERMINAL
.* FOLLOWED BY <CARRIAGE RETURN>
.;*THE FOLLOWING BAUD RATES ARE ACCEPTED IN DECIMAL
.*          50
.*          75
.*          110           (ROUNDED OFF 134.5)
.*          135
.*          150
.*          300
.*          600
.*          1200
.*          1800
.*          2000
.*          2400
.*          3600
.*          4800
.*          7200
.*          9600
.*ALL OTHERS ARE REJECTED
.*PROGRAM WILL TYPE 'ECHO' OR 'CABLE TEST' TO INDICATE THAT TESTING HAS STARTED
.REM !
.;SWITCH REGISTER OPTIONS
;-----
SW15=100000           ;=1,HALT ON ERROR
```

```
(1) SW14=40000 :=1, LOOP ON CURRENT TEST
(1) SW13=20000 :=1, INHIBIT ERROR TYPEOUT
(1) SW12=10000 :=1, DELETE TYPEOUT/BELL ON ERROR.
(1) SW11=4000 :=1, INHIBIT ITERATIONS
(1) SW10=2000 :=1, ESCAPE TO NEXT TEST ON ERROR
(1) SW09=1000 :=1, LOOP WITH CURRENT DATA
(1) SW08=400 :=1, LOOP ON ERROR
(1) SW07=200 :=1, DO 'AUTO SIZING' ON INITIAL START UP.
(1) SW06=100 :=1, DESELECT SPECIFIC DEVICES
(1) :=NOTE: THIS MUST NOT EXCEED ORIGINAL COUNT
(1)
(1) SW05=40
(1) SW04=20 :=1, SELECT DELAY PARAMETER
(1) SW03=10 :=1, SELECT SPECIFIC PARAMETERS
(1) SW02=4 :=1, LOCK ON TEST SELECT
(1) SW01=2 :=1, RESTART PROGRAM AT SELECTED TEST
(1) SW00=1 :=1, SELECT DEVICE ADDRESS, VECTOR, ETC.
(1)
(2) .SBTTL BASIC DEFINITIONS
(2)
(2) :*INITIAL ADDRESS OF THE STACK POINTER *** 1120 ***
(2) 001120 STACK= 1120
(2) .EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
(2) .EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
(2)
(2) :*MISCELLANEOUS DEFINITIONS
(2) 000011 HT= 11 ;;CODE FOR HORIZONTAL TAB
(2) 000012 LF= 12 ;;CODE FOR LINE FEED
(2) 000015 CR= 15 ;;CODE FOR CARRIAGE RETURN
(2) 000200 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
(2) 177776 PS= 177776 ;;PROCESSOR STATUS WORD
(2) .EQUIV PS,PSW
(2) 177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
(2) 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
(2) 177570 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
(2) 177570 DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
(2)
(2) :*GENERAL PURPOSE REGISTER DEFINITIONS
(2) 000000 R0= %0 ;;GENERAL REGISTER
(2) 000001 R1= %1 ;;GENERAL REGISTER
(2) 000002 R2= %2 ;;GENERAL REGISTER
(2) 000003 R3= %3 ;;GENERAL REGISTER
(2) 000004 R4= %4 ;;GENERAL REGISTER
(2) 000005 R5= %5 ;;GENERAL REGISTER
(2) 000006 R6= %6 ;;GENERAL REGISTER
(2) 000007 R7= %7 ;;GENERAL REGISTER
(2) 000006 SP= %6 ;;STACK POINTER
(2) 000007 PC= %7 ;;PROGRAM COUNTER
(2)
(2) :*PRIORITY LEVEL DEFINITIONS
(2) 000000 PR0= 0 ;;PRIORITY LEVEL 0
(2) 000040 PR1= 40 ;;PRIORITY LEVEL 1
(2) 000100 PR2= 100 ;;PRIORITY LEVEL 2
(2) 000140 PR3= 140 ;;PRIORITY LEVEL 3
(2) 000200 PR4= 200 ;;PRIORITY LEVEL 4
(2) 000240 PR5= 240 ;;PRIORITY LEVEL 5
(2) 000300 PR6= 300 ;;PRIORITY LEVEL 6
```



```

(2)          000340          PR7=    340          ;;PRIORITY LEVEL 7
(2)
(2)          100000          ;*'SWITCH REGISTER' SWITCH DEFINITIONS
(2)          040000          SW15=   100000
(2)          020000          SW14=   40000
(2)          010000          SW13=   20000
(2)          004000          SW12=   10000
(2)          002000          SW11=   4000
(2)          001000          SW10=   2000
(2)          000400          SW09=   1000
(2)          000200          SW08=   400
(2)          000100          SW07=   200
(2)          000040          SW06=   100
(2)          000020          SW05=   40
(2)          000010          SW04=   20
(2)          000004          SW03=   10
(2)          000002          SW02=   4
(2)          000001          SW01=   2
(2)          000001          SW00=   1
(2)          .EQUIV          SW09,SW9
(2)          .EQUIV          SW08,SW8
(2)          .EQUIV          SW07,SW7
(2)          .EQUIV          SW06,SW6
(2)          .EQUIV          SW05,SW5
(2)          .EQUIV          SW04,SW4
(2)          .EQUIV          SW03,SW3
(2)          .EQUIV          SW02,SW2
(2)          .EQUIV          SW01,SW1
(2)          .EQUIV          SW00,SW0
(2)
(2)          100000          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(2)          040000          BIT15=  100000
(2)          020000          BIT14=  40000
(2)          010000          BIT13=  20000
(2)          004000          BIT12=  10000
(2)          002000          BIT11=  4000
(2)          001000          BIT10=  2000
(2)          000400          BIT09=  1000
(2)          000200          BIT08=  400
(2)          000100          BIT07=  200
(2)          000040          BIT06=  100
(2)          000020          BIT05=  40
(2)          000010          BIT04=  20
(2)          000004          BIT03=  10
(2)          000002          BIT02=  4
(2)          000001          BIT01=  2
(2)          000001          BIT00=  1
(2)          .EQUIV          BIT09,BIT9
(2)          .EQUIV          BIT08,BIT8
(2)          .EQUIV          BIT07,BIT7
(2)          .EQUIV          BIT06,BIT6
(2)          .EQUIV          BIT05,BIT5
(2)          .EQUIV          BIT04,BIT4
(2)          .EQUIV          BIT03,BIT3
(2)          .EQUIV          BIT02,BIT2
(2)          .EQUIV          BIT01,BIT1

```

```
(2) .EQUIV BIT00,BIT0
(2)
(2) ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(2) 000004 ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS
(2) 000010 RESVEC= 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
(2) 000014 TBITVEC=14 ;: 'T' BIT
(2) 000014 TRTVEC= 14 ;:TRACE TRAP
(2) 000014 BPTVEC= 14 ;:BREAKPOINT TRAP (BPT)
(2) 000020 IOTVEC= 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
(2) 000024 PWRVEC= 24 ;:POWER FAIL
(2) 000030 EMTVEC= 30 ;:EMULATOR TRAP (EMT) **ERROR**
(2) 000034 TRAPVEC=34 ;: 'TRAP' TRAP
(2) 000060 TKVEC= 60 ;:TTY KEYBOARD VECTOR
(2) 000064 TPVEC= 64 ;:TTY PRINTER VECTOR
(2) 000240 PIRQVEC=240 ;:PROGRAM INTERRUPT REQUEST VECTOR
(1)
(1) ;INSTRUCTION DEFINITIONS
(1) :-----
(1)
(1) 005746 PUSH1SP=5746 ;:DECREMENT PROCESSOR STACK 1 WORD
(1) 005726 POP1SP=5726 ;:INCREMENT PROCESSOR STACK 1 WORD
(1) 010046 PUSHRO=10046 ;:SAVE R0 ON STACK
(1) 012600 POPRO=12600 ;:RESTORE R0 FROM STACK
(1) 024646 PUSH2SP=24646 ;:DECREMENT STACK TWICE
(1) 022626 POP2SP=22626 ;:INCREMENT STACK TWICE
(1) 000200 MASK=BIT7 ;:SET INTERRUPT MASK (INHIBIT FURTHER INTERRUPTS)
(1) 000000 CLEAR=0 ;:ALLOW INTERRUPTS (CLEAR PROCESSOR STATUS)
(1)
(1) ;DZV11 CONTROL AND STATUS REGISTER DEFINITIONS
(1) ;(DZVCSR) BIT DEFINITIONS
(1) :-----
(1)
(1) 000010 MAINT = BIT3 ;:MAINTENANCE MODE ENABLE
(1) 000020 DCLR=BIT4 ;:DEVICE CLEAR
(1) 000040 MSENAB=BIT5 ;:MASTER SCAN ENABLE
(1) 000100 RIE=BIT6 ;:RECEIVER INTERRUPT ENABLE
(1) 000200 RDONE=BIT7 ;:RECEIVER DONE
(1) 010000 SILOEN= BIT12 ;:SILO ALARM ENABLE
(1) 020000 SILOAL = BIT13 ;:SILO ALARM
(1) 040000 TIE=BIT14 ;:TRANSMITTER INTERRUPT ENABLE
(1) 100000 TRDY=BIT15 ;:TRANSMITTER READY
(1)
(1) ;DZVCSR WORD DEFINITIONS
(1) :-----
(1) 000000 TLO=0 ;:TRANSMIT LINE 0
(1) 000400 TL1=BIT8 ;:TRANSMIT LINE 1
(1) 001000 TL2=BIT9 ;:TRANSMIT LINE 2
(1) 001400 TL3=BIT9!BIT8 ;:TRANSMIT LINE 3
(1)
(1) ;DZVRBUF BIT DEFINITIONS
(1) :-----
(1) 010000 PARER=BIT12 ;:PARITY ERROR
```

GENERAL DEFINITIONS AND EQUIVALENCES

(1)	020000	FRMERR=BIT13	:FRAME ERROR
(1)	040000	OVRRUN=BIT14	:OVERRUN ERROR
(1)	100000	DVALID=BIT15	:DATA VALID
(1)			
(1)			:DZVRBUF WORD DEFINITIONS
(1)			:-----
(1)	000000	RL0=0	:RECEIVER LINE 0
(1)	000400	RL1=BIT8	:RECEIVER LINE 1
(1)	001000	RL2=BIT9	:RECEIVER LINE 2
(1)	001400	RL3=BIT9!BIT8	:RECEIVER LINE 3
(1)			
(1)			:DZVLPR WORD DEFINITIONS
(1)			:-----
(1)	000000	LP0=0	:LINE PARAMETER 0
(1)	000001	LP1=BIT0	:LINE PARAMETER 1
(1)	000002	LP2=BIT1	:LINE PARAMETER 2
(1)	000003	LP3=BIT1!BIT0	:LINE PARAMETER 3
(1)			
(1)	000000	FIVE=0	:FIVE BITS/CHAR,1 STOP BIT
(1)	000010	SIX=BIT3	:SIX BITS/CHAR,1 STOP BIT
(1)	000020	SEVEN=BIT4	:SEVEN BITS/CHAR,1 STOP BIT
(1)	000030	EIGHT=BIT4!BIT3	:EIGHT BITS/CHAR,1 STOP BIT
(1)	000040	FIVES=BIT5	:FIVE BITS/CHAR,2 STOP BITS
(1)	000050	SIXS=BIT5!BIT3	:SIX BITS/CHAR,2 STOP BITS
(1)	000060	SEVENS=BIT5!BIT4	:SEVEN BITS/CHAR, 2 STOP BITS
(1)	000070	EIGHTS=BIT5!BIT4!BIT3	:EIGHT BITS/CHAR, 2 STOP BITS
(1)			
(1)	000100	PARITY=BIT6	:PARITY ENABLED
(1)	000200	ODDPAR=BIT7	:ODD PARITY ENABLED
(1)	000000	ONESTOP=0	:ONE STOP BIT ENABLED
(1)	000040	TWOSTOP=BIT5	:TWO STOP BITS ENABLED
(1)	000000	EVEPAR=0	:EVEN PARITY ENABLED
(1)	010000	RCVON=BIT12	:ENABLE RECEIVER (RECEIVER ON)
(1)			
(1)	000000	S50=0	:SPEED 50 BAUD
(1)	000400	S75=BIT8	:SPEED 75 BAUD
(1)	001000	S110=BIT9	:SPEED 110 BAUD
(1)	001400	S134=BIT9!BIT8	:SPEED 134.5 BAUD
(1)	002000	S150=BIT10	:SPEED 150 BAUD
(1)	002400	S300=BIT10!BIT8	:SPEED 300 BAUD
(1)	003000	S600=BIT10!BIT9	:SPEED 600 BAUD
(1)	003400	S1200=BIT10!BIT9!BIT8	:SPEED 1200 BAUD
(1)	004000	S1800=BIT11	:SPEED 1800 BAUD
(1)	004400	S2000=BIT11!BIT8	:SPEED 2000 BAUD
(1)	005000	S2400=BIT11!BIT9	:SPEED 2400 BAUD
(1)	005400	S3600=BIT11!BIT9!BIT8	:SPEED 3600 BAUD
(1)	006000	S4800=BIT11!BIT10	:SPEED 4800 BAUD
(1)	006400	S7200=BIT11!BIT10!BIT8	:SPEED 7200 BAUD
(1)	007000	S9600=BIT11!BIT10!BIT9	:SPEED 9600 BAUD
(1)	007400	S19200=BIT11!BIT10!BIT9!BIT8	:SPEED 19200 BAUD
(1)			
(1)			:DZVTCR BIT DEFINITIONS
(1)			:-----
(1)	000001	TCRC=BIT0	:ENABLE TRANSMISSION ON LINE 0

CVDZC-B MACY11 30G(1063) 10-AUG-81 11:15 PAGE 21-5
 CVDZCB.P11 10-AUG-81 10:56 GENERAL DEFINITIONS AND EQUIVALENCES

SEQ 0019

(1)	000002	TCR1=BIT1	:ENABLE TRANSMISSION ON LINE 1
(1)	000004	TCR2=BIT2	:ENABLE TRANSMISSION ON LINE 2
(1)	000010	TCR3=BIT3	:ENABLE TRANSMISSION ON LINE 3
(1)	000400	DTR0=BIT8	:DATA TERMINAL READY FOR LINE 0
(1)	001000	DTR1=BIT9	:DATA TERMINAL READY FOR LINE 1
(1)	002000	DTR2=BIT10	:DATA TERMINAL READY FOR LINE 2
(1)	004000	DTR3=BIT11	:DATA TERMINAL READY FOR LINE 3

:DZVMSR BIT DEFINITIONS

(1)	000001	RING0=BIT0	:RING INDICATED ON LINE 0
(1)	000002	RING1=BIT1	:RING INDICATED ON LINE 1
(1)	000004	RING2=BIT2	:RING INDICATED ON LINE 2
(1)	000010	RING3=BIT3	:RING INDICATED ON LINE 3
(1)	000400	C00=BIT8	:CARRIER PRESENT ON LINE 0
(1)	001000	C01=BIT9	:CARRIER PRESENT ON LINE 1
(1)	002000	C02=BIT10	:CARRIER PRESENT ON LINE 2
(1)	004000	C03=BIT11	:CARRIER PRESENT ON LINE 3

:DZVTDR BIT DEFINITIONS

(1)	000400	BRK0=BIT8	:BREAK FOR LINE 0
(1)	001000	BRK1=BIT9	:BREAK FOR LINE 1
(1)	002000	BRK2=BIT10	:BREAK FOR LINE 2
(1)	004000	BRK3=BIT11	:BREAK FOR LINE 3

- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)
- (1)

TABLE OF LOOP AROUND FUNCTIONS (H325)

I	^
V	^
REC	TRANS
DATA	DATA

I	^
V	^
CO	RTS

I	^
V	^
RING	DTR

TRAPCATCHER FOR UNEXPECTED INTERRUPTS

```
(1) ;:*****  
(1) ;-----  
(1) ;TRAPCATCHER FOR ILLEGAL INTERRUPTS  
(1) ;THE STANDARD 'TRAP CATCHER' IS PLACED  
(1) ;BETWEEN ADDRESS 0 TO ADDRESS 776.  
(1) ;IT LOOKS LIKE 'PC+2 HALT'.  
(1) ;-----  
(1) ;:*****  
(1) ;  
(1) 000000 . =0 ;STANDARD INTERRUPT VECTORS  
(1) ;-----  
(1) ;  
(1) 000024 . =24 ;POWER FAIL HANDLER  
(1) 000024 005702 $PWRDN ;SERVICE AT PRIORITY LEVEL 7  
(1) 000026 000340 340 ;ERROR HANDLER  
(1) 000030 005010 $ERRR ;SERVICE AT PRIORITY LEVEL 7  
(1) 000032 000340 340 ;GENERAL HANDLER DISPATCH SERVICE  
(1) 000034 004602 .TRPSRV ;SERVICE AT PRIORITY LEVEL 7  
(1) 000036 000340 340 ;  
(2) .SBTTL ACT11 HOOKS  
(2) ;:*****  
(3) ;HOOKS REQUIRED BY ACT11  
(2) 000040 $SVPC= . ;SAVE PC  
(2) 000046 . =46 ;:1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP  
(2) 000046 002676 $ENDAD ;:2)SET LOC.52 TO ZERO  
(2) 000052 000052 . =52 ;: RESTORE PC  
(2) 000052 000000 .WORD 0  
(2) 000052 000040 . = $SVPC  
(1) ;  
(1) 000174 . =174 ;SOFTWARE DISPLAY REGISTER FOR SWITCHLESS 11S  
(1) 000174 000000 DISPREG:0 ;SOFTARE SWITCH REGISTER FOR SWITCHLESS 11S  
(1) 000176 000000 SWREG: 0  
(1) 000200 000200 . =200 ;GO TO START OF PROGRAM  
(1) 000200 000137 002116 JMP .START  
(2) ;  
(2) 001000 . =1000  
(2) 001000 005200 053103 055104 MTITLE: .ASCIZ <200><12>/CVDZCB/<200>/DZV11 ECHO AND CABLE TESTS /<200>  
(2)
```

```
(3)          001120          . =1120
(4)          :*****
(4)          :SBTTL  APT MAILBOX-ETABLE
(4)          :*****
(4)          :EVEN
(4) 001120   $MAIL:          ::APT MAILBOX
(4) 001120   $MSGTY: .WORD   AMSGTY ::MESSAGE TYPE CODE
(4) 001122   $FATAL: .WORD   AFATAL ::FATAL ERROR NUMBER
(4) 001124   $TESTN: .WORD   ATESTN ::TEST NUMBER
(4) 001126   $PASS:  .WORD   APASS  ::PASS COUNT
(4) 001130   $DEVCT: .WORD   ADEVCT ::DEVICE COUNT
(4) 001132   $UNIT:  .WORD   AUNIT  ::I/O UNIT NUMBER
(4) 001134   $MSGAD: .WORD   AMSGAD ::MESSAGE ADDRESS
(4) 001136   $MSGLG: .WORD   AMSGLG ::MESSAGE LENGTH
(4) 001140   $ETABLE:      ::APT ENVIRONMENT TABLE
(4) 001140   $ENV:  .BYTE   AENV    ::ENVIRONMENT BYTE
(4) 001141   $ENVM: .BYTE   AENVM  ::ENVIRONMENT MODE BITS
(4) 001142   $SWREG: .WORD   ASWREG ::APT SWITCH REGISTER
(4) 001144   $USWR:  .WORD   AUSWR  ::USER SWITCHES
(4) 001146   $CPUOP: .WORD   ACPUOP ::CPU TYPE,OPTIONS
(4)          :*
(4)          :*
(4)          :*
(4)          :*
(4)          :*
(4)          :*
(4)          :*
(4)          :*
(4) 001150   $MAMS1: .BYTE   AMAMS1 ::HIGH ADDRESS,M.S. BYTE
(4) 001151   $MTYP1: .BYTE   AMTYP1 ::MEM. TYPE,BLK#1
(4)          :*
(4)          :*
(4)          :*
(4)          :*
(4) 001152   $MADR1: .WORD   AMADR1 ::HIGH ADDRESS,BLK#1
(4)          :*
(4) 001154   $MAMS2: .BYTE   AMAMS2 ::HIGH ADDRESS,M.S. BYTE
(4) 001155   $MTYP2: .BYTE   AMTYP2 ::MEM. TYPE,BLK#2
(4) 001156   $MADR2: .WORD   AMADR2 ::MEM.LAST ADDRESS,BLK#2
(4) 001160   $MAMS3: .BYTE   AMAMS3 ::HIGH ADDRESS,M.S.BYTE
(4) 001161   $MTYP3: .BYTE   AMTYP3 ::MEM. TYPE,BLK#3
(4) 001162   $MADR3: .WORD   AMADR3 ::MEM.LAST ADDRESS,BLK#3
(4) 001164   $MAMS4: .BYTE   AMAMS4 ::HIGH ADDRESS,M.S.BYTE
(4) 001165   $MTYP4: .BYTE   AMTYP4 ::MEM. TYPE,BLK#4
(4) 001166   $MADR4: .WORD   AMADR4 ::MEM.LAST ADDRESS,BLK#4
(4) 001170   $VECT1: .WORD   AVECT1 ::INTERRUPT VECTOR#1,BUS PRIORITY#1
(4) 001172   $VECT2: .WORD   AVECT2 ::INTERRUPT VECTOR#2BUS PRIORITY#2
(4) 001174   $BASE:  .WORD   ABASE  ::BASE ADDRESS OF EQUIPMENT UNDER TEST
(4) 001176   $DEV#1: .WORD   ADEV#1 ::DEVICE MAP
(4) 001200   $CDW1:  .WORD   ACDW1  ::CONTROLLER DESCRIPTION WORD#1
(4) 001202   $CDW2:  .WORD   ACDW2  ::CONTROLLER DESCRIPTION WORD#2
(4) 001204   $DDW0:  .WORD   ADDW0  ::DEVICE DESCRIPTOR WORD#0
(4) 001206   $DDW1:  .WORD   ADDW1  ::DEVICE DESCRIPTOR WORD#1
(4) 001210   $DDW2:  .WORD   ADDW2  ::DEVICE DESCRIPTOR WORD#2
(4) 001212   $DDW3:  .WORD   ADDW3  ::DEVICE DESCRIPTOR WORD#3
(4) 001214   $DDW4:  .WORD   ADDW4  ::DEVICE DESCRIPTOR WORD#4
(4) 001216   $DDW5:  .WORD   ADDW5  ::DEVICE DESCRIPTOR WORD#5
```



```

(3)          .SBTTL COMMON TAGS
(3)
(4)          ::*****
(3)          ::*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(3)          ::*USED IN THE PROGRAM.
(3)
(3)          $CMTAG:                ::START OF COMMON TAGS
(3)          001244                000000
(3)          001244                000000      .WORD 0
(3)          001246                000      $TSTNM: .BYTE 0
(3)          001247                000      $ERFLG: .BYTE 0
(3)          001250                000000      $ICNT:  .WORD 0
(3)          001252                000000      $LPADR: .WORD 0
(3)          001254                000000      $LPERR: .WORD 0
(3)          001256                000000      $ERTTL: .WORD 0
(3)          001260                000      $ITEMB: .BYTE 0
(3)          001261                001      $ERMAX: .BYTE 1
(3)          001262                000000      $ERRPC: .WORD 0
(3)          001264                000000      $GDADR: .WORD 0
(3)          001266                000000      $BDADR: .WORD 0
(3)          001270                000000      $GDDAT: .WORD 0
(3)          001272                000000      $BDDAT: .WORD 0
(3)          001274                000000      .WORD 0
(3)          001276                000000      .WORD 0
(3)          001300                000      $AUTOB: .BYTE 0
(3)          001301                000      $INTAG: .BYTE 0
(3)          001302                000000      .WORD 0
(3)          001304                177570      $SWR:   .WORD DSWR
(3)          001306                177570      $DISPLAY: .WORD DDISP
(3)          001310                177560      $TKS:   177560
(3)          001312                177562      $TKB:   177562
(3)          001314                177564      $TPS:   177564
(3)          001316                177566      $TPB:   177566
(3)          001320                000      $NULL:  .BYTE 0
(3)          001321                002      $FILLS: .BYTE 2
(3)          001322                012      $FILLC: .BYTE 12
(3)          001323                000      $TPFLG: .BYTE 0
(3)          001324                000000      $REGAD: .WORD 0
(3)
(5)          001326                000000      $REG0:  .WORD 0
(5)          001330                000000      $REG1:  .WORD 0
(5)          001332                000000      $REG2:  .WORD 0
(5)          001334                000000      $REG3:  .WORD 0
(5)          001336                000000      $REG4:  .WORD 0
(5)          001340                000000      $REG5:  .WORD 0
(5)          001342                000000      $TMP0:  .WORD 0
(5)          001344                000000      $TMP1:  .WORD 0
(5)          001346                000000      $TMP2:  .WORD 0
(5)          001350                000000      $TMP3:  .WORD 0
(5)          001352                000000      $TMP4:  .WORD 0
(3)          001354                000000      $TIMES: 0
(3)          001356                077      $QUES:  .ASCII /?/
(3)          001357                015      $CRLF:  .ASCII <15>
(3)          001360                000012      $LF:   .ASCIIZ <12>

```

```

::CONTAINS THE TEST NUMBER
::CONTAINS ERROR FLAG
::CONTAINS SUBTEST ITERATION COUNT
::CONTAINS SCOPE LOOP ADDRESS
::CONTAINS SCOPE RETURN FOR ERRORS
::CONTAINS TOTAL ERRORS DETECTED
::CONTAINS ITEM CONTROL BYTE
::CONTAINS MAX. ERRORS PER TEST
::CONTAINS PC OF LAST ERROR INSTRUCTION
::CONTAINS ADDRESS OF 'GOOD' DATA
::CONTAINS ADDRESS OF 'BAD' DATA
::CONTAINS 'GOOD' DATA
::CONTAINS 'BAD' DATA
::RESERVED--NOT TO BE USED
::AUTOMATIC MODE INDICATOR
::INTERRUPT MODE INDICATOR
::ADDRESS OF SWITCH REGISTER
::ADDRESS OF DISPLAY REGISTER
::TTY KBD STATUS
::TTY KBD BUFFER
::TTY PRINTER STATUS REG. ADDRESS
::TTY PRINTER BUFFER REG. ADDRESS
::CONTAINS NULL CHARACTER FOR FILLS
::CONTAINS # OF FILLER CHARACTERS REQUIRED
::INSERT FILL CHARS. AFTER A 'LINE FEED'
::"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
::CONTAINS THE ADDRESS FROM
::WHICH ($REG0) WAS OBTAINED
::CONTAINS (($REGAD)+0)
::CONTAINS (($REGAD)+2)
::CONTAINS (($REGAD)+4)
::CONTAINS (($REGAD)+6)
::CONTAINS (($REGAD)+10)
::CONTAINS (($REGAD)+12)
::USER DEFINED
::USER DEFINED
::USER DEFINED
::USER DEFINED
::USER DEFINED
::MAX. NUMBER OF ITERATIONS
::QUESTION MARK
::CARRIAGE RETURN
::LINE FEED

```

```

(3) .SBTTL ERROR POINTER TABLE
(3)
(3) ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(3) ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(3) ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(3) ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(3) ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(3)
(3) ;* EM ;:POINTS TO THE ERROR MESSAGE
(3) ;* DH ;:POINTS TO THE DATA HEADER
(3) ;* DT ;:POINTS TO THE DATA
(3) ;* DF ;:POINTS TO THE DATA FORMAT
(3)
(3) 001362 $ERRTB:
(2) ;PROGRAM CONTROL PARAMETERS
(2) ;-----
(2) 001362 000000 NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED
(2) 001364 000000 LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT TEST,TIGHT LOOP
(2)
(2) ;PROGRAM VARIABLES
(2) ;-----
(2) 001366 000017 LINE: 17 ;DEFAULT ALL FOUR LINES RUNNING
(2) 001370 017470 PAR: 17470 ;PARAMETERS: 8 BITS/CHAR,2 STOP BITS,19200 BAUD,NO PARIT
(2) 001372 000000 MODE: 0 ;DEFAULT MAINTENANCE MODE
(2) 001374 000000 SAVLIN: 0 ;LINE NUMBER
(2) 001376 000000 XMTLIN: 0 ;TRANSMISSION LINE NUMBER
(2) 001400 000000 XMTCNT: 0 ;COUNT OF WORDS IN A TRANSMISSION PATTERN
(2) 001402 000000 REGIST: 0 ;DEVICE ADDRESS STORAGE LOCATION
(2) 001404 000000 SAVPC: 0 ;PROGRAM COUNTER STORAGE
(2) 001406 000001 DZVACTV:.BLKW 1 ;*DZV11'S SELECTED ACTIVE.
(2) 001410 000001 SAVACTV:.BLKW 1 ;*A BIT MAP OF DZV11'S IN THE SYSTEM
(2) 001412 000001 RUN: 1 ;*POINTER ONE PAST RUNNING DEVICE.
(2) 001414 000001 DZVNUM: .BLKB 1 ;*OCTAL NUMBER OF DZV11'S IN THE SYSTEM
(2) 001415 001 SAVNUM: .BYTE 1 ;*WORKABLE NUMBER.
(2) 001416 000001 SAVNO: .BLKB 1 ;*OCTAL NUMBER OF DZV11'S BEING TESTED
(2) 001420 001420 .EVEN
(2) 001420 001500 ACTIVE: DZV.MAP ;TABLE POINTER.

```

```
(2)
(2)                                ;PROGRAM CONTROL FLAGS
(2)                                ;-----
(2)                                ;
(2)                                ;
(2) 001422      000      INIFLG: .BYTE 0      ;PROGRAM INITIALIZATION FLAG
(2) 001423      000      HDRFLG: .BYTE 0      ;PROGRAM INITIALIZATION FLAG FOR HEADER MAP
(2) 001424      000      MNTFLG: .BYTE 0      ;MAINTENANCE BIT SET FLAG
(2) 001425      000      DONFLG: .BYTE 0      ;TRANSMISSION COMPLETION FLAG
(2)                                ;
(2)                                ;
(2)                                ;DATA VARIABLES
(2) 001426      000000    TD0:      .WORD 0
(2) 001430      000000    TD1:      .WORD 0
(2) 001432      000000    TD2:      .WORD 0
(2) 001434      000000    TD3:      .WORD 0
(2) 001436      000000    TR0:      .WORD 0
(2) 001440      000000    TR1:      .WORD 0
(2) 001442      000000    TR2:      .WORD 0
(2) 001444      000000    TR3:      .WORD 0
(2) 001446
(2)                                ;STOP:
(2)                                ;.SBTTL APT PARAMETER BLOCK
(2)                                ;
(2)                                ;*****
(2)                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(2)                                ;*****
(2)                                ;
(2)                                ;$X=      ;SAVE CURRENT LOCATION
(2)                                ;=24      ;SET POWER FAIL TO POINT TO START OF PROGRAM
(2) 000024      000024    200      ;FOR APT START UP
(2)                                ;=44      ;POINT TO APT INDIRECT ADDRESS PNTR.
(2) 000044      001446    $APTHDR ;POINT TO APT HEADER BLOCK
(2)                                ;=.SX     ;RESET LOCATION COUNTER
(2)                                ;*****
(2)                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(2)                                ;INTERFACE SPEC.
(2)                                ;
(2)                                ;
(2) 001446      000000    $APTHD:
(2) 001446      000000    $HIBTS: .WORD 0      ;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(2) 001450      001120    $MBADR: .WORD $MAIL ;ADDRESS OF APT MAILBOX (BITS 0-15)
(2) 001452      000000    $TSTM:  .WORD 0      ;RUN TIM OF LONGEST TEST
(2) 001454      000000    $PASTM: .WORD 0      ;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(2) 001456      000000    $SUNITM: .WORD 0     ;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(2) 001460      000052    .WORD $ETEND-$MAIL/2 ;LENGTH MAILBOX-ETABLE(WORDS)
(1)                                ;DZV11 STATUS TABLE AND ADDRESS ASSIGNMENTS
(1)                                ;-----
(1)                                ;
(1)                                ;
(1) 001500      001500    .=1500
(1) 001500      001500    DZV.MAP:
(3)                                ;
(3) 001500      000001    DZCR0:  .BLKW 1      ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 0
(3) 001502      000001    DZVC0:  .BLKW 1      ;RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 0
(3) 001504      000001    LINE0:  .BLKW 1      ;ALL LINES SELECTED
(3) 001506      000001    PAR0:   .BLKW 1      ;PARAMETERS
(3) 001510      000001    MANT0:  .BLKW 1      ;MAINTENANCE MODE FOR THIS DEVICE
(3)                                ;
(3) 001512      000001    DZCR1:  .BLKW 1      ;CONTROL STATUS REGISTER FOR DZV11 NUMBER 1
(3) 001514      000001    DZVC1:  .BLKW 1      ;RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 1
(3) 001516      000001    LINE1:  .BLKW 1      ;ALL LINES SELECTED
```

(3)	001520	000001	PAR1:	.BLKW	1	:PARAMETERS
(3)	001522	000001	MANT1:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)	001524	000001	DZCR2:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 2
(3)	001526	000001	DZVC2:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 2
(3)	001530	000001	LINE2:	.BLKW	1	:ALL LINES SELECTED
(3)	001532	000001	PAR2:	.BLKW	1	:PARAMETERS
(3)	001534	000001	MANT2:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)	001536	000001	DZCR3:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 3
(3)	001540	000001	DZVC3:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 3
(3)	001542	000001	LINE3:	.BLKW	1	:ALL LINES SELECTED
(3)	001544	000001	PAR3:	.BLKW	1	:PARAMETERS
(3)	001546	000001	MANT3:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)	001550	000001	DZCR4:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 4
(3)	001552	000001	DZVC4:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 4
(3)	001554	000001	LINE4:	.BLKW	1	:ALL LINES SELECTED
(3)	001556	000001	PAR4:	.BLKW	1	:PARAMETERS
(3)	001560	000001	MANT4:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)	001562	000001	DZCR5:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 5
(3)	001564	000001	DZVC5:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 5
(3)	001566	000001	LINE5:	.BLKW	1	:ALL LINES SELECTED
(3)	001570	000001	PAR5:	.BLKW	1	:PARAMETERS
(3)	001572	000001	MANT5:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)	001574	000001	DZCR6:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 6
(3)	001576	000001	DZVC6:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 6
(3)	001600	000001	LINE6:	.BLKW	1	:ALL LINES SELECTED
(3)	001602	000001	PAR6:	.BLKW	1	:PARAMETERS
(3)	001604	000001	MANT6:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)	001606	000001	DZCR7:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 7
(3)	001610	000001	DZVC7:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 7
(3)	001612	000001	LINE7:	.BLKW	1	:ALL LINES SELECTED
(3)	001614	000001	PAR7:	.BLKW	1	:PARAMETERS
(3)	001616	000001	MANT7:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)	001620	000001	DZCR10:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 10
(3)	001622	000001	DZVC10:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 10
(3)	001624	000001	LINE10:	.BLKW	1	:ALL LINES SELECTED
(3)	001626	000001	PAR10:	.BLKW	1	:PARAMETERS
(3)	001630	000001	MANT10:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)	001632	000001	DZCR11:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 11
(3)	001634	000001	DZVC11:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 11
(3)	001636	000001	LINE11:	.BLKW	1	:ALL LINES SELECTED
(3)	001640	000001	PAR11:	.BLKW	1	:PARAMETERS
(3)	001642	000001	MANT11:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)	001644	000001	DZCR12:	.BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 12
(3)	001646	000001	DZVC12:	.BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 12
(3)	001650	000001	LINE12:	.BLKW	1	:ALL LINES SELECTED
(3)	001652	000001	PAR12:	.BLKW	1	:PARAMETERS
(3)	001654	000001	MANT12:	.BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE

CVDZC-B MACY11 30G(1063) 10-AUG-81 11:15 PAGE 21-14
 CVDZCB.P11 10-AUG-81 10:56 APT PARAMETER BLOCK

SEQ 0028

(3)					
(3)	001656	000001	DZCR13: .BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 13
(3)	001660	000001	DZVC13: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 13
(3)	001662	000001	LINE13: .BLKW	1	:ALL LINES SELECTED
(3)	001664	000001	PAR13: .BLKW	1	:PARAMETERS
(3)	001666	000001	MANT13: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001670	000001	DZCR14: .BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 14
(3)	001672	000001	DZVC14: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 14
(3)	001674	000001	LINE14: .BLKW	1	:ALL LINES SELECTED
(3)	001676	000001	PAR14: .BLKW	1	:PARAMETERS
(3)	001700	000001	MANT14: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001702	000001	DZCR15: .BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 15
(3)	001704	000001	DZVC15: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 15
(3)	001706	000001	LINE15: .BLKW	1	:ALL LINES SELECTED
(3)	001710	000001	PAR15: .BLKW	1	:PARAMETERS
(3)	001712	000001	MANT15: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001714	000001	DZCR16: .BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 16
(3)	001716	000001	DZVC16: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 16
(3)	001720	000001	LINE16: .BLKW	1	:ALL LINES SELECTED
(3)	001722	000001	PAR16: .BLKW	1	:PARAMETERS
(3)	001724	000001	MANT16: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(3)					
(3)	001726	000001	DZCR17: .BLKW	1	:CONTROL STATUS REGISTER FOR DZV11 NUMBER 17
(3)	001730	000001	DZVC17: .BLKW	1	:RECEIVER AND BASE VECTOR FOR DZV11 NUMBER 17
(3)	001732	000001	LINE17: .BLKW	1	:ALL LINES SELECTED
(3)	001734	000001	PAR17: .BLKW	1	:PARAMETERS
(3)	001736	000001	MANT17: .BLKW	1	:MAINTENANCE MODE FOR THIS DEVICE
(1)					
(1)	001740	177777	DZV.END:	177777	

(1) :DEFINITIONS FOR TRAP SUBROUTINE CALLS
(1) :POINTERS TO SUBROUTINES CAN BE FOUND
(1) :IN THE TABLE IMMEDIATELY FOLLOWING THE DEFINITIONS
(1)

:*****

(1) :-----
(1) .TRPTAB:
(3) 001742 104400 ADVANCE=TRAP+0 :CALL TO ADVANCE TO NEXT TEST
(2) 001742 004676 .ADVANCE
(3) 104401 SCOP1=TRAP+1 :CALL TO LOOP ON CURRENT DATA HANDLER
(2) 001744 003136 .SCOP1
(3) 104402 TYPE=TRAP+2 :CALL TO TELETYPE OUTPUT ROUTINE
(2) 001746 003162 .TYPE
(3) 104403 INSTR=TRAP+3 :CALL TO ASCII STRING INPUT ROUTINE
(2) 001750 004002 .INSTR
(3) 104404 INSTER=TRAP+4 :CALL TO INPUT ERROR HANDLER
(2) 001752 004106 .INSTER
(3) 104405 PARAM=TRAP+5 :CALL TO NUMERICAL DATA INPUT ROUTINE
(2) 001754 004126 .PARAM
(3) 104406 SETFLG=TRAP+6 :CALL TO SET FLAG ROUTINE
(2) 001756 006536 .SETFLG
(3) 104407 SAV05=TRAP+7 :CALL TO REGISTER SAVE ROUTINE
(2) 001760 004326 .SAV05
(3) 104410 RES05=TRAP+10 :CALL TO REGISTER RESTORE ROUTINE
(2) 001762 004366 .RES05
(3) 104411 CONVRT=TRAP+11 :CALL TO DATA OUTPUT ROUTINE
(2) 001764 004420 .CONVRT
(3) 104412 CNVRT=TRAP+12 :CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
(2) 001766 004424 .CNVRT
(3) 104413 DEVICE.CLR=TRAP+13 :CALL TO ISSUE A DEVICE CLEAR
(2) 001770 004624 .DEVICE.CLR
(3) 104414 DELAY=TRAP+14 :CALL TO DELAY FOR FAST CPU'S
(2) 001772 004656 .DELAY
(3) 104415 PARMD=TRAP+15 :CONVERT DECIMAL STRING TO OCTAL
(2) 001774 002742 .PARMD
(3) 104416 PAWCH=TRAP+16 :SET FLAG ECHO OR CABLE
(2) 001776 006656 .PAWCH
(3) 104417 DCLASM=TRAP+17 :CLEAR DEVICE, SET MAINT. BIT IF I MODE
(2) 002000 004644 .DCLASM
(3) 104420 SHIFT=TRAP+20 :CALL TO ROTATE LINE POINTER
(2) 002002 004710 .SHIFT
(3) 104421 LPRSET=TRAP+21 :CALL TO SET UP LPR DEVICE REGISTER
(2) 002004 004726 .LPRSET
(3) 104422 BUFSET=TRAP+22 :CALL TO ZERO BUFFER AREA
(2) 002006 004766 .BUFSET

:*****

(1)
(1)
(1)

```
(1)                                     :DZV11 VECTOR AND REGISTER INDIRECT POINTERS
(1)                                     :WORKING AREA
(1)
(1) 002010 160040 DZVCSR: 160040 :R/W
(1) 002012 160041 HDZVCSR:160041 :R/W
(1) 002014 160042 DZVRBUF:160042 :READ ONLY
(1) 002016 160043 HDZVRBUF:160043 :READ ONLY
(1) 002020 160042 DZVLPR: 160042 :WRITE ONLY
(1) 002022 160043 HDZVLPR:160043 :WRITE ONLY
(1) 002024 160044 DZVTCR: 160044 :R/W
(1) 002026 160045 HDZVTCR:160045 :R/W
(1) 002030 160046 DZVMSR: 160046 :READ ONLY
(1) 002032 160047 HDZVMSR:160047 :READ ONLY
(1) 002034 160046 DZVTDR: 160046 :WRITE ONLY
(1) 002036 160047 HDZVTDR:160047 :WRITE ONLY
(1)
(1)                                     :DEFAULT DZV VECTORS
(1)
(1) 002040 000300 DZVRIV: 300 :REC INTR VECTOR
(1) 002042 000302 DZVRIS: 302 :REC INTR STATUS
(1) 002044 000304 DZVTIV: 304 :XMIT INTR VECTOR
(1) 002046 000306 DZVTIS: 306 :XMIT INTR STATUS
(1)
(1)
```

(1)
(1) :TIME TABLE FOR RELATIVE TIMING TESTS
(1) :-----
(1)
(1) 002050 TMTBL:
(1) 002050 000000 T50: 0
(1) 002052 000000 T75: 0
(1) 002054 000000 T110: 0
(1) 002056 000000 T134: 0
(1) 002060 000000 T150: 0
(1) 002062 000000 T300: 0
(1) 002064 000000 T600: 0
(1) 002066 000000 T1200: 0
(1) 002070 000000 T1800: 0
(1) 002072 000000 T2000: 0
(1) 002074 000000 T2400: 0
(1) 002076 000000 T3600: 0
(1) 002100 000000 T4800: 0
(1) 002102 000000 T7200: 0
(1) 002104 000000 T9600: 0
(1) 002106 000000 TEIGHT: 0
(1) 002110 000000 TSEVEN: 0
(1) 002112 000000 TSIX: 0
(1) 002114 000000 TFIVE: 0


```

(1)
(1) :PROGRAM INITIALIZATION
(1) :LOCK OUT INTERRUPTS
(1) :SET UP PROCESSOR STACK
(1) :SET UP POWER FAIL VECTOR
(1) :CLEAR PROGRAM CONTROL FLAGS AND COUNTS
(1) :TYPE TITLE MESSAGE
(1)
(1)
(1) 002116 000005 .START: RESET :CLEAR THE WORLD
(1) 002120 012706 001120 MOV #STACK,SP :SET UP PROCESSOR STACK
(1) 002124 106427 000200 MTPS #MASK :LOCK OUT INTERRUPTS
(1) 002130 012737 005702 000024 MOV #SPWRDN,#24 :SET UP FOR POWER FAIL
(1) 002136 012737 002116 001252 MOV #.START,$LPADR :SET UP IN CASE OF POWER FAIL
(1) 002144 105737 001141 TSTB $ENVM :RUNNING UNDER APT?
(1) 002150 100004 BPL 1$ :IF NOT SKIP SWR SETUP FOR APT
(1) 002152 012737 001142 001304 MOV #$$SWREG,SWR :SETUP FOR APT SWR
(1) 002160 000405 BR 2$ :SKIP SOFTWARE SWR SETUP
(1) 002162 012737 000176 001304 1$: MOV #SWREG,SWR :SETUP SOFTWARE SWITCH REGISTER OTHERWISE
(1) 002170 004737 005464 CALL GETSWR : GET INITIAL SWICT SETTING ::GPA
(1) 002174 012737 000174 001306 2$: MOV #DISPREG,DISPLAY :SETUP DISPLAY REGISTER
(1) 002202 005037 007130 CLR STFLG :CLEAR TEST START FLAG
(1) 002206 005037 001126 CLR $PASS :CLEAR PASS COUNT
(1) 002212 005037 001256 CLR $ERTTL :CLEAR ERROR COUNT
(1) 002216 105037 001247 CLR $ERFLG :CLEAR ERROR FLAG
(1) 002222 005037 001246 CLR $STNM :CLEAR TEST NO. INDICATOR
(1) 002226 005037 007134 CLR LAST :CLEAR LAST ERROR PC
(1) 002232 004737 013304 CALL FALCON : TEST FOR FALCON (KXT11) ::GPA
(1) 002236 001402 BEQ 1000$ : BR IF NOT ::GPA
(1) 002240 004737 000570 CALL FALCINI : INIT FOR FALCON SYSTEM ::GPA
(1) 002244 1000$: TSTB INIFLG :HAS TITLE BEEN TYPED YET?
(1) 002244 105737 001422 BNE VEC1 :IF YES SKIP PRINTING AGAIN
(1) 002250 001010 CMP #42,$SENDAD :RUNNING UNDER ACT?
(1) 002252 023727 000042 002676 BEQ 3$ :IF YES DON'T PRINT TITLE
(1) 002260 001402 TYPE ,MTITLE :PRINT TITLE
(1) 002262 104402 001000 DECIB INIFLG :INDICATE TITLE ALREADY TYPED
(1) 002266 105337 001422 3$: MOV #300,R1
(1) 002272 012701 000300 VEC1: MOV #302,R2
(1) 002276 012702 000302 1$: MOV R2,(R1)+
(1) 002302 010221 CLR (R2)+ :RESTORE TRAPCATCHER
(1) 002304 005022 CMP (R1)+,(R2)+ :IN FLOATING VECTOR AREA
(1) 002306 022122 TST KXTFLAG :UPDATE THE POINTERS
(1) 002310 005737 013320 BEQ 1001$ : IF FALCON... ::GPA
(1) 002314 001403 BEQ 1001$ : :GPA
(1) 002316 020127 000400 CMP R1,#400 :...QUIT AT 400. :GPA
(1) 002322 000402 402 : SKIP NEXT :GPA
(1) 002324 1001$: CMP R1,#1000 :GPA
(1) 002324 020127 001000 BNE 1$
(1)
(1) 002332 002332 GETCSR= . : POINTER FOR FALCON TWEAKER. ::GPA
(1) 002332 104403 INSTR :INPUT ADDRESS OF DEVICE CSR
(1) 002334 007204 MREGAD :MESSAGE "CONTROL REGISTER ADDRESS-"
(1) 002336 104405 PARAM :CONVERT STRING TO OCTAL
(1) 002340 160000 160000 :LOW LIMIT
(1) 002342 163770 163770 :HIGH LIMIT
(1) 002344 001174 $BASE :LOCATION TO BE FILLED

```

```

(1) 002346      007      .BYTE 7      ;LSB MASK
(1) 002347      001      .BYTE 1      ;NUMBER OF LOCATIONS
(1)              002350      GETVEC= .      ; POINTER FOR FALCON TWEAKER.  ;:GPA
(1) 002350      104403      INSTR      ;INPUT ADDRESS OF DEVICE VECTOR
(1) 002352      007162      MVECTOR   ;MESSAGE 'VECTOR ADDRESS-'
(1) 002354      104405      PARAM      ;CONVERT STRING TO OCTAL
(1) 002356      000300      300      ;LOW LIMIT
(1) 002360      000770      770      ;HIGH LIMIT
(1) 002362      002040      DZVRIV   ;LOCATIONS TO BE FILLED
(1) 002364      003      .BYTE 3      ;LSB MASK
(1) 002365      004      .BYTE 4      ;NUMBER OF LOCATIONS
(1) 002366      004737 007764 JSR      PC,DZVLEV ;GO BUILD DEVICE POINTERS
(1)
(1) 002372      104403      INSTR      ;INPUT WHICH TEST YOU ARE RUNNING
(1) 002374      007371      MWHICH   ;ECHO OR CABLE
(1) 002376      104416      PAUCH     ;SET FLAG
(1) 002400      007126      WCHFL   ;THIS FLAG
(1) 002402      104403      BAUD: INSTR ;INPUT BAUD RATE
(1) 002404      007312      MSPEED  ;MESSAGE 'BAUD RATE-'
(1) 002406      104415      PARM    ;CONVERT DECIMAL STRING TO OCTAL
(1) 002410      000062      50      ;LOW LIMIT
(1) 002412      022600      9600   ;HIGH LIMIT
(1) 002414      007144      LINESP  ;LOCATION TO BE FILLED
(1) 002416      000      .BYTE 0      ;LSB MASK
(1) 002417      001      .BYTE 1      ;NUMBER OF LOCATIONS
(1) 002420      104413      LINEX: DEVICE.CLR ;CLEAR DEVICE
(1) 002422      005037 007130 CLR      STFLG  ;CLEAR PROGRAM START FLAG
(1) 002426      104403      INSTR      ;INPUT LINE NUMBER
(1) 002430      007302      MLINE   ;MESSAGE 'LINE NUMBER-'
(1) 002432      104405      PARAM   ;CONVERT STRING TO OCTAL
(1) 002434      000000      0      ;LOW LIMIT
(1) 002436      000003      3      ;HIGH LIMIT
(1) 002440      001374      SAVLIN  ;LOCATION TO BE FILLED
(1) 002442      000      .BYTE 0      ;LSB MASK
(1) 002443      001      .BYTE 1      ;NUMBER OF LOCATIONS
(1) 002444      004537 006732 JSR      R5,SET
(1)
(1) 002450      106427 000200 XBEGIN: MTPS  #MASK ;LOCK OUT INTERRUPTS
(1) 002454      012706 001120 MOV      #STACK,SP ;SET UP PROCESSOR STACK
(1) 002460      005037 007132 CLR      LOCKUP ;CLEAR TIMEOUT
(1) 002464      005737 007126 TST     WCHFLG ;ECHO OR CABLE TEST ?
(1) 002470      001413      BEQ     2$      ;ECHO
(1) 002472      012737 010500 001252 MOV     #TST2,$LPADR ;CABLE TEST
(1) 002500      005737 007130 TST     STFLG  ;ARE YOU LOOPING ?
(1) 002504      001017      BNE     1$      ;YES
(1) 002506      005137 007130 COM     STFLG  ;NO
(1) 002512      104402 007464 TYPE   ,MCABLE ;TYPE CABLE TEST
(1) 002516      000412      BR      1$      ;
(1) 002520      012737 010124 001252 2$: MOV     #TST1,$LPADR ;SET UP ECHO TEST
(1) 002526      005737 007130 TST     STFLG  ;ARE YOU LOOPING ?
(1) 002532      001004      BNE     1$      ;YES
(1) 002534      005137 007130 COM     STFLG  ;NO
(1) 002540      104402 007437 TYPE   ,MTERM  ;TYPE ECHO TEST
(1) 002544      000177 176502 1$: RESTART:JMP @SLPADR ;START TESTING,THIS LOCATION IS ALSO

```

CVDZC-B MACY11 30G(1063) 10-AUG-81 11:15
CVDZCB.P11 10-AUG-81 10:56

PAGE 21-20
PROGRAM INITIALIZATION AND START UP.

SEQ 0034

(1)
2146

::GPA PRGEND DZV11,<END PASS DVDZC-A >,10. :USED BY THE POWER UP ROUTINE

2147
(2)
(2)
(2)
(2)
(3)
(3)
(4)
(3)
(3)
(3)
(3)
(3)
(3)
(5) 002550
(5) 002550 005037 001262
(5) 002554 105037 001247
(5) 002560 104402 006117
(5) 002564 104402 006301
(5) 002570 104412 002712
(5) 002574 104402 006307
(5) 002600 104412 002720
(5) 002604 005237 001126
(5) 002610 104402 006315
(5) 002614 104412 002726
(5) 002620 005337 001126
(5) 002624 104402 006326
(5) 002630 104412 002734
(3) 002634 005037 001354
(3) 002640 005237 001126
(3) 002644 042737 100000 001126
(3) 002652 005327
(3) 002654 000001
(3) 002656 003013
(3) 002660 012737
(3) 002662 000001
(3) 002664 002654
(3) 002666 013700 000042
(3) 002672 001405
(3) 002674 000005
(3) 002676 004710
(3) 002700 000240
(3) 002702 000240
(3) 002704 000240
(3) 002706
(3) 002706 000137
(3) 002710 002450
(2)
(2) 002712 000001
(2) 002714 006 002
(2) 002716 002010
(2) 002720 000001
(2) 002722 003 002
(2) 002724 002040
(2) 002726 000001
(2) 002730 006 002
(2) 002732 001126
(2) 002734 000001

```

:END OF PASS
:TYPE NAME OF TEST
:UPDATE PASS COUNT
:CHECK FOR EXIT TO ACT-11
:RESTART TEST
.SBTTL END OF PASS ROUTINE

:*****
:*INCREMENT THE PASS NUMBER ($PASS)
:*IF THERES A MONITOR GO TO IT
:*IF THERE ISN'T JUMP TO XBEGIN

$EOP:
CLR $ERRPC ;(CLEAR LAST ERROR PC
CLRB $ERFLG ;CLEAR ERROR FLAG
TYPE ,MEPASS ;TYPE END PASS
TYPE ,MCSRX ;TYPE CSR
CNVRT ,XCSR ;SHOW IT
TYPE ,MVECX ;TYPE VECTOR
CNVRT ,XVEC ;SHOW IT
INC $PASS ;RAISE PASS COUNT
TYPE ,MPASSX ;TYPE PASSES
CNVRT ,XPASS ;SHOW IT
DEC $PASS ;RESTORE PASS COUNT
TYPE ,MERRX ;TYPE ERRORS
CNVRT ,XERR ;SHOW IT
CLR $TIMES ;ZERO THE NUMBER OF ITERATIONS
INC $PASS ;INCREMENT THE PASS NUMBER
BIC #100000,$PASS ;DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;LOOP?

$EOPCT: .WORD 1 ;YES
BGT $DOAGN ;RESTORE COUNTER
MOV (PC)+,@(PC)+

$SENDCT: .WORD 1

$GET42: MOV @#42,R0 ;GET MONITOR ADDRESS
BEQ $DOAGN ;BRANCH IF NO MONITOR
RESET ;CLEAR THE WORLD
$SENDAD: JSR PC,(R0) ;GO TO MONITOR
NOP ;SAVE ROOM
NOP ;FOR
NOP ;ACT11

$DOAGN: JMP @ (PC)+ ;RETURN
$RTNAD: .WORD XBEGIN

XCSR: 1
.BYTE 6,2
DZVCSR

XVEC: 1
.BYTE 3,2
DZVRIV

XPASS: 1
.BYTE 6,2
$PASS

XERR: 1
```

(2)	002736	006	002	.BYTE	6,2	
(2)	002740	001256		\$ERTTL		
(1)						
(1)						
(1)	002742	011605		: CONVERT DECIMAL ASCII STRING TO OCTAL		
(1)	002744	012537	003126	.PARMD:	MOV	(SP),R5
(1)	002750	012537	003130		MOV	(R5)+,6\$
(1)	002754	012537	003132		MOV	(R5)+,7\$
(1)	002760	112537	003134		MOV	(R5)+,8\$
(1)	002764	112537	003135		MOVB	(R5)+,9\$
(1)	002770	010516			MOVB	(R5)+,10\$
(1)	002772	005005			MOV	R5,(SP)
(1)	002774	012704	007616	2\$:	CLR	R5
(1)	003000	122714	000015		MOV	#INBUF,R4
(1)	003004	001424			CMPB	#15,(R4)
(1)	003006	121427	000060	1\$:	BEQ	3\$
(1)	003012	002421			CMPB	(R4),#'0
(1)	003014	121427	000071		BLT	3\$
(1)	003020	003016			CMPB	(R4),#'9
(1)	003022	142714	000060		BGT	3\$
(1)	003026	005002			BICB	#'0,(R4)
(1)	003030	152402			CLR	R2
(1)	003032	060205			BISB	(R4)+,R2
(1)	003034	122714	000015		ADD	R2,R5
(1)	003040	001410			CMPB	#15,(R4)
(1)	003042	006305			BEQ	4\$
(1)	003044	010502			ASL	R5 ;X2
(1)	003046	006305			MOV	R5,R2 ;SAVE X2
(1)	003050	006305			ASL	R5 ;X4
(1)	003052	060205			ASL	R5 ;X8
(1)	003054	000754			ADD	R2,R5 ;TIMES 10
(1)	003056	104404		3\$:	BR	1\$
(1)	003060	000744			INSTR	
(1)					BR	2\$
(1)						
(1)					: TEST TO SEE IF NUMBER IS WITHIN LIMITS	
(1)	003062	020537	003130	4\$:	CMP	R5,7\$
(1)	003066	101373			BHI	3\$
(1)	003070	020537	003126		CMP	R5,6\$
(1)	003074	103770			BLO	3\$
(1)	003076	133705	003134		BITB	9\$,R5
(1)	003102	001365			BNE	3\$
(1)						
(1)					: STORE NUMBER AT SPECIFIED ADDRESS	
(1)	003104	013704	003132		MOV	8\$,R4
(1)	003110	010524		5\$:	MOV	R5,(R4)+
(1)	003112	062705	000002		ADD	#2,R5
(1)	003116	105337	003135		DECB	10\$
(1)	003122	001372			BNE	5\$
(1)	003124	000002			RTI	
(1)	003126	000000		6\$:	0	
(1)	003130	000000		7\$:	0	
(1)	003132	000000		8\$:	0	
(1)	003134	000		9\$:	.BYTE 0	
(1)	003135	000		10\$:	.BYTE 0	

```
(1)
(1) ;CHECK FOR FREEZE ON CURRENT DATA
(1) ;-----
(1)
(1) 003136 032777 001000 176140 .SCOP1: BIT #SW09,@SWR ;IS SW09=1(SET)?
(1) 003144 001405 BEQ 1$ ;BR IF NOT SET.
(1) 003146 005737 001364 TST LOCK ;IS THERE A TIGHT LOOP SPECIFIED?
(1) 003152 001402 BEQ 1$ ;IF NO, RETURN
(1) 003154 013716 001364 MOV LOCK,(SP) ;IF YES, GOTO THE ADDRESS IN LOCK.
(1) 003160 000002 1$: RTI ;GO BACK.
(1)
(1) 003162 032777 010000 176114 .TYPE: BIT #SW12,@SWR ;INHIBIT ALL PRINTOUT??
(1) 003170 001403 BEQ $TYPE ;IF NOT, GO TYPE
(1) 003172 062716 000002 ADD #2,(SP) ;SKIP OVER MESSAGE POINTER
(1) 003176 000002 RTI ;RETURN TO WHERE PROCEDURE WAS INVOKED
(2) .SBTTL TYPE ROUTINE
(2)
(2) ;*****
(2) ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(2) ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(2) ;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(2) ;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(2) ;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(2) ;*
(2) ;*CALL:
(2) ;*1) USING A TRAP INSTRUCTION
(2) ;* TYPE ,MESADR ;:MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(2) ;*OR
(2) ;* TYPE
(2) ;* MESADR
(2) ;*
(2)
(2) 003200 105737 001323 $TYPE: TSTB $TPFLG ;:IS THERE A TERMINAL?
(2) 003204 100002 BPL 1$ ;:BR IF YES
(2) 003206 000000 HALT ;:HALT HERE IF NO TERMINAL
(2) 003210 000430 BR 3$ ;:LEAVE
(2) 003212 010046 1$: MOV R0,-(SP) ;:SAVE R0
(2) 003214 017600 000002 MOV @2(SP),R0 ;:GET ADDRESS OF ASCIZ STRING
(2) 003220 122737 000001 001140 CMPB #APTENV,$ENV ;:RUNNING IN APT MODE
(2) 003226 001011 BNE 62$ ;:NO,GO CHECK FOR APT CONSOLE
(2) 003230 132737 000100 001141 BITB #APTSPOOL,$ENVM ;:SPOOL MESSAGE TO APT
(2) 003236 001405 BEQ 62$ ;:NO,GO CHECK FOR CONSOLE
(2) 003240 010037 003250 MOV R0,61$ ;:SETUP MESSAGE ADDRESS FOR APT
(2) 003244 004737 003542 JSR PC,$ATY3 ;:SPOOL MESSAGE TO APT
(2) 003250 000000 61$: .WORD 0 ;:MESSAGE ADDRESS
(2) 003252 132737 000100 001141 62$: BITB #APTCSUP,$ENVM ;:APT CONSOLE SUPPRESSED
(2) 003260 001003 BNE 60$ ;:YES,SKIP TYPE OUT
(2) 003262 112046 2$: MOVB (R0)+,-(SP) ;:PUSH CHARACTER TO BE TYPED ONTO STACK
(2) 003264 001005 BNE 4$ ;:BR IF IT ISN'T THE TERMINATOR
(2) 003266 005726 TST (SP)+ ;:IF TERMINATOR POP IT OFF THE STACK
(2) 003270 012600 60$: MOV (SP)+,R0 ;:RESTORE R0
(2) 003272 062716 000002 3$: ADD #2,(SP) ;:ADJUST RETURN PC
(2) 003276 000002 RTI ;:RETURN
(2) 003300 122716 000011 4$: CMPB #HT,(SP) ;:BRANCH IF <HT>
(2) 003304 001430 BEQ 8$
(2) 003306 122716 000200 CMPB #CRLF,(SP) ;:BRANCH IF NOT <CRLF>
```

```
(2) 003312 001006 BNE 5$  
(2) 003314 005726 TST (SP)+ ;:POP <CR><LF> EQUIV  
(2) 003316 104402 TYPE ;:TYPE A CR AND LF  
(2) 003320 001357 $CRLF  
(2) 003322 105037 003530 CLRB $CHARCNT ;:CLEAR CHARACTER COUNT  
(2) 003326 000755 BR 2$ ;:GET NEXT CHARACTER  
(2) 003330 004737 003412 5$: JSR PC,$TYPEC ;:GO TYPE THIS CHARACTER  
(2) 003334 123726 001322 6$: CMPB $FILLC,(SP)+ ;:IS IT TIME FOR FILLER CHARS.?  
(2) 003340 001350 BNE 2$ ;:IF NO GO GET NEXT CHAR.  
(2) 003342 013746 001320 MOV $NULL,-(SP) ;:GET # OF FILLER CHARS. NEEDED  
(2) ;:AND THE NULL CHAR.  
(2) 003346 105366 000001 7$: DECB 1(SP) ;:DOES A NULL NEED TO BE TYPED?  
(2) 003352 002770 BLT 6$ ;:BR IF NO--GO POP THE NULL OFF OF STACK  
(2) 003354 004737 003412 JSR PC,$TYPEC ;:GO TYPE A NULL  
(2) 003360 105337 003530 DECB $CHARCNT ;:DO NOT COUNT AS A COUNT  
(2) 003364 000770 BR 7$ ;:LOOP  
(2)  
(2) ;:HORIZONTAL TAB PROCESSOR  
(2)  
(2) 003366 112716 000040 8$: MOVB #' (SP) ;:REPLACE TAB WITH SPACE  
(2) 003372 004737 003412 9$: JSR PC,$TYPEC ;:TYPE A SPACE  
(2) 003376 132737 000007 003530 BITB #7,$CHARCNT ;:BRANCH IF NOT AT  
(2) 003404 001372 BNE 9$ ;:TAB STOP  
(2) 003406 005726 TST (SP)+ ;:POP SPACE OFF STACK  
(2) 003410 000724 BR 2$ ;:GET NEXT CHARACTER  
(2) 003412 $TYPEC:  
(2) 003412 105777 175672 TSTB @STKS ;:CHAR IN KYBD BUFFER? ;:MJD001  
(2) 003416 100022 BPL 10$ ;:BR IF NOT ;:MJD001  
(2) 003420 017746 175666 MOV @STKB,-(SP) ;:GET CHAR ;:MJD001  
(2) 003424 042716 177600 BIC #177600,(SP) ;:STRIP EXTRANEIOUS BITS ;:MJD001  
(2) 003430 122716 000023 CMPB #$XOFF,(SP) ;:WAS CHAR XOFF ;:MJD001  
(2) 003434 001012 BNE 102$ ;:BR IF NOT ;:MJD001  
(2) 003436 105777 175646 101$: TSTB @STKS ;:WAIT FOR CHAR ;:MJD001  
(2) 003442 100375 BPL 101$ ;:MJD001  
(2) 003444 117716 175642 MOVB @STKB,(SP) ;:GET CHAR ;:MJD001  
(2) 003450 042716 177600 BIC #177600,(SP) ;:STRIP IT ;:MJD001  
(2) 003454 122716 000021 CMPB #$XON,(SP) ;:WAS IT XON? ;:MJD001  
(2) 003460 001366 BNE 101$ ;:BR IF NOT ;:MJD001  
(2) 003462 102$: TST (SP)+ ;:FIX STACK ;:MJD001  
(2) 003464 10$: ;:MJD001  
(2) 003464 105777 175624 TSTB @STPS ;:WAIT UNTIL PRINTER IS READY ;:MJD001  
(2) 003470 100375 BPL 10$ ;:MJD001  
(2) 003472 116677 000002 175616 MOVB 2(SP),@STPB ;:LOAD CHAR TO BE TYPED INTO DATA REG.  
(2) 003500 122766 000015 000002 CMPB #CR,2(SP) ;:IS CHARACTER A CARRIAGE RETURN?  
(2) 003506 001003 BNE 1$ ;:BRANCH IF NO  
(2) 003510 105037 003530 CLRB $CHARCNT ;:YES--CLEAR CHARACTER COUNT  
(2) 003514 000406 BR $TYPEX ;:EXIT  
(2) 003516 122766 000012 000002 1$: CMPB #LF,2(SP) ;:IS CHARACTER A LINE FEED?  
(2) 003524 001402 BEQ $TYPEX ;:BRANCH IF YES  
(2) 003526 105227 INCB (PC)+ ;:COUNT THE CHARACTER  
(2) 003530 000000 $CHARCNT: .WORD 0 ;:CHARACTER COUNT STORAGE  
(2) 003532 000207 $TYPEX: RTS PC  
(2)  
(2) .SBTTL APT COMMUNICATIONS ROUTINE
```

```

(2)
(3)
(2) 003534 112737 000001 004000 *****
(2) 003542 112737 000001 003776 SATY1:  MOVB  #1,$FFLG  ::TO REPORT FATAL ERROR
(2) 003550 000403 SATY3:  MOVB  #1,$MFLG  ::TO TYPE A MESSAGE
(2) 003552 112737 000001 004000 BR      SATYC
(2) 003560 SATY4:  MOVB  #1,$FFLG  ::TO ONLY REPORT FATAL ERROR
(4) 003560 010046 SATYC:
(4) 003562 010146 MOV     R0,-(SP)  ::PUSH R0 ON STACK
(2) 003564 105737 003776 MOV     R1,-(SP)  ::PUSH R1 ON STACK
(2) 003570 001450 TSTB   $MFLG     ::SHOULD TYPE A MESSAGE?
(2) 003572 122737 000001 001140 BEQ     5$        ::IF NOT: BR
(2) 003600 001031 CMPB   #APTENV,$ENV  ::OPERATING UNDER APT?
(2) 003602 132737 000100 001141 BNE     3$        ::IF NOT: BR
(2) 003610 001425 BITB   #APTPOOL,$ENVM ::SHOULD SPOOL MESSAGES?
(2) 003612 017600 000004 BEQ     3$        ::IF NOT: BR
(2) 003616 062766 000002 000004 MOV     @4(SP),R0  ::GET MESSAGE ADDR.
(2) 003624 005737 001120 ADD     #2,4(SP)  ::BUMP RETURN ADDR.
(2) 003630 001375 1$: TST    $MSGTYPE  ::SEE IF DONE W/ LAST XMISSION?
(2) 003632 010037 001134 BNE     1$        ::IF NOT: WAIT
(2) 003636 105720 2$: MOV     R0,$MSGAD  ::PUT ADDR IN MAILBOX
(2) 003640 001376 (R0)+  ::FIND END OF MESSAGE
(2) 003642 163700 001134 BNE     2$
(2) 003646 006200 SUB     $MSGAD,R0  ::SUB START OF MESSAGE
(2) 003650 010037 001136 ASR     R0        ::GET MESSAGE LNGTH IN WORDS
(2) 003654 012737 000004 001120 MOV     R0,$MSGGLT  ::PUT LENGTH IN MAILBOX
(2) 003662 000413 MOV     #4,$MSGTYPE  ::TELL APT TO TAKE MSG.
(2) 003664 017637 000004 003710 BR      5$
(2) 003672 062766 000002 000004 MOV     @4(SP),4$  ::PUT MSG ADDR IN JSR LINKAGE
(4) 003700 013746 177776 ADD     #2,4(SP)  ::BUMP RETURN ADDRESS
(2) 003704 004737 003200 MOV     177776,-(SP)  ::PUSH 177776 ON STACK
(2) 003710 000000 JSR     PC,$TYPE   ::CALL TYPE MACRO
(2) 003712 105737 004000 4$: .WORD 0
(2) 003716 001416 5$:
(2) 003720 005737 001140 10$: TSTB   $FFLG     ::SHOULD REPORT FATAL ERROR?
(2) 003724 001413 BEQ     12$      ::IF NOT: BR
(2) 003726 005737 001120 TST    $ENV      ::RUNNING UNDER APT?
(2) 003732 001375 BEQ     12$      ::IF NOT: BR
(2) 003734 017637 000004 001122 11$: TST    $MSGTYPE  ::FINISHED LAST MESSAGE?
(2) 003742 062766 000002 000004 BNE     11$     ::IF NOT: WAIT
(2) 003750 005237 001120 MOV     @4(SP),$FATAL  ::GET ERROR #
(2) 003754 105037 004000 ADD     #2,4(SP)  ::BUMP RETURN ADDR.
(2) 003760 105037 003777 12$: INC     $MSGTYPE  ::TELL APT TO TAKE ERROR
(2) 003764 105037 003776 CLRB   $FFLG     ::CLEAR FATAL FLAG
(4) 003770 012601 CLRB   $LFLG     ::CLEAR LOG FLAG
(4) 003772 012600 CLRB   $MFLG     ::CLEAR MESSAGE FLAG
(2) 003774 000207 MOV     (SP)+,R1  ::POP STACK INTO R1
(2) 003776 000 MOV     (SP)+,R0  ::POP STACK INTO R0
(2) 003777 000 RTS     PC        ::RETURN
(2) 004000 000 $MFLG: .BYTE 0  ::MESSG. FLAG
(2) 004002 000 $LFLG: .BYTE 0  ::LOG FLAG
(2) 000200 000 $FFLG: .BYTE 0  ::FATAL FLAG
(2) 000001 APTSIZE=200
(2) 000100 APTENV=001
(2) 000040 APTPOOL=100
(2) APTCSUP=040

```



```

(1)
(1)                                     ;STRING INPUT ROUTINE
(1)                                     :-----
(1)
(1) 004002 010346 .INSTR: MOV R3,-(SP) ;SAVE R3 ON STACK
(1) 004004 010446 MOV R4,-(SP) ;SAVE R4 ON STACK
(1) 004006 017637 000004 004024 MOV @4(SP),.MSG ;GET THE ADDRESS OF THE MESSAGE TO BE PRINTED
(1) 004014 062766 000002 000004 ADD #2,4(SP) ;POINT TO INSTRUCTION AFTER ADDRESS POINTER
(1) 004022 104402 .INST1: TYPE ;PRINT THE MESSAGE
(1) 004024 000000 .MSG: 0 ;MESSAGE IS POINTED TO FROM HERE
(1) 004026 012704 007616 MOV #INBUF,R4 ;POINT R4 TO THE INPUT BUFFER
(1) 004032 012703 000007 MOV #7,R3 ;SET THE MAXIMUM NUMBER OF CHARACTERS ALLOWED
(1) 004036 105777 175246 1$: TSTB @STKS ;HAS A CHARACTER BEEN RECEIVED?
(1) 004042 100375 BPL 1$ ;IF NO, KEEP WAITING FOR IT
(1) 004044 117714 175242 MOVB @STKB,(R4) ;IF YES, SAVE IT IN THE INPUT BUFFER
(1) 004050 142714 000200 BICB #200,(R4) ;KEEP ONLY THE 7-BIT ASCII INFORMATION
(1) 004054 122427 000015 CMPB (R4)+,#15 ;IS THIS CHARACTER A LINE FEED?
(1) 004060 001417 BEJ INSTR2 ;IF SO, TERMINATE THE INPUT SEQUENCE
(1) 004062 105777 175226 2$: TSTB @STPS ;IF NOT, CHECK TO SEE IF THE CHARACTER CAN PRINT
(1) 004066 100375 BPL 2$ ;IF WE CAN'T, WAIT UNTIL WE CAN
(1) 004070 017777 175216 175220 MOV @STKB,@STPB ;ECHO THE CHARACTER BACK
(1) 004076 005303 DEC R3 ;REDUCE THE NUMBER OF CHARACTERS RECEIVED
(1) 004100 001356 BNE 1$ ;IF WE DON'T HAVE 7, GO GET SOME MORE
(1) 004102 012604 MOV (SP)+,R4 ;IF WE HAVE 7, RESTORE R4
(1) 004104 012603 MOV (SP)+,R3 ;RESTORE R3
(1) 004106 010346 .INSTE: MOV R3,-(SP) ;SAVE R3 ON THE STACK
(1) 004110 010446 MOV R4,-(SP) ;SAVE R4 ON THE STACK
(1) 004112 104402 001356 TYPE ,SQUES ;PRINT A QUESTION MARK... WHAT'S GOING ON?
(1) 004116 000741 BR .INST1 ;GO PRINT THE MESSAGE AGAIN
(1) 004120 012604 INSTR2: MOV (SP)+,R4 ;RESTORE R4
(1) 004122 012603 MOV (SP)+,R3 ;RESTORE R3
(1) 004124 000002 RTI ;RETURN TO THE MAIN PROCEDURE
    
```

```

(1)
(1)                                     ;CONVERT ASCII STRING TO OCTAL
(1)                                     :-----
(1)
(1) 004126 010546 .PARAM: MOV R5,-(SP) ;SAVE R5 ON THE STACK
(1) 004130 010446 MOV R4,-(SP) ;SAVE R4 ON THE STACK
(1) 004132 016605 000004 MOV 4(SP),R5 ;GET THE SETUP INFORMATION POINTER
(1) 004136 012537 004316 MOV (R5)+,LOLIM ;SET THE LOW LIMIT FOR THE INPUT
(1) 004142 012537 004320 MOV (R5)+,HILIM ;SET THE HIGH LIMIT FOR THE INPUT
(1) 004146 012537 004322 MOV (R5)+,DEVADR ;SAVE THE ADDRESS WHERE THE RESULT WILL BE STORED
(1) 004152 112537 004324 MOV (R5)+,LOBITS ;GET THE MASK OF THE INCORRECT BITS
(1) 004156 112537 004325 MOV (R5)+,ADRCNT ;GET THE COUNT OF ITEMS TO BE STORED
(1) 004162 010566 000004 MOV R5,4(SP) ;POINT TO WHERE MAIN LINE PROGRAM WILL RESUME
(1) 004166 005005 PARAM1: CLR R5 ;INITIALIZE THE ASCII TO OCTAL RESULT WORD
(1) 004170 012704 007616 MOV #INBUF,R4 ;POINT TO THE INPUT BUFFER
(1) 004174 122714 000015 CMPB #15,(R4) ;IS THIS CHARACTER A CARRIAGE RETURN?
(1) 004200 001420 BEQ PARERR ;IF SO, PRINT THE MESSAGE AGAIN
(1) 004202 121427 000060 1$: CMPB (R4),#60 ;IS THIS CHARACTER BELOW THE NUMERIC RANGE?
(1) 004206 002415 BLT PARERR ;IF SO, GO PRINT THE MESSAGE AGAIN
(1) 004210 121427 000067 CMPB (R4),#67 ;IS THIS CHARACTER ABOVE THE NUMERIC RANGE?
(1) 004214 003012 BGT PARERR ;IF SO, GO PRINT THE MESSAGE AGAIN
(1) 004216 142714 000060 BICB #60,(R4) ;ISOLATE THE NUMBER THE CHARACTER REPRESENTS
(1) 004222 152405 BISB (R4)+,R5 ;CONCATENATE THESE BITS TO THE ALREADY EXISTING STRING
(1) 004224 122714 000015 CMPB #15,(R4) ;IS THE NEXT CHARACTER A CARRIAGE RETURN?
    
```

```

(1) 004230 001406      BEQ     LIMITS      ;IF SO, GO SEE IF NUMBER IS WITHIN LIMITS
(1) 004232 006305      ASL     R5           ;CLEAR BIT POSITION 0, MOVE EXISTING STRING TO LEFT
(1) 004234 006305      ASL     R5           ;CLEAR POSITION 1, MOVE STRING TO LEFT AGAIN
(1) 004236 006305      ASL     R5           ;MOVE THE STRING ONE MORE TIME TO MAKE ROOM FOR
(1)                                ;NEXT THREE BITS
(1) 004240 000760      BR      1$          ;GO GET THE NEXT CHARACTER
(1) 004242 104404      PARERR: INSTER    ;THERE WAS AN ERROR... GO PRINT MESSAGE AGAIN
(1) 004244 000750      BR      PARAM1     ;TRY GETTING THE PARAMETERS AGAIN
(1)
(1)                                ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
(1)                                ;-----
(1)
(1) 004246 020537 004320  LIMITS: CMP     R5,HILIM    ;DOES RESULT EXCEED ITS MAXIMUM CORRECT VALUE?
(1) 004252 101373      BHI     PARERR      ;IF YES, GO PRINT THE MESSAGE AGAIN
(1) 004254 020537 004316  CMP     R5,LOLIM    ;IS THE RESULT LOWER THAN ALLOWED?
(1) 004260 103770      BLO     PARERR      ;IF YES, GO PRINT THE MESSAGE AGAIN
(1) 004262 133705 004324  BITB   LOBITS,R5   ;ARE ANY INCORRECT BITS SET IN THE RESULT?
(1) 004266 001365      BNE     PARERR      ;IF SO, GO PRINT THE MESSAGE AGAIN
(1)
(1)                                ;STORE NUMBER AT SPECIFIED ADDRESS
(1)
(1) 004270 013704 004322  1$:   MOV     DEVADR,R4   ;POINT TO THE LOCATION WHERE THE RESULT WILL BE STORED
(1) 004274 010524      MOV     R5,(R4)+    ;STORE THE RESULT
(1) 004276 062705 000002  ADD     #2,R5       ;CALCULATE THE NEXT DATUM
(1) 004302 105337 004325  DECB   ADRCNT      ;REDUCE COUNT OF STORED RESULTS. IS IT EXCEEDED?
(1) 004306 001372      BNE     1$         ;IF NOT, GO STORE THE NEXT DATUM
(1) 004310 012604      MOV     (SP)+,R4    ;RESTORE R4
(1) 004312 012605      MOV     (SP)+,R5    ;RESTORE R5
(1) 004314 000002      RTI                    ;RETURN TO THE MAIN PROGRAM
(1)
(1) 004316 000000      LOLIM: 0             ;LOWEST ACCEPTABLE VALUE
(1) 004320 000000      HILIM: 0           ;HIGHEST ACCEPTABLE
(1) 004322 000000      DEVADR: 0         ;LOCATION WHERE RESULT WILL BE STORED
(1) 004324 000      LOBITS: .BYTE 0      ;INCORRECT BITS MASK
(1) 004325 000      ADRCNT: .BYTE 0    ;COUNT OF ITEMS TO BE STORED
(1)
(1)                                ;SAVE PC OF TEST THAT FAILED AND R0-R5
(1)                                ;-----
(1)
(1) 004326 016637 000004 001404 .SAV05: MOV     4(SP),SAVPC   ;SAVE R7 (PC)
(1)
(1)                                ;SAVE R0-R5
(1)
(1) 004334 010537 001340  SV05: MOV     R5,$REG5   ;SAVE R5
(1) 004340 010437 001336  MOV     R4,$REG4   ;SAVE R4
(1) 004344 010337 001334  MOV     R3,$REG3   ;SAVE R3
(1) 004350 010237 001332  MOV     R2,$REG2   ;SAVE R2
(1) 004354 010137 001330  MOV     R1,$REG1   ;SAVE R1
(1) 004360 010037 001326  MOV     R0,$REG0   ;SAVE R0
(1) 004364 000002      RTI                    ;LEAVE.
(1)
(1)                                ;RESTORE R0-R5
(1)
(1) 004366 013700 001326  .RES05: MOV     $REG0,R0 ;RESTORE R0
(1) 004372 013701 001330  MOV     $REG1,R1    ;RESTORE R1
(1) 004376 013702 001332  MOV     $REG2,R2    ;RESTORE R2

```

```

(1) 004402 013703 001334      MOV      $REG3,R3      ;RESTORE R3
(1) 004406 013704 001336      MOV      $REG4,R4      ;RESTORE R4
(1) 004412 013705 001340      MOV      $REG5,R5      ;RESTORE R5
(1) 004416 000002              RTI                    ;LEAVE
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1) 004420 104402 001357      .CONVR: TYPE          ,SCLF          ;PRINT A CARRIAGE RETURN
(1) 004424 010046              .CNVRT: MOV      R0,-(SP)      ;SAVE R0
(1) 004426 010146              MOV      R1,-(SP)      ;SAVE R1
(1) 004430 010346              MOV      R3,-(SP)      ;SAVE R3
(1) 004432 010446              MOV      R4,-(SP)      ;SAVE R4
(1) 004434 010546              MOV      R5,-(SP)      ;SAVE R5
(1) 004436 017601 000012      MOV      @12(SP),R1     ;PLACE THE ADDRESS OF THE ARGUMENTS IN R1
(1) 004442 062766 000002 000012  ADD      #2,12(SP)      ;POINT TO WHERE MAIN PROGRAM WILL RESUME
(1) 004450 012137 004574      MOV      (R1)+,WRDCNT   ;GET NUMBER OF WORDS TO BE PRINTED
(1) 004454 112105              1$:  MOV      (R1)+,R5     ;GET THE NUMBER OF CHARACTERS TO BE PRINTED
(1) 004456 112100              MOV      (R1)+,R0     ;GET THE NUMBER OF SPACES TO PRINT
(1) 004460 013104              MOV      @ (R1)+,R4    ;COPY THE WORD TO BE CONVERTED
(1) 004462 110537 004576      MOV      R5,CHRCNT     ;COPY THE CHARACTER COUNT
(1) 004466 010403              3$:  MOV      R4,R3       ;COPY THE ARGUMENT WORD AGAIN
(1) 004470 042703 177770      BIC      #^C<7>,R3     ;ISOLATE THREE BITS TO BE TREATED AS A CHARACTER
(1) 004474 062703 000060      ADD      #060,R3      ;MAKE AN ASCII CHARACTER OUT OF THEM
(1) 004500 110346              MOV      R3,-(SP)     ;SAVE THAT CHARACTER
(1) 004502 006004              ROR      R4            ;MOVE THE NEXT THREE BITS INTO PLACE
(1) 004504 006204              ASR      R4            ;MOVE THEM AGAIN
(1) 004506 006204              ASR      R4            ;AND FINALLY A THIRD TIME
(1) 004510 005305              DEC      R5            ;REDUCE CHARACTER COUNT. ARE ALL CHARACTERS
(1)
(1)
(1) 004512 001365              BNE      3$           ;BUILT?
(1) 004514 012703 007722      MOV      #MDATA,R3     ;IF NO, GO BUILD THE NEXT ONE.
(1) 004520 112623              4$:  MOV      (SP)+,(R3)+  ;NOW POINT TO WHERE NUMBER WILL BE PRINTED FROM
(1) 004522 105337 004576      DECB    CHRCNT        ;STORE THE CHARACTER, STARTING WITH THE MOST
(1) 004526 001374              BNE      4$           ;REDUCE COUNT. ARE ALL CHARACTERS TRANSFERRED?
(1) 004530 105700              TSTB    R0            ;IF NO, GO TRANSFER ANOTHER
(1) 004532 001404              BEQ     6$           ;ARE ANY SPACES TO BE PRINTED?
(1) 004534 112723 000040      5$:  MOV      #040,(R3)+  ;IF NO, DON'T SET UP ANY
(1) 004540 105300              DECB    R0            ;ADD A SPACE TO THE OUTPUT BUFFER
(1) 004542 001374              BNE      5$           ;REDUCE THE COUNT. SHOULD WE PRINT MORE?
(1) 004544 105013              6$:  CLRB    (R3)        ;IF YES, GO ADD ANOTHER SPACE
(1) 004546 104402 007722      TYPE    ,MDATA        ;TERMINATE THE OUTPUT BUFFER WITH A ZERO
(1) 004552 005337 004574      DEC     WRDCNT        ;PRINT THE STRING WE JUST BUILT
(1) 004556 001336              BNE     1$           ;REDUCE THE WORD COUNT. ARE ANY MORE WORDS LEFT?
(1) 004560 012605              MOV     (SP)+,R5     ;IF YES, GO CONVERT THEM
(1) 004562 012604              MOV     (SP)+,R4     ;RESTORE R5
(1) 004564 012603              MOV     (SP)+,R3     ;RESTORE R4
(1) 004566 012601              MOV     (SP)+,R3     ;RESTORE R3
(1) 004570 012600              MOV     (SP)+,R1     ;RESTORE R1
(1) 004572 000002              MOV     (SP)+,R0     ;RESTORE R0
(1) 004574 000000              RTI                    ;RETURN TO THE MAIN PROGRAM
(1)
(1) 004576 000          WRDCNT: 0
(1) 004577 000          CHRCNT: .BYTE
(1)
(1)
(1) 004600 000000          SPACNT: .BYTE 0
(1)
(1)
(1)
(1)
(1) 004600 000000          BINWRD: 0
(1)

```



```

(1)                                     ;LINE PARAMETER REGISTER SETUP ROUTINE
(1)
(1) 004726 010146 .LPRSET:MOV R1,-(SP) ;SAVE CONTENTS OF R1
(1) 004730 010246 MOV R2,-(SP) ;SAVE CONTENTS OF R2
(1) 004732 013701 001370 MOV PAR,R1 ;MOVE DEFAULT PARAM. INTO R1
(1) 004736 012702 000001 MOV #1,R2 ;INIT. FOR LINE 1
(1) 004742 010177 175052 1$: MOV R1,@DZVLPR ;LOAD PARAM. REGISTER
(1) 004746 005201 INC R1 ;SET R1 FOR NEXT LINE
(1) 004750 106302 ASLB R2 ;SET R2 FOR NEXT LINE
(1) 004752 032702 000020 BIT #BIT4,R2 ;ALL LINES DONE?
(1) 004756 001771 BEQ 1$ ;IF NO LOAD NEXT LINE
(1) 004760 012602 MOV (SP)+,R2 ;RELOAD R2
(1) 004762 012601 MOV (SP)+,R1 ;RELOAD R1
(1) 004764 000002 RTI ;RETURN

(1)                                     ;ROUTINE TO ZERO DATA BUFFER
(1)
(1) 004766 010046 .BUFSET:MOV R0,-(SP) ;SAVE CONTENTS OF R0
(1) 004770 012700 001426 MOV #TDO,R0 ;SET R0 TO TOP OF BUFFER
(1) 004774 005020 1$: CLR (R0)+ ;CLEAR BUFFER LOCATION
(1) 004776 022700 001446 CMP #STOP,R0 ;IS BUFFER ALL CLEARED
(1) 005002 001374 BNE 1$ ;IF NOT CLEAR NEXT LOCATION
(1) 005004 012600 MOV (SP)+,R0 ;RELOAD R0
(1) 005006 000002 RTI ;RETURN

(1)                                     ;ERROR HANDLER
(1) :-----
(1)
(1) 005010 004737 005436 $ERROR: JSR PC,SERV.G ;FIND OUT IF <^G> WAS HIT
(1) 005014 032777 010000 174262 BIT #SW12,@SWR ;BELL ON ERROR?
(1) 005022 001406 BEQ XBX ;BR IF NO BELL
(1) 005024 105777 174264 TSTB @STPS ;TTY READY.
(1) 005030 100003 BPL XBX ;DON'T WAIT IF TTY NOT READY.
(1) 005032 112777 000207 174256 MOVB #207,@STPB ;PUSH A BELL AT THE TTY.
(1) 005040 032777 020000 174236 XBX: BIT #SW13,@SWR ;DELETE ERROR PRINT OUT?
(1) 005046 001113 BNE HALTS ;BR IF NO PRINT OUT WANTED.
(1) 005050 021637 001262 CMP (SP),$ERRPC ;WAS THIS ERROR FOUND LAST TIME?
(1) 005054 001404 BEQ 1$ ;BR IF YES
(1) 005056 011637 001262 MOV (SP),$ERRPC ;RECORD BEING HERE
(1) 005062 105037 001247 CLRB $ERFLG ;PREPARE HEADER
(1) 005066 104407 1$: SAVO5 ;SAVE ALL PROC REGISTERS
(1) 005070 011605 MOV (SP),R5 ;GET THE PC OF ERROR
(1) 005072 162705 000002 SUB #2,R5 ;GET ADDRESS OF TRAP CALL
(1) 005076 011504 MOV (R5),R4 ;GET ERROR INSTRUCTION
(1) 005100 110437 001260 MOVB R4,$ITEMB ;COPY TEST NUMBER FOR APT HANDLING
(1) 005104 006304 ASL R4 ;MULT BY TWO
(1) 005106 061504 ADD (R5),R4 ;DOUBLE IT
(1) 005110 006304 ASL R4 ;MULT AGAIN
(1) 005112 042704 177001 BIC #177001,R4 ;CLEAR JUNK
(1) 005116 062704 011170 ADD #.ERRTAB,R4 ;GET POINTER
(1) 005122 012437 005246 MOV (R4)+,ERRMSG ;GET ERROR MESSAGE
(1) 005126 012437 005260 MOV (R4)+,DATAHD ;GET DATA HEADRER
(1) 005132 011437 005272 MOV (R4),DATABP ;GET DATA TABLE
(1) 005136 105737 001247 TSTB $ERFLG ;TYPE HEADER
(1) 005142 001403 BEQ TYPMSG ;BR IF YES
(1) 005144 005737 005272 TST DATABP ;DOES DATA TABLE EXIST?

```

(1)	005150	001044			BNE	TYPDAT		:BR IF YES.
(1)	005152	104402	001357		TYPMSG: TYPE	,SCRLF		:TYPE A CARRIAGE RETURN
(1)	005156	104402	001357		TYPE	,SCRLF		:AND TYPE ANOTHER
(1)	005162	005737	001364		TST	LOCK		
(1)	005166	001402			BEQ	1\$		
(1)	005170	104402	006351		TYPE	,MASTEK		
(1)	005174	104402	006337	1\$:	TYPE	,MTSTN		
(1)	005200	104412	005430		CNVRT	,XTSTN		:SHOW IT
(1)	005204	104402	006427		TYPE	,MERRPC		:TYPE PC.
(1)	005210	104412	005422		CNVRT	,ERTABO		:SHOW IT
(1)	005214	104402	006301		TYPE	,MCSRX		
(1)	005220	104412	002712		CNVRT	,XCSR		
(1)	005224	104402	001357		TYPE	,SCRLF		:GIVE A CR/LF
(1)	005230	112737	177777	001247	MOVB	#-1,\$ERFLG		:NO MORE HEADER UNLESS NO DATA TABLE.
(1)	005236	005737	005246		TST	ERRMSG		:IS THERE AN ERROR MESSAGE?
(1)	005242	001402			BEQ	WTBS.FM		:BR IF NO.
(1)	005244	104402			TYPE			:TYPE
(1)	005246	000000			ERRMSG: 0			: ERROR MESSAGE
(1)	005250				WTBS.FM:			
(1)	005250	005737	005260		TST	DATAHD		:DATA HEADER?
(1)	005254	001402			BEQ	TYPDAT		:BR IF NO
(1)	005256	104402			TYPE			:TYPE
(1)	005260	000000			DATAHD: 0			: DATA HEADER
(1)	005262	005737	005272		TYPDAT: TST	DATABP		:DATA TABLE?
(1)	005266	001402			BEQ	RESREG		:BR IF NO.
(1)	005270	104411			CNVRT			:SHOW
(1)	005272	000000			DATABP: 0			: DATA TABLE
(1)	005274	104410			RESREG: RES05			:RESTORE PROC REGISTERS
(1)	005276	122737	000001	001140	HALTS: CMPB	#APTENV,\$ENV		:IS APT RUNNING?
(1)	005304	001007			BNE	1\$:SKIP APT CALL IF NOT
(1)	005306	113737	001260	005320	MOVB	\$ITEMB,\$\$:COPY ERROR NUMBER
(1)	005314	004737	003552		JSR	PC,\$ATY4		:CALL APT SERVICE
(1)	005320	000000			5\$:	.WORD	0	:ERROR NUMBER STUCK HERE
(1)	005322	000777			10\$:	BR	10\$:LOCK UP HERE
(1)	005324	022737	002676	000042	15\$:	CMP	#SENDAD,@#42	:CHECK TO SEE IF IN ACT-11 MODE
(1)	005332	001403			BEQ	20\$:IF SO, HANDLE ACCORDINGLY
(1)	005334	005777	173744		TST	@SWR		:HALT ON ERROR?
(1)	005340	100004			BPL	EXITER		:BR IF NO HALT ON ERROR
(1)	005342	016677	000002	173736	20\$:	MOV	2(SP),@DISPLAY	:SHOW ERROR PC IN DATA DISPLAY
(1)	005350	000000			HALT			:HALT
(1)	005352	005237	001256		EXITER: INC	\$ERTTL		:UPDATE ERROR COUNT
(1)	005356	004737	005436		JSR	PC,\$SERV.G		:FIND OUT IF ^G WAS TYPED
(1)	005362	032777	000400	173714	BIT	#SW08,@SWR		:GOTO TOP OF TEST?
(1)	005370	001007			BNE	1\$:BR IF YES
(1)	005372	032777	002000	173704	BIT	#SW10,@SWR		:GOTO NEXT TEST?
(1)	005400	001407			BEQ	2\$:BR IF NO
(1)	005402	013737	001362	001252	MOV	NEXT,\$LPADR		:SET FOR NEXT TEST
(1)	005410	012706	001120		1\$:	MOV	#STACK,SP	:RESET SP
(1)	005414	000177	173632		JMP	@\$LPADR		:GOTO SPECIFIED TEST
(1)	005420	000002			2\$:	RTI		:RETURN
(1)	005422	000001			ERTABO: 1			
(1)	005424	006	002		.BYTE	6,2		
(1)	005426	001404			SAVPC			
(1)	005430	000001			XTSTN: 1			
(1)	005432	002	002		.BYTE	2,2		
(1)	005434	001246			\$TSTNM			

APT COMMUNICATIONS ROUTINE

```
005436 017746 173650 SERV.G: MOV @STKB,-(SP) ;OTHERWISE, GET THE LAST CHARACTER TYPED
005442 042716 000200 BIC #BIT7,(SP) ;STRIP PARITY(EIGHTH) BIT
(1) 005446 122726 000007 CMPB #7,(SP)+ ;IS IT ^G?
(1) 005452 001076 BNE 6$ ;IF NOT, IGNORE INPUT
(1) 005454 032777 004000 173626 BIT #4000,@STKS ;RX BUSY?
(1) 005462 001365 BNE SERV.G ;BR IF YES
(1) 005464 005464 GETSWR= ;:GPA
(1) 005464 017737 173614 005672 MOV @SWR,90$ ;SAVE (SWR).
(1) 005472 104402 005652 1$: TYPE ,89$ ;TYPE HEADER FOR OLD SWITCH REGISTER
(1) 005476 104412 005664 CNVRT ,88$ ;TYPE THE NUMBER ITSELF
(1) 005502 104402 005674 TYPE ,91$ ;AFTER HAVING CONVERTED IT TO ASCII
(1) 005506 105037 005700 CLRB 92$ ;CLEAR SWR CHANGE FLAG
(1) 005512 005077 173566 CLR @SWR ;CLEAR THE SOFTWARE SWITCH REGISTER
(1) 005516 105777 173566 3$: TSTB @STKS ;WAIT FOR DONE.
(1) 005522 100375 BPL 3$ ;CONTINUE WAITING FOR IT
(1) 005524 017746 173562 MOV @STKB,-(SP) ;PUT THE CHARACTER ON THE STACK
(1) 005530 042716 000200 BIC #BIT7,(SP) ;STRIP PARITY BIT
(1) 005534 122726 000015 CMPB #15,(SP)+ ;IS IT THE CARRIAGE RETURN CHAR?
(1) 005540 001433 BEQ 4$ ;IF SO, GO PRINT CRLF
(1) 005542 105777 173546 2$: TSTB @STPS ;IS THE OUTPUT BUFFER AVAILABLE
(1) 005546 100375 BPL 2$ ;IF NOT, WAIT FOR IT TO BE READY
(1) 005550 105237 005700 INCB 92$ ;INDICATE THAT THE SWR WAS CHANGED
(1) 005554 014677 173536 MOV -(SP),@STPB ;PLACE THE CHARACTER THERE(ECHO BACK)
(1) 005560 000241 CLC ;GET READY TO ROTATE
(1) 005562 003177 173516 ROL @SWR ;MOVE THE EXISTING BITS OVER
(1) 005566 006177 173512 ROL @SWR ;TO MAKE ROOM FOR THE INCOMING
(1) 005572 006177 173506 ROL @SWR ;THREE BITS FROM THIS CHARACTER
(1) 005576 103735 BCS 1$ ;ERROR
(1) 005600 022627 000060 CMP (SP)+,#60 ;IS IT LOWER THAN 0?
(1) 005604 002732 BLT 1$ ;IF SO, GO ASK AGAIN
(1) 005606 026627 177776 000067 CMP -2(SP),#67 ;IS IT HIGHER THAN 7?
(1) 005614 003326 BGT 1$ ;IF SO, GO ASK AGAIN
(1) 005616 042746 177770 BIC #^C<7>,-(SP) ;ISOLATE INFORMATION BITS
(1) 005622 052677 173456 BIS (SP)+,@SWR ;ADD THEM TO THE SWITCH REGISTER
(1) 005626 000733 BR 3$ ;GO CHECK FOR THE NEXT CHARACTER
(1) 005630 105737 005700 4$: TSTB 92$ ;HAS THE SWR BEEN CHANGED?
(1) 005634 001003 BNE 5$ ;IF YES GO TYPE CRLF
(1) 005636 013777 005672 173440 MOV 90$,@SWR ;IF NOT RESTORE SWR
(1) 005644 104402 001357 5$: TYPE ,$CRLF ;TYPE A CARRIAGE RETURN AND LINE FEED
(1) 005650 000207 6$: RTS PC ;RETURN TO CALLING PROCEDURE
(1) 005652 020200 051450 051127 89$: .ASCIZ <200>? (SWR)=/?
(1) .EVEN
(1) 005664 000001 88$: 1
(1) 005666 006 000 .BYTE 6,0
(1) 005670 005672 90$: .WORD 0
(1) 005672 000000 91$: .ASCIZ ?/=/?
(1) 005674 036457 000057 92$: .BYTE 0
(1) 005700 000 .EVEN
(2) .SBTTL POWER DOWN AND UP ROUTINES
(2)
(3)
(2)
(2) 005702 012737 006046 000024 :*****
(2) 005710 012737 000340 000026 :POWER DOWN ROUTINE
$PWRDN: MOV #SILLUP,@#PWRVEC ;;SET FOR FAST UP
MOV #340,@#PWRVEC+2 ;;PRIO:7
```

```

(4) 005716 010046      MOV      R0,-(SP)          ;;PUSH R0 ON STACK
(4) 005720 010146      MOV      R1,-(SP)          ;;PUSH R1 ON STACK
(4) 005722 010246      MOV      R2,-(SP)          ;;PUSH R2 ON STACK
(4) 005724 010346      MOV      R3,-(SP)          ;;PUSH R3 ON STACK
(4) 005726 010446      MOV      R4,-(SP)          ;;PUSH R4 ON STACK
(4) 005730 010546      MOV      R5,-(SP)          ;;PUSH R5 ON STACK
(4) 005732 017746 173346  MOV      @SWR,-(SP)        ;;PUSH @SWR ON STACK
(2) 005736 010637 006052  MOV      SP,$SAVR6        ;;SAVE SP
(2) 005742 012737 005754 000024  MOV      #SPWRUP,@#PWRVEC ;;SET UP VECTOR
(2) 005750 000000      HALT
(2) 005752 000776      BR      -2                ;;HANG UP
(2)
(3)
(2)
(2) 005754 012737 006046 000024  $PWRUP: MOV      #SILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(2) 005762 013706 006052      MOV      $SAVR6,SP        ;;GET SP
(2) 005766 005037 006052      CLR      $SAVR6          ;;WAIT LOOP FOR THE TTY
(2) 005772 005237 006052 1$:      INC      $SAVR6          ;;WAIT FOR THE INC
(2) 005776 001375      BNE     1$                ;;OF WORD
(4) 006000 012677 173300      MOV      (SP)+,@SWR        ;;POP STACK INTO @SWR
(4) 006004 012605      MOV      (SP)+,R5         ;;POP STACK INTO R5
(4) 006006 012604      MOV      (SP)+,R4         ;;POP STACK INTO R4
(4) 006010 012603      MOV      (SP)+,R3         ;;POP STACK INTO R3
(4) 006012 012602      MOV      (SP)+,R2         ;;POP STACK INTO R2
(4) 006014 012601      MOV      (SP)+,R1         ;;POP STACK INTO R1
(4) 006016 012600      MOV      (SP)+,R0         ;;POP STACK INTO R0
(2) 006020 012737 005702 000024  MOV      #SPWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(2) 006026 012737 000340 000026  MOV      #340,@#PWRVEC+2  ;;PRIO:7
(2) 006034 104402      TYPE
(2) 006036 006054      SPWRMG: .WORD  MPFAIL      ;;REPORT THE POWER FAILURE
(2) 006040 012716      MOV      (PC)+,(SP)        ;;POWER FAIL MESSAGE POINTER
(2) 006042 002544      SPWRAD: .WORD  RESTART     ;;RESTART AT RESTART
(2) 006044 000002      RTI
(2) 006046 000000      $SILLUP: HALT             ;;THE POWER UP SEQUENCE WAS STARTED
(2) 006050 000776      BR      -2                ;; BEFORE THE POWER DOWN WAS COMPLETE
(2) 006052 000000      $SAVR6: 0                 ;;PUT THE SP HERE
(2) 006054 050200 051127 043040  MPFAIL: .ASCIZ <200>/PWR FAILED. RESTART AT LAST TEST /
(2) 006117 200 047105 020104  MEMPASS: .ASCIZ <200>/END PASS CVDZC-B /
(2) 006143 200 052522 047116  MR:      .ASCIZ <200>/RUNNING /
(2) 006157 200 051120 043517  MERR2:   .ASCIZ <200>/PROGRAM INDICATES NO DEVICES PRESENT./
(2) 006226 044600 051516 043125  MERR3:   .ASCIZ <200>/INSUFFICIENT DATA!/
(2) 006252 046200 041517 020113  MLOCK:   .ASCIZ <200>/LOCK ON SELECTED TEST/
(2) 006301 103 051123 020072  MCSRX:   .ASCIZ /CSR: /
(2) 006307 126 041505 020072  MVECX:   .ASCIZ /VEC: /
(2) 006315 120 051501 042523  MPASSX:  .ASCIZ /PASSES: /
(2) 006326 051105 047522 051522  MERRX:   .ASCIZ /ERRORS: /
(2) 006337 124 051505 020124  MTSTN:   .ASCIZ /TEST NO: /
(2) 006351 052 000040      MASTEK:  .ASCIZ /* /
(2) 006354 051600 052105 051440  MNEW:    .ASCIZ <200>/SET SWITCH REG TO DZV11'S DESIRED ACTIVE./
(2) 006427 120 035103 000040  MERRPC:  .ASCIZ /PC: /
(2) 006434 046600 050101 047440  XHEAD:   .ASCIZ <200>/MAP OF DZV11 STATUS/<200>
(2) 006462 044600 046114 043505  MBADLN:  .ASCIZ <200>/ILLEGAL ENTRY IN STAGGERED MODE/<200>
(2)
(2) 006524 000002      .EVEN
(2) 006526 006 003      XSTATQ: 2
(2) 006530 001344      .BYTE 6,3
          $TMP1
    
```


(2) 006532 006 002 .BYTE 6.2
(2) 006534 001346 \$TMP2
(1) .EVEN

```

(2) ;THIS ROUTINE ESTABLISHES WHICH MAINTENANCE MODE THE DEVICE IS IN
(2) ;-----
(2) ;E=EXTERNAL LOOP BACK
(2) ;I=INTERNAL LOOP BACK
(2) ;S=STAGGERED LOOP BACK
(2) 006536 017605 000000 .SETFLG:MOV @ (SP),R5 ;PICK UP ADDRESS OF TAG
(2) 006542 042737 000040 007616 BIC #40,INBUF ;STRIP LOWER CASE
(2) 006550 122737 000105 007616 CMPB #'E,INBUF ;IS IT EXTERNAL LOOP BACK ?
(2) 006556 001005 BNE 4$ ;NO
(2) 006560 013715 006650 MOV 1$, (R5) ;YES STORE INFO
(2) 006564 105037 001424 CLRB MNTFLG ;SET MAINT BIT =0
(2) 006570 000422 BR 7$ ;GET OUT
(2) 006572 122737 000111 007616 4$: CMPB #'I,INBUF ;IS IT INTERNAL LOOP BACK ?
(2) 006600 001006 BNE 5$ ;NO
(2) 006602 013715 006652 MOV 2$, (R5) ;YES STORE INFO
(2) 006606 112737 000010 001424 MOVB #MAINT,MNTFLG ;SET UP THE MAINTENANCE FLAG LOADER
(2) 006614 000410 BR 7$ ;GET OUT
(2) 006616 122737 000123 007616 5$: CMPB #'S,INBUF ;IS IT STAGGERED LOOP BACK ?
(2) 006624 001007 BNE 6$ ;WHAT ?
(2) 006626 013715 006654 MOV 3$, (R5) ;YES STORE INFO
(2) 006632 105037 001424 CLRB MNTFLG ;ZERO BITS
(2) 006636 062716 000002 7$: ADD #2, (SP) ;POP AROUND
(2) 006642 000002 RTI
(2) 006644 104404 6$: INSTER ;RETRY
(2) 006646 000733 BR .SETFLG ;DITTO
(2) 006650 000200 1$: .WORD 200 ;EXTERNAL = E
(2) 006652 000000 2$: .WORD 0 ;INTERNAL = I
(2) 006654 100000 3$: .WORD 100000 ;STAGGERED = S
(2)
(2) ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
(2) ;BUFFER TO THE CHARACTERS 'E' AND 'C'.
(2) ;IF THE CHARACTER IS 'E' CLEAR THE FLAG
(2) ;IF THE CHARACTER IS 'C' SET THE FLAG
(2)
(2) 006656 017605 000000 .PAWCH:MOV @ (SP),R5
(2) 006662 142737 000040 007616 BICB #40,INBUF ;SET FOR LOWER CASE INPUT
(2) 006670 122737 000105 007616 CMPB #'E,INBUF ;IS IT 'E' ?
(2) 006676 001002 BNE 1$
(2) 006700 105015 CLRB (R5) ;000
(2) 006702 000406 BR 2$
(2) 006704 122737 000103 007616 1$: CMPB #'C,INBUF ;IS IT 'C' ?
(2) 006712 001005 BNE 3$
(2) 006714 112715 177777 MOVB #-1, (R5) ;3177
(2) 006720 062716 000002 2$: ADD #2, (SP)
(2) 006724 000002 RTI
(2) 006726 104404 3$: INSTER ;RETRY
(2) 006730 000752 BR .PAWCH
    
```

```

(2) ;THIS ROUTINE CONVERTS LINE SPEED (LINESP) AND
(2) ;LINE NUMBER (SAVLIN) FOR DZVLPR, DZVTCR AND DZVCSR
(2) ;REGISTER USAGE.
(2)
(2) 006732 013737 001374 007150 SET: MOV SAVLIN,NUMLIN ;SAVE SAVLIN
(2) 006740 013700 001374 MOV SAVLIN,R0 ;COPY THE LINE NUMBER FOR LOOP CONTROL
(2) 006744 005037 007152 CLR NUMTCR ;SET A DEFAULT OF LINE 0 OR NO LINES
(2) 006750 012702 000001 MOV #1,R2 ;SET A BIT POINTER TO THE FIRST LINE
(2) 006754 005300 XTCR1: DEC R0 ;REDUCE THE INDICATOR.IS IT MINUS YET?
(2) 006756 100402 BMI SET1 ;IF SO, R2 POINTS TO THE RIGHT LINE
(2) 006760 006302 ASL R2 ;IF NOT, MOVE THE POINTER TO THE NEXT LINE
(2) 006762 000774 BR XTCR1 ;GO SEE IF THIS LINE IS THE ONE
(2) 006764 012701 007026 SET1: MOV #TABLE2,R1
(2) 006770 010237 007152 MOV R2,NUMTCR ;COPY THE CORRECT BIT POINTER
(2) 006774 022137 007144 1$: CMP (R1)+,LINESP
(2) 007000 001407 BEQ 2$
(2) 007002 005721 TST (R1)+ ;IS IT THE END OF TABLE?
(2) 007004 001373 BNE 1$ ;NO
(2) 007006 104402 007254 TYPE ,MINVAL ;INVALID BAUD RATE,BEGIN AGAIN
(2) 007012 012705 002402 MOV #BAUD,R5 ;JUMP TO BAUD THRU R5
(2) 007016 000402 BR 3$
(2) 007020 011137 007146 2$: MOV (R1),SPEED ;SET UP BAUD RATE
(2) 007024 000205 3$: RTS R5

```

```

(2)
(2)
(2)
(2) ;THE FOLLOWING IS A TABLE OF LEGAL BAUD RATES (8 BITS/CHAR)
(2) TABLE2: .WORD 50. ;50 BAUD
(2) .WORD 10070 ;
(2) .WORD 75. ;75 BAUD
(2) .WORD 10470 ;
(2) .WORD 110. ;110 BAUD
(2) .WORD 11070 ;TWO STOP BITS
(2) .WORD 135. ;134.5 BAUD
(2) .WORD 11470 ;TWO STOP BITS
(2) .WORD 150. ;150 BAUD
(2) .WORD 12070 ;TWO STOP BITS
(2) .WORD 300. ;300 BAUD
(2) .WORD 12430 ;ONE STOP BIT
(2) .WORD 600. ;600 BAUD
(2) .WORD 13030 ;ONE STOP BIT
(2) .WORD 1200. ;1200 BAUD
(2) .WORD 13430 ;ONE STOP BIT
(2) .WORD 1800. ;1800 BAUD
(2) .WORD 14030 ;ONE STOP BIT
(2) .WORD 2000. ;2000 BAUD
(2) .WORD 14430 ;ONE STOP BIT
(2) .WORD 2400. ;2400 BAUD
(2) .WORD 15030 ;ONE STOP BIT
(2) .WORD 3600. ;3600 BAUD
(2) .WORD 15430 ;ONE STOP BIT
(2) .WORD 4800. ;4800 BAUD
(2) .WORD 16030 ;ONE STOP BIT
(2) .WORD 7200. ;7200 BAUD
(2) .WORD 16430 ;ONE STOP BIT
(2) .WORD 9600. ;9600 BAUD

```

```
(2) 007120 017030 .WORD 17030  
(2) 007122 177777 000000 .WORD -1,0 ;TABLE TERMINATOR  
(2)  
(2) 007126 000000 WCHFLG:0 ;ECHO OR CABLE FLAG  
(2) 007130 000000 STFLG: 0 ;PROGRAM START FLAG  
(2) 007132 000000 LOCKUP: 0 ;TIMEOUT FLAG  
(2) 007134 000000 LAST: 0 ;LAST ERROR PC  
(2) 007136 000000 TDATA: 0  
(2) 007140 000000 RDATA: 0  
(2) 007142 000000 BYTCNT: 0  
(2) 007144 000156 LINESP: 110 ;DEFAULT BAUD RATE  
(2) 007146 011070 SPEED: 11070 ;DEFAULT 110 BAUD, 8 BITS/CHAR,  
;FDX, 2 STOP BITS  
(2) 007150 000000 NUMLIN: 0  
(2)  
(2) 007152 000001 NUMTCR: 1 ;DEFAULT VALUE, TCR BIT 0  
(2) 007154 000200 PRIO: 200 ;DEFAULT DEVICE PRIORITY  
;MASK OUT INTERRUPTS  
(2)  
(2) 007156 000000 RECDAT: 0  
(2) 007160 000000 TBUF: 0  
(2) 007162 053200 041505 047524 MVECTO: .ASCIZ <200>/VECTOR ADDRESS- /  
(2) 007204 041600 047117 051124 MREGAD: .ASCIZ <200>/CONTROL REGISTER ADDRESS- /  
(2) 007240 050200 051501 020123 MPASS: .ASCIZ <200>/PASS DONE./  
(2) 007254 044600 053116 046101 MINVAL: .ASCIZ <200>/INVALID BAUD RATE - /  
(2) 007302 046200 047111 035105 MLINE: .ASCIZ <200>/LINE: /  
(2) 007312 041200 052501 020104 MSPEED: .ASCIZ <200>/BAUD RATE - /  
(2) 007330 052200 050131 020105 MCHAR: .ASCIZ <200>/TYPE A CHAR. ON DZV11 TERMINAL /  
(2) 007371 200 044127 041511 MWHICH: .ASCIZ <200>/WHICH TEST ? ECHO OR CABLE (E OR C) /  
(2) 007437 200 042524 046522 MTERM: .ASCIZ <200>/TERMINAL ECHO TEST /  
(2) 007464 041600 041101 042514 MCABLE: .ASCIZ <200>/CABLE TEST /  
(2) 007501 377 177415 005377 MQUICK: .ASCII <377><15><377><377><12><377><377>  
(2) 007510 044124 020105 052521 .ASCII /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/  
(2) 007605 377 177415 005377 .ASCII <377><15><377><377><12><377><377><377><0>  
(2)  
(2) .EVEN  
(2)  
(2) ;BUFFERS FOR INPUT-OUTPUT  
(2)  
(2) 007616 000000 INBUF: 0  
(2) 007660 000000 .=. +40  
(2) 007722 000000 TEMP: 0  
(2) 007764 000000 .=. +40  
(2) 007722 000000 MDATA: 0  
(2) 007764 000000 .=. +40
```

```

(1)                                     ;THIS UTILITY SETS UP CSR'S,SETS UP VECTORS.
(1) 007764 013700 002040 DZVLEV: MOV DZVRIV,RO ;PLACE THE BASE VECTOR ADDRESS IN RO
(1) 007770 062700 000002 ADD #2,RO ;CALCULATE THE RECEIVER INTERRUPT STATUS ADDR.
(1) 007774 010037 002042 MOV RO,DZVRIS ;STORE IT HERE
(1) 010000 062700 000002 ADD #2,RO ;CALCULATE THE TRANSMITTER INTERRUPT VECTOR
(1) 010004 010037 002044 MOV RO,DZVTIV ;STORE IT HERE
(1) 010010 062700 000002 ADD #2,RO ;CALCULATE THE TRANSMITTER VECTOR STATUS ADDRESS
(1) 010014 010037 002046 MOV RO,DZVTIS ;STORE IT HERE

(1)                                     ;THIS SEGMENT SETS UP POINTERS FOR THE GIVEN DZV11. $BASE IS THE BASE ADDRESS
(1)                                     ;OF THE DEVICE
(1) 010020 013700 001174 MOV $BASE,RO ;COPY THE ADDRESS BEING LOADED
(1) 010024 010037 002010 MOV RO,DZVCSR ;XXX0
(1) 010030 005200 INC RO
(1) 010032 010037 002012 MOV RO,HDZVCSR ;XXX1
(1) 010036 005200 INC RO
(1) 010040 010037 002014 MOV RO,DZVRBUF ;XXX2
(1) 010044 010037 002020 MOV RO,DZVLPR ;XXX2
(1) 010050 005200 INC RO
(1) 010052 010037 002016 MOV RO,HDZVRBUF ;XXX3
(1) 010056 010037 002022 MOV RO,HDZVLPR ;XXX3
(1) 010062 005200 INC RO
(1) 010064 010037 002024 MOV RO,DZVTCR ;XXX4
(1) 010070 005200 INC RO
(1) 010072 010037 002026 MOV RO,HDZVTCR ;XXX5
(1) 010076 005200 INC RO
(1) 010100 010037 002030 MOV RO,DZVMSR ;XXX6
(1) 010104 010037 002034 MOV RO,DZVTDR ;XXX6
(1) 010110 005200 INC RO
(1) 010112 010037 002032 MOV RO,HDZVMSR ;XXX7
(1) 010116 010037 002036 MOV RO,HDZVTDR ;XXX7
(1) 010122 000207 RTS PC

```

```
2151
2152
2153
2154
2155
2156
2157 010124 104413
2158 010126 012737 000001 001246
2159 010134 013777 007152 171662
2160 010142 013737 007150 001370
2161 010150 053737 007146 001370
2162 010156 013777 001370 171634
2163 010164 012777 000040 171616
2164 010172 005004
2165 010174 012705 007501
2166 010200 005777 171604
2167 010204 100404
2168 010206 104414
2169 010210 005304
2170 010212 001372
2171 010214 104003
2172 010216 005004
2173 010220 112577 171610
2174 010224 001365
2175 010226 004737 005436
2176 010232 122777 000377 171044
2177 010240 001755
2178 010242 012737 002550 001362
2179 010250 012777 010324 171562
2180 010256 012777 000200 171556
2181 010264 106427 000000
2182 010270 012777 000140 171512
2183 010276 104402 007330
2184 010302 105777 171002
2185 010306 100375
2186 010310 106427 000200
2187 010314 004737 005436
2188 010320 000137 002420
2189
2190
2191
2192 010324 105777 171460
2193 010330 100401
2194 010332 104004
2195 010334 017737 171454 007156
2196 010342 100401
2197 010344 104023
2198 010346 032737 020000 007156
2199 010354 001401
2200 010356 104025
2201
2202 010360 113737 007156 007160
2203 010366 113737 007156 007616
2204 010374 042737 177600 007616
2205 010402 042737 176377 007156
2206 010410 000337 007156

:***** ECHO TEST *****
:*THIS TEST WILL ACCEPT 1 CHARACTER AT A TIME
:*(IN INTERRUPT MODE) AND TRANSMIT THAT SAME CHARACTER,
:*ONE LINE AT A TIME, ANY LINE 0 THRU 7 (OCTAL)

TST1: DEVICE.CLR :CLEAR DZV11
MOV #1,$STSTM
MOV NUMTCR,@DZVTCR :SET TCR BIT
MOV NUMLIN,PAR :SET PARAMETERS
BIS SPEED,PAR :SET BAUD RATE
MOV PAR,@DZVLPR :LOAD PARAM.
MOV #MSENAB,@DZVCSR :SET SCANN ENABLE
CLR R4
4$: MOV #MOUICK,R5 :SET MESSAGE BUFFER
3$: TST @DZVCSR :TRDY?
BMI 2$ :BR IF YES
DELAY :WAIT
DEC R4
BNE 3$
ERROR 3 :NO TRDY SET! WHY?
2$: CLR R4 :RESET COUNTER TO 0
MOVB (R5)+,@DZVTDR :LOAD CHAR
BNE 3$
JSR PC,SERV.G :<^G>?
CMPB #377,@SWR :SWR SET TO 377?
BEQ 4$ :IF YES LOOP ON QUICK MESSAGE
MOV #SEOP,NEXT
MOV #INTSVC,@DZVRIV :SET UP INTERRUPT SERVICE
MOV #MASK,@DZVRIS :AND LEVEL
MTPS #CLEAR :ALLOW INTERRUPTS
MOV #RIE!MSENAB,@DZVCSR :SET RECEIVER INTERRUPT ENABLE
TYPE ,MCHAR :TYPE "ANY CHARACTER"
1$: TSTB @STKS :IF SOMEBODY HITS A KEY- GET NEW LINE #
BPL 1$ :LOOP HERE
MTPS #MASK :MASK FURTHER INTERRUPTS
JSR PC,SERV.G :MAKE SURE IT WASN'T <^G>
JMP LINEX

INTSVC: :THE FOLLOWING IS THE RECEIVER INTERRUPT SVC ROUTINE
TSTB @DZVCSR :TEST REC. FLAG
BMI .+4
ERROR 4 :ERROR - INTERRUPT NOT CAUSED BY FLAG
MOV @DZVRBUF,RECDAT
BMI .+4
ERROR 23 :NON- VALID CHARACTER
BIT #BIT13,RECDAT :CHECK FOR FRAMING ERROR
BEQ .+4 :BR IF NO ERROR
ERROR 25 :EITHER SOMEBODY HIT THE
:"BREAK KEY" OR YOU HAVE AN ERROR!
MOVB RECDAT,TBUF :MOVE CHARACTER TO OUTPUT AREA
MOVB RECDAT,INBUF :MOVE CHARACTER TO CHECK FOR ^C
BIC #^C<177>,INBUF :STRIP JUNK PLUS PARITY
BIC #176377,RECDAT :SAVE ONLY LINE NUMBER
SWAB RECDAT
```

```

2207 010414 023737 001374 007156      CMP    SAVLIN,RECDAT    ;DOES THE LINE # COMPARE?
2208 010422 001407                      BEQ    2$
2209 010424 013737 007156 001374      MOV    RECDAT,SAVLIN    ;ADJUST LINE NO. FOR ERROR
2210 010432 104015                      ERROR  15                ;*WRONG LINE NUMBER
2211 010434 013737 007150 001374      MCV    NUMLIN,SAVLIN    ;CORRECT LINE NO. INDICATOR
2212 010442 123727 007616 000003 2$:  CMPB   INBUF,#3          ;IS IT A ^C ?
2213 010450 001004                      BNE    1$                ;NO
2214 010452 104413                      DEVICE.CLR
2215 010454 012716 002550              MOV    #SEOP,(SP)       ;CRUNCH STACK
2216 010460 000002                      RTI
2217 010462 005777 171322              1$:  TST    @DZVCSR          ;TRDY SET
2218 010466 100375                      BPL    1$                ;IF NOT THEN WAIT
2219 010470 113777 007160 171336      MOVB   TBUF,@DZVTDR     ;TRANSMIT THE CHARACTER
2220 010476 000002                      RTI
2221
2222
2223                                     :***** CABLE TEST *****
2224                                     :*THIS TEST TRANSMITS A BINARY COUNT PATTERN
2225                                     :*VIA INTERRUPT MODE TO THE RECEIVER
2226                                     :*...THE LINE UNDER TEST MUST BE TERMINATED WITH THE TEST CONNECTOR
2227
2228 010500 106427 000200              TST2: MTPS   #MASK          ;DISABLE INTERRUPTS
2229 010504 012737 000002 001246      MOV    #2,$STSTM
2230 010512 012737 002550 001362      MOV    #SEOP,NEXT
2231 010520 104413                      DEVICE.CLR
2232                                     :*TEST TO VERIFY THAT SETTING DTR FOR A GIVEN LINE
2233                                     :*WILL BRING UP "CO" AND "RING" FOR THE SAME LINE
2234                                     :*JUMPERS W1,W2,W3 AND W4 MUST BE INSTALLED ON THE
2235                                     :*INTERFACE MODULE OTHERWISE AN ERROR REPORT WILL RESULT.
2236 010522 012737 010530 001364      MOV    #1$,LOCK         ;LOOP
2237 010530 113777 007152 171270 1$:  MOVB   NUMTCR,@HDZVTCR  ;SET DTR
2238 010536 005005                      CLR    R5
2239 010540 153705 007152              BISB   NUMTCR,R5        ;BUILD EXPECTED
2240 010544 000305                      SWAB   R5                ;PUT IN HIGH BYTE
2241 010546 153705 007152              BISB   NUMTCR,R5
2242 010552 104414                      DELAY
2243 010554 017704 171250              MOV    @DZVMSR,R4       ;WAIT FOR CABLE DELAY
2244 010560 020504                      CMP    R5,R4             ;READY MODEM BITS
2245 010562 001401                      BEQ    2$                ;ARE THEY OK?
2246 010564 104022                      ERROR  22                ;BR IF YES
2247                                     ;IS THE TEST CONNECTOR ON?
2248                                     ;HAS RIGHT LINE BEEN SELECTED?
2249                                     ;IF SO- YOU HAVE A PROBLEM!
2250 010566 104401              2$:  SCOP1
2251 010570 104413              3$:  DEVICE.CLR
2252 010572 005037 001364              CLR    LOCK              ;CLEAR SCOP1 LOCK ADDRESS
2253 010576 013737 007146 001370      MOV    SPEED,PAR        ;SET LINE SPEED
2254 010604 053737 007150 001370      BIS    NUMLIN,PAR       ;SELECT LINE #
2255 010612 052737 010000 001370      BIS    #RCVON,PAR       ;ENABLE THE RECEIVER FOR THIS LINE
2256 010620 013777 001370 171172      MOV    PAR,@DZVLPR      ;SET THE PARAMETERS AND TURN ON RECEIVER
2257 010626 012777 010750 171204      MOV    #INTRC,@DZVRIV   ;SET UP INTR SERVICE
2258 010634 012777 000200 171200      MOV    #MASK,@DZVRIS    ;SET UP LEVEL
2259 010642 012777 011140 171174      MOV    #INTRAN,@DZVTIV  ;SET UP INTR SERVICE
2260 010650 012777 000200 171170      MOV    #MASK,@DZVTIS    ;SET UP LEVEL
2261 010656 012777 040140 171124      MOV    #TIE!RIE!MSENAB,@DZVCSR ;SET TRANSMITTER INTERRUPT ENABLE
2262 010664 105037 001425              CLRB   DONFLG           ;INIT INTERRUPT DONE INDICATOR

```

```

2263 010670 005001          CLR      R1          :RX DATA POINTER- SET TO 0
2264 010672 005002          CLR      R2          :TX DATA POINTER- SET TO 0
2265 010674 013777 007152 171122  MOV     NUMTCR,@DZVTCR :SET UP TCR BIT
2266 010702 106427 000000  MTPS    #CLEAR       :ALLOW INTERRUPTS
2267
2268          :YOU RETURN HERE AFTER EVERY RECEIVER INTERRUPT
2269 010706 105777 170376  SPIN:   TSTB    @STKS    :IF SOMEBODY HITS A KEY- GET A NEW LINE #
2270 010712 100004          BPL      1$          :BRANCH IF NO KEY HIT
2271 010714 004737 005436  JSR     PC,SERV.G    :MAKE SURE IT WASN'T <^G>
2272 010720 000137 002420  JMP     LINEX        :SW02=1
2273 010724 105737 001425  1$:    TSTB    DONFLG   :ARE ALL RECEIVER INTER. DONE
2274 010730 001004          BNE      QUILTS     :IF YES GET OUT OF TIMING LOOP
2275 010732 005237 007132  INC     LOCKUP       :INC TIMEOUT FLAG
2276 010736 001363          BNE      SPIN       :IF NOT 0 RETURN SPINNING
2277 010740 104011          ERROR   11         :*RECEIVER FAILED TO INTERRUPT CHECK CABLE/TERMINATOR
2278 010742 104413          QUILTS: DEVICE.CLR
2279 010744 000137 002550  JMP     SEOP         :CALL FOR END OF PASS
2280 010750 005037 007132  INTREC: CLR     LOCKUP  :CLEAR TIMEOUT FLAG
2281 010754 105777 171030  TSTB    @DZVCSR     :TEST REC DONE
2282 010760 100401          BMI     .+4         :YES
2283 010762 104004          ERROR   4          :*FALSE INTERRUPT
2284 010764 017737 171024 007156  MOV     @DZVRBUF,RECDAT :SAVE WORD
2285 010772 100401          BMI     .+4
2286 010774 104023          ERROR   23         :*NON VALID CHARACTER
2287 010776 032737 040000 007156  BIT     #BIT14,RECDAT :DATA OVERRUN ?
2288 011004 001401          BEQ     .+4         :NO
2289 011006 104024          ERROR   24         :*YES
2290 011010 032737 020000 007156  BIT     #BIT13,RECDAT :FRAMING ERROR ?
2291 011016 001401          BEQ     .+4         :NO
2292 011020 104025          ERROR   25         :*YES
2293 011022 032737 010000 007156  BIT     #BIT12,RECDAT :PARITY ERROR ?
2294 011030 001401          BEQ     .+4         :NO
2295 011032 104026          ERROR   26         :*YES
2296 011034 110105          MOVB   R1,R5       :SET EXPECTED
2297 011036 113704 007156  MOVB   RECDAT,R4    :GET FOUND
2298 011042 042704 177400  BIC    #^C<377>,R4  :CLEAR HIGH BYTE
2299 011046 042705 177400  BIC    #^C<377>,R5  :CLEAR HIGH BYTE
2300 011052 020504          CMP     R5,R4       :OK?
2301 011054 001401          BEQ     .+4
2302 011056 104005          ERROR   5          :DATA ERROR
2303 011060 042737 176377 007156  BIC    #176377,RECDAT :SAVE ONLY LINE NUMBER
2304 011066 000337 007156  SWAB   RECDAT
2305 011072 023737 001374 007156  CMP    SAVLIN,RECDAT :DOES THE LINE # COMPARE ?
2306 011100 001407          BEQ     4$         :YES
2307 011102 013737 007156 001374  MOV    RECDAT,SAVLIN :ADJUST LINE NO. FOR ERROR
2308 011110 104015          ERROR   15         :*WRONG LINE #
2309 011112 013737 007150 001374  MOV    NUMLIN,SAVLIN :READJUST LINE NO.
2310 011120 120127 000377  4$:    CMPB   R1,#377    :LAST CHARACTER ?
2311 011124 001003          BNE     1$         :NO
2312 011126 105237 001425  INCB   DONFLG      :INDICATE RECEIVER INTERRUPTS DONE
2313 011132 000401          BR     2$
2314 011134 105201  1$:    INCB   R1
2315 011136 000002  2$:    RTI
2316
2317 011140 005777 170644  INTRAN: TST    @DZVCSR :TEST TRANSMIT FLAG
2318 011144 100401          BMI     .+4

```


2319	011146	104003			ERROR	3		:*FALSE INTERRUPT
2320	011150	110277	170660		MOVB	R2,@DZVTDR		:TRANSMIT A CHARACTER
2321	011154	105202			INCB	R2		:UPDATE TX DATA
2322	011156	001003			BNE	1\$:BIT PATTERN DONE?
2323	011160	042777	040000	170622	BIC	#TIE,@DZVCSR		:IF YES THEN CLEAR TIE
2324	011166	000002			RTI			:IF NOT THEN RETURN

1\$:

			:ERROR TABLE	
			.ERRTAB:	
2326				
2327	011170	000000	0	:ERROR 0
2328	011172	000000	0	
2329	011174	000000	0	
2330				
2331	011176	011416	EM1	:ERROR
2332	011200	012746	DH1	
2333	011202	013146	DT1	
2334				
2335	011204	011471	EM2	:ERROR 2
2336	011206	012772	DH2	
2337	011210	013160	DT2	
2338				
2339	011212	011517	EM3	:ERROR 3
2340	011214	013025	DH3	
2341	011216	013176	DT3	
2342				
2343	011220	011556	EM4	:ERROR 4
2344	011222	013025	DH3	
2345	011224	013176	DT3	
2346				
2347	011226	011605	EM5	:ERROR 5
2348	011230	013037	DH4	
2349	011232	013204	DT4	
2350				
2351	011234	011634	EM6	:ERROR 6
2352	011236	013037	DH4	
2353	011240	013204	DT4	
2354				
2355	011242	011673	EM7	:ERROR 7
2356	011244	013025	DH3	
2357	011246	013176	DT3	
2358				
2359	011250	011734	EM8	:ERROR 10
2360	011252	013025	DH3	
2361	011254	013176	DT3	
2362				
2363	011256	011776	EM9	:ERROR 11
2364	011260	013025	DH3	
2365	011262	013176	DT3	
2366				
2367	011264	012034	EM10	:ERROR 12
2368	011266	013025	DH3	
2369	011270	013176	DT3	
2370				
2371	011272	012073	EM13	:ERROR 13
2372	011274	013025	DH3	
2373	011276	013176	DT3	
2374				
2375	011300	012124	EM14	:ERROR 14
2376	011302	013025	DH3	
2377	011304	013176	DT3	
2378				
2379	011306	012156	EM15	:ERROR 15
2380	011310	000000	0	
2381	011312	000000	0	

2382				
2383	011314	012220	EM16	
2384	011316	013025	DH3	
2385	011320	013176	DT3	
2386				
2387	011322	012272	EM17	;ERROR 17
2388	011324	013025	DH3	
2389	011326	013176	DT3	
2390				
2391	011330	012330	EM20	
2392	011332	013025	DH3	
2393	011334	013176	DT3	
2394				
2395	011336	012371	EM21	;ERROR 21
2396	011340	013066	DH5	
2397	011342	013222	DT5	
2398				
2399	011344	012421	EM22	;ERROR 22
2400	011346	013037	DH4	
2401	011350	013204	DT4	
2402				
2403	011352	012463	EM23	;ERROR 23
2404	011354	013025	DH3	
2405	011356	013176	DT3	
2406				
2407	011360	012513	EM24	
2408	011362	013025	DH3	
2409	011364	013176	DT3	
2410				
2411	011366	012541	EM25	
2412	011370	013025	DH3	
2413	011372	013176	DT3	
2414				
2415	011374	012571	EM26	
2416	011376	013025	DH3	
2417	011400	013176	DT3	
2418				
2419	011402	012620	EM27	
2420	011404	013025	DH3	
2421	011406	013176	DT3	
2422				
2423	011410	012666	EM30	
2424	011412	013025	DH3	
2425	011414	013176	DT3	

```

2427                                     :ERROR MESSAGES
2431 011416 047200 020117 052502 EM1: .ASCIZ <200>/NO BUS REPLY RESPONSE FROM DZV11 REGISTER/
2432 011471      200 042522 044507 EM2: .ASCIZ <200>/REGISTER R/W FAILURE?
2433 011517      200 051124 047101 EM3: .ASCIZ <200>/TRANSMIT READY (TRDY) NOT SET/
2434 011556 051200 041505 044505 EM4: .ASCIZ <200>/RECEIVER DONE NOT SET/
2435 011605      200 040504 040524 EM5: .ASCIZ <200>/DATA COMPARISON ERROR/
2436 011634 042200 053132 030461 EM6: .ASCIZ <200>/DZV11 *RECEIVER BUFFER* ERROR/
2437 011673      200 051124 047101 EM7: .ASCIZ <200>/TRANSMITTER FAILED TO INTERRUPT/
2438 011734 052600 042516 050130 EM8: .ASCIZ <200>/UNEXPECTED TRANSMITTER INTERRUPT/
2439 011776 051200 041505 044505 EM9: .ASCIZ <200>/RECEIVER FAILED TO INTERRUPT/
2440 012034 052600 042516 050130 EM10: .ASCIZ <200>/UNEXPECTED RECEIVER INTERRUPT/
2441 012073      200 044523 047514 EM13: .ASCIZ <200>/SILO ALARM SET TOO SOON/
2442 012124 051600 046111 020117 EM14: .ASCIZ <200>/SILO ALARM FAILED TO SET/
2443 012156 040600 052103 047511 EM15: .ASCIZ <200>/ACTION DETECTED ON INVALID LINE./
2444 012220 051200 040505 044504 EM16: .ASCIZ <200>/READING DZVRBUF DID NOT CLEAR SILO ALARM/
2445 012272 042200 052101 020101 EM17: .ASCIZ <200>/DATA VALID SHOULD NOT BE SET/
2446 012330 051200 041505 044505 EM20: .ASCIZ <200>/RECEIVER DONE SHOULD NOT BE SET/
2447 012371      200 042522 040514 EM21: .ASCIZ <200>/RELATIVE TIMING ERROR./
2448 012421      200 047515 042504 EM22: .ASCIZ <200>/MODEM SIGNAL ERROR ON CABLE TEST/
2449 012463      200 040504 040524 EM23: .ASCIZ <200>/DATA VALID IS NOT SET!/
2450 012513      200 040504 040524 EM24: .ASCIZ <200>/DATA OVERRUN IS SET!/
2451 012541      200 051106 046501 EM25: .ASCIZ <200>/FRAMING ERROR OCCURRED/
2452 012571      200 040520 044522 EM26: .ASCIZ <200>/PARITY ERROR OCCURRED/
2453 012620 051600 046111 020117 EM27: .ASCIZ <200>/SILO ALARM FAILED TO CAUSE INTERRUPT/
2454 012666 046200 047111 020105 EM30: .ASCIZ <200>/LINE DID NOT RECEIVE FULL BINARY COUNT PATTERN/
2455
2456 012746 052200 040522 020120 DH1: .ASCIZ <200>/TRAP PC DZV11 REG/
2457 012772 042600 050130 041505 DH2: .ASCIZ <200>/EXPECTED FOUND REGISTER/
2458 013025      200 044514 042516 DH3: .ASCIZ <200>/LINE NO./
2459 013037      200 054105 042520 DH4: .ASCIZ <200>/EXPECTED FOUND LINE/
2460 013066 052200 020130 044514 DH5: .ASCIZ <200>/TX LINE PREVIOUS TIME ACTUAL TIME PARAMETER/
2461
2462      013146          .EVEN
2466                                     :DATA TABLES FOR ERROR MESSAGES
2467 013146 000002          DT1: 2
2468 013150      006      003      .BYTE 6.3
2469 013152 001330          $REG1
2470 013154      006      001      .BYTE 6.1
2471 013156 001326          $REG0
2472
2473 013160 000003          DT2: 3
2474 013162      006      004      .BYTE 6.4
2475 013164 001340          $REG5
2476 013166      006      001      .BYTE 6.1
2477 013170 001336          $REG4
2478 013172      006      001      .BYTE 6.1
2479 013174 001326          $REG0
2480
2481 013176 000001          DT3: 1
2482 013200      003      001      .BYTE 3.1
2483 013202 001374          SAVLIN
2484
2485 013204 000003          DT4: 3
2486 013206      006      004      .BYTE 6.4
2487 013210 001340          $REG5
2488 013212      006      001      .BYTE 6.1

```

2489	013214	001336		\$REG4	
2490	013216	003	001	.BYTE	3.1
2491	013220	001374		SAVLIN	
2492					
2493	013222	000004		DTS:	4
2494	013224	003	005	.BYTE	3.5
2495	013226	001374		SAVLIN	
2496	013230	006	011	.BYTE	6.9.
2497	013232	001340		\$REG5	
2498	013234	006	007	.BYTE	6.7
2499	013236	001344		\$TMP1	
2500	013240	006	001	.BYTE	6.1
2501	013242	001402		REGIST	

:TABLE OF DELAY TIMES FOR INDIVIDUAL BAUD RATES

2505	013244	002450		DLYTBL:	2450	:TIME FOR	50	BAUD
2506	013246	001560			1560	:TIME FOR	75	BAUD
2507	013250	001120			1120	:TIME FOR	110	BAUD
2508	013252	000750			750	:TIME FOR	134	BAUD
2509	013254	000660			660	:TIME FOR	150	BAUD
2510	013256	000330			330	:TIME FOR	300	BAUD
2511	013260	000150			150	:TIME FOR	600	BAUD
2512	013262	000060			60	:TIME FOR	1200	BAUD
2513	013264	000040			40	:TIME FOR	1800	BAUD
2514	013266	000030			30	:TIME FOR	2000	BAUD
2515	013270	000020			20	:TIME FOR	2400	BAUD
2516	013272	000010			10	:TIME FOR	3600	BAUD
2517	013274	000001			1	:TIME FOR	4800	BAUD
2518	013276	000001			1	:TIME FOR	7200	BAUD
2519	013300	000001			1	:TIME FOR	9600	BAUD
2520	013302	000001			1	:TIME OF DELAY FOR	19200	BAUD

:DELAYS WERE COMPUTED TO ALLOW MAXIMUM TIME AT EACH BAUD RATE
:FOR ALL TESTS TO FUNCTION CORRECTLY ON A LSI11.

2521
2522
2523

2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541 013304 005227 177777
2542 013310 001002
2543 013312 004737 000400
2544 013316 005727
2545 013320 000000
2546 013322 000207
2547
2548
2549
2550 000400 005037 013320
2551 000404 013746 000004
2552 000410 012737 000504 000004
2553 000416 012700 160010
2554 000422 005720
2555 000424 000240
2556 000426 020027 174000
2557 000432 103773
2558 000434 010037 013320
2559 000440 012700 000040
2560 000444 040037 000006
2561 000450 040037 000016
2562 000454 040037 000022
2563 000460 040037 000032
2564 000464 040037 000036
2565 000470 012737 170000 000140
2566 000476 012637 000004
2567 000502 000207
2568
2569 000504 012716 000512
2570 000510 000002
2571 000512 012637 000004
2572 000516 012700 000402
2573 000522 013701 000376
2574 000526 010602
2575 000530 012704 000570
2576 000534 014446
2577 000536 020427 000546
2578 000542 101374
2579 000544 010607

```
.SBTTL FALCON (KXT-11) UPGRADE ROUTINES.                ::GPA
:
: THE FOLLOWING ROUTINES HAVE BEEN ADDED TO ALLOW DIAGNOSTIC(S)
: TO RUN ON A FALCON (KXT-11) BASED SYSTEM.
: TO DETERMINE WHETHER WE'RE A FALCON OR NOT, WE'LL SIZE THE 1ST 3/4 OF
: THE I/O PAGE (28K TO 31K). FALCON HAS 2KW LOCAL RAM AT 28K(+4) TO 30K
: AND A MACRO-ODT AT 30K TO 31K. CONSEQUENTLY, ALL I/O DEVICES MUST
: BE PLACED BETWEEN 174000 AND 177776. ADDITIONALLY, WE'LL STRAP THE
: EMT AND TRAP SERVICE LEVEL TO PRI6, AND SET THE HALT VECTOR SO THAT
: WE CAN STOP THE SUCKER !!
:
: TO MINIMIZE THE IMPACT OF THESE CHANGES ON FINAL PROGRAM SIZE, THE
: BULK OF THIS CODE IS PLACED IN THE FLOATING VECTOR SPACE (400-776).
: IF THE CPU AT HAND IS A FALCON (KXT11), IT STAYS THERE (NO HARM DONE).
: OTHERWISE, THE AREA IS RESTORED TO ITS ORIGINAL 'TRAP-CATCHER' STATE.
:
FALCON: INC      #-1                ; ONCE-ONLY !!!                ::GPA
        BNE      1$                ;                               ::GPA
        CALL     KXTCHK             ; EXECUTE FALCON CHECK          ::GPA
1$:     TST      (PC)+              ; TEST FALCON FLAG...        ::GPA
KXTFLAG: 0                    ; ...NZ = FALCON...          ::GPA
        RETURN                    ; ...AND RETURN TO CALLER...  ::GPA
:
        $SVPC= 0                  ;                               ::GPA
        = 400                    ;                               ::GPA
KXTCHK: CLR      KXTFLAG           ; RESTORE FROM 374:376 AT END  ::GPA
        MOV     @#4,-(SP)          ; ASSUME NOT FALCON.         ::GPA
        MOV     #2$,@#4           ; SAVE ERROR VECTOR.        ::GPA
        MOV     #160010,R0        ; SET A TRAP CATCHER.       ::GPA
1$:     TST     (R0)+             ; FALCON RAM STARTS AT 28K+4. ::GPA
        240                       ;                               ::GPA
        CMP     R0,#174000        ; SIZE TO 31K.              ::GPA
        BLO     1$                ;                               ::GPA
        MOV     R0,KXTFLAG        ; MUST BE FALCON, SET THE FLAG ::GPA
        MOV     #40,R0            ; GET PRI1 BIT...           ::GPA
        BIC     R0,@#6            ; ...AND LOWER BUS-ERROR...  ::GPA
        BIC     R0,@#16          ; ...BPT...                 ::GPA
        BIC     R0,@#22          ; ...IOT...                 ::GPA
        BIC     R0,@#32          ; ...EMT...                 ::GPA
        BIC     R0,@#36          ; ...AND TRAP SERVICE TO PRI6 ::GPA
        MOV     #170000,@#140     ; ENABLE 'BREAK' HALT.      ::GPA
        MOV     (SP)+,@#4         ; RESTORE ERROR VECTOR...   ::GPA
        RETURN                    ; ...AND RETURN.            ::GPA
:
2$:     MOV     #3$,(SP)          ; TRAP -- NOT A FALCON...    ::GPA
        RTI                       ; ...CONTINUE.              ::GPA
3$:     MOV     (SP)+,@#4         ; RESET ERROR VECTOR        ::GPA
        MOV     #402,R0           ; SET-UP TO RESTORE FLOATING... ::GPA
        MOV     @#376,R1         ; ...VECTORS (400 - 776).   ::GPA
        MOV     SP,R2            ; SAVE STACK POINTER IN R2  ::GPA
        MOV     #6$,R4           ;                               ::GPA
4$:     MOV     -(R4),-(SP)       ; PUSH THE RESTORE CODE...   ::GPA
        CMP     R4,#5$           ; ...ONTO THE STACK.        ::GPA
        BHI     4$                ;                               ::GPA
        MOV     SP,PC            ; AND EXECUTE IT.           ::GPA
```

```

2581
2582
2583
2584 000546 010060 177776
2585 000552 010110
2586 000554 022020
2587 000556 020027 000776
2588 000562 101771
2589 000564 010206
2590 000566 000207
2591 000570
2592
2593
2594
2595
2596
2597
2598
2599
2600 000570 023727 001174 160010
2601 000576 001003
2602 000600 012737 174040 001174
2603 000606 023727 001170 000000
2604 000614 001003
2605 000616 012737 000370 001170
2606 000624 012737 000670 002334
2607 000632 012737 174000 002340
2608 000640 012737 177770 002342
2609 000646 012737 000732 002352
2610 000654 005037 002356
2611 000660 012737 000370 002360
2612 000666 000207
2613
2614 000670 030600 052123 041440
2615 000732 030600 052123 053040
2616
2617
2618
2622
2623
2624 013324
2628 000001

:
: THIS CODE IS RELOCATED TO AND EXECUTED IN THE STACK AREA.
:
5$:  MOV     R0,-2(R0)      : RESTORE .+2...           ::GPA
    MOV     R1,(R0)       : ...HALT (OR IOT).       ::GPA
    CMP     (R0)+,(R0)+   :                          :GPA
    CMP     R0,#776      :                          :GPA
    BLOS   5$            : LOOP 'TIL DONE         :GPA
    MOV     R2,SP        : THEN RESTORE SP...     :GPA
    RETURN  : ...AND RETURN TO CALLER ::GPA
6$:
:
: IF FALCON, THIS AREA IS FREE FOR ANY PROGRAM UNIQUE
: CHANGES OR DATA STRUCTURES.
: BE SURE IT DOESN'T GET SCREWED UP !!
:
: INIT SBASE AND SVECT1 AND TWEAK THE '$GETPAR' CALLING
: SEQUENCE TO ACCEPT THE VALID FALCON RANGE.
:
FALCINI: CMP     SBASE,#ABASE : IS SBASE VIRGIN ??      ::GPA
      BNE     1$           : SKIP NEXT IF NOT       :GPA
      MOV     #174040,SBASE : YES, SET ENGINEERING DEFAULT ::GPA
1$:   CMP     SVECT1,#AVECT1 : IS SVECT1 VIRGIN ??   :GPA
      BNE     2$           : SKIP NEXT IF NOT       :GPA
      MOV     #370,SVECT1  : YES, SET ENGINEERING DEFAULT ::GPA
2$:   MOV     #3$,GETCSR+2  : SUBSTITUE CSR TEXT...  :GPA
      MOV     #174000,GETCSR+6 :                          :GPA
      MOV     #177770,GETCSR+10 : ...AND VALID RANGE.   :GPA
      MOV     #4$,GETVEC+2  : SUBSTITUTE VECTOR TEXT... :GPA
      CLR     GETVEC+6      :                          :GPA
      MOV     #370,GETVEC+10 : ...AND VALID RANGE.   :GPA
      RETURN  : RETURN TO CALLER. ::GPA
3$:   .ASCIZ  <200>'1ST CSR ADDRESS (174000:177770) ' :GPA
4$:   .ASCIZ  <200>'1ST VECTOR ADDRESS (000:370) ' :GPA
      .EVEN :GPA
$FREE= <1000-.>/2 : FREE WORDS LEFT. :GPA
      .=$SVPC :GPA
CORMAX:
.END

```

ABASE = 160010	AVECT = 000300	DDISP = 177570	DZVC2 = 001526	FIVE = 000000
ACDW1 = 000000	AVECT1 = 000000	DELAY = 104414	DZVC3 = 001540	FIVES = 000040
ACDW2 = 000000	AVECT2 = 000000	DEVADR = 004322	DZVC4 = 001552	FRMERR = 020000
ACPUOP = 000000	BAUD = 002402	DEVICE = 104413	DZVC5 = 001564	GETCSR = 002332
ACTIVE = 001420	BINWRD = 004600	DH1 = 012746	DZVC6 = 001576	GETSWR = 005464
ADDW0 = 000000	BIT0 = 000001	DH2 = 012772	DZVC7 = 001610	GETVEC = 002350
ADDW1 = 000000	BIT00 = 000001	DH3 = 013025	DZVLEV = 007764	HALTS = 005276
ADDW10 = 000000	BIT01 = 000002	DH4 = 013037	DZVLPV = 002020	HDRFLG = 001423
ADDW11 = 000000	BIT02 = 000004	DH5 = 013066	DZVMSR = 002030	HDZVCS = 002012
ADDW12 = 000000	BIT03 = 000010	DISPLA = 001306	DZVNUM = 001414	HDZVLP = 002022
ADDW13 = 000000	BIT04 = 000020	DISPRE = 000174	DZVRBU = 002014	HDZVMS = 002032
ADDW14 = 000000	BIT05 = 000040	DLYCNT = 004674	DZVVIS = 002042	HDZVRB = 002016
ADDW15 = 000000	BIT06 = 000100	DLYTBL = 013244	DZVRIV = 002040	HDZVTC = 002026
ADDW2 = 000000	BIT07 = 000200	DONFLG = 001425	DZVTCR = 002024	HDZVTD = 002036
ADDW3 = 000000	BIT08 = 000400	DSWR = 177570	DZVTDR = 002034	HILIM = 004320
ADDW4 = 000000	BIT09 = 001000	DTR0 = 000400	DZVTIS = 002046	HT = 000011
ADDW5 = 000000	BIT1 = 000002	DTR1 = 001000	DZVTIV = 002044	INBUF = 007616
ADDW6 = 000000	BIT10 = 002000	DTR2 = 002000	DZVEN = 001740	INIFLG = 001422
ADDW7 = 000000	BIT11 = 004000	DTR3 = 004000	DZVMA = 001500	INSTER = 104404
ADDW8 = 000000	BIT12 = 010000	DT1 = 013146	EIGHT = 000030	INSTR = 104403
ADDW9 = 000000	BIT13 = 020000	DT2 = 013160	EIGHTS = 000070	INSTR2 = 004120
ADEVCT = 000000	BIT14 = 040000	DT3 = 013176	EMTVEC = 000030	INTRAN = 011140
ADEVN = 000000	BIT15 = 100000	DT4 = 013204	EM1 = 011416	INTREC = 010750
ADRCNT = 004325	BIT2 = 000004	DT5 = 013222	EM10 = 012034	INTSVC = 010324
ADVANC = 104400	BIT3 = 000010	DVALID = 100000	EM13 = 012073	IOTVEC = 000020
AENV = 000000	BIT4 = 000020	DZCRO = 001500	EM14 = 012124	KXTCHK = 000400
AENVN = 000000	BIT5 = 000040	DZCR1 = 001512	EM15 = 012156	KXTFLA = 013320
AFATAL = 000000	BIT6 = 000100	DZCR10 = 001620	EM16 = 012220	LAST = 007134
AMADR1 = 000000	BIT7 = 000200	DZCR11 = 001632	EM17 = 012272	LF = 000012
AMADR2 = 000000	BIT8 = 000400	DZCR12 = 001644	EM2 = 011471	LIMITS = 004246
AMADR3 = 000000	BIT9 = 001000	DZCR13 = 001656	EM20 = 012330	LINE = 001366
AMADR4 = 000000	BPTVEC = 000014	DZCR14 = 001670	EM21 = 012371	LINESP = 007144
AMAMS1 = 000000	BRK0 = 000400	DZCR15 = 001702	EM22 = 012421	LINEX = 002420
AMAMS2 = 000000	BRK1 = 001000	DZCR16 = 001714	EM23 = 012463	LINE0 = 001504
AMAMS3 = 000000	BRK2 = 002000	DZCR17 = 001726	EM24 = 012513	LINE1 = 001516
AMAMS4 = 000000	BRK3 = 004000	DZCR2 = 001524	EM25 = 012541	LINE10 = 001624
AMSGAD = 000000	BUFSET = 104422	DZCR3 = 001536	EM26 = 012571	LINE11 = 001636
AMSGLG = 000000	BYTCNT = 007142	DZCR4 = 001550	EM27 = 012620	LINE12 = 001650
AMSGTY = 000000	CHRCNT = 004576	DZCR5 = 001562	EM3 = 011517	LINE13 = 001662
AMTYP1 = 000000	CLEAR = 000000	DZCR6 = 001574	EM30 = 012666	LINE14 = 001674
AMTYP2 = 000000	CNVRT = 104412	DZCR7 = 001606	EM4 = 011556	LINE15 = 001706
AMTYP3 = 000000	CONVRT = 104411	DZVACT = 001406	EM5 = 011605	LINE16 = 001720
AMTYP4 = 000000	CORMAX = 013324	DZVCSR = 002010	EM6 = 011634	LINE17 = 001732
APASS = 000000	CO0 = 000400	DZVCO = 001502	EM7 = 011673	LINE2 = 001530
APRIOR = 000000	CO1 = 001000	DZVC1 = 001514	EM8 = 011734	LINE3 = 001542
APTCSU = 000040	CO2 = 002000	DZVC10 = 001622	EM9 = 011776	LINE4 = 001554
APTENV = 000001	CO3 = 004000	DZVC11 = 001634	ERRMSG = 005246	LINE5 = 001566
APTSIZ = 000200	CR = 000015	DZVC12 = 001646	ERRVEC = 000004	LINE6 = 001600
APTSPO = 000100	CRLF = 000200	DZVC13 = 001660	ERTABO = 005422	LINE7 = 001612
ASWREG = 000000	DATAHP = 005272	DZVC14 = 001672	EVEPAR = 000000	LOBITS = 004324
ATESTN = 000000	DATAHD = 005260	DZVC15 = 001704	EXITER = 005352	LOCK = 001364
AUNIT = 000000	DCLASM = 104417	DZVC16 = 001716	FALCIN = 000570	LOCKUP = 007132
AUSWR = 000000	DCLR = 000020	DZVC17 = 001730	FALCON = 013304	LOLIM = 004316

LPRSET= 104421
LPO = 000000
LP1 = 000001
LP2 = 000002
LP3 = 000003
MAINT = 000010
MANT0 001510
MANT1 001522
MANT10 001630
MANT11 001642
MANT12 001654
MANT13 001666
MANT14 001700
MANT15 001712
MANT16 001724
MANT17 001736
MANT2 001534
MANT3 001546
MANT4 001560
MANT5 001572
MANT6 001604
MANT7 001616
MASK = 000200
MASTEK 006351
MBADLN 006462
MCABLE 007464
MCHAR 007330
MCSRX 006301
MDATA 007722
MEPASS 006117
MERRPC 006427
MERRX 006326
MERR2 006157
MERR3 006226
MINVAL 007254
MLINE 007302
MLOCK 006252
MNEW 006354
MNTFLG 001424
MODE 001372
MPASS 007240
MPASSX 006315
MPFAIL 006054
MQUICK 007501
MR 006143
MREGAD 007204
MSENAB= 000040
MSPEED 007312
MTERM 007437
MTITLE 001000
MTSTN 006337
MVECTO 007162
MVECX 006307

MWHICH 007371
NEXT 001362
NUMLIN 007150
NUMTCR 007152
ODDPAR= 000200
ONESTO= 000000
OVRRUN= 040000
PAR 001370
PARAM = 104405
PARAM1 004166
PARER = 010000
PARERR 004242
PARITY= 000100
PARMD = 104415
PAR0 001506
PAR1 001520
PAR10 001626
PAR11 001640
PAR12 001652
PAR13 001664
PAR14 001676
PAR15 001710
PAR16 001722
PAR17 001734
PAR2 001532
PAR3 001544
PAR4 001556
PAR5 001570
PAR6 001602
PAR7 001614
PAWCH = 104416
PIRQ = 177772
PIRQVE= 000240
POPPO = 012600
POP1SP= 005726
POP2SP= 022626
PRIO 007154
PR0 = 000000
PR1 = 000040
PR2 = 000100
PR3 = 000140
PR4 = 000200
PR5 = 000240
PR6 = 000300
PR7 = 000340
PS = 177776
PSW = 177776
PUSHRO= 010046
PUSH1S= 005746
PUSH2S= 024646
PWRVEC= 000024
QUITS 010742
RCVON = 010000

RDATA 007140
RDONE = 000200
RECDAT 007156
REGIST 001402
RESREG 005274
RESTAR 002544
RESVEC= 000010
RES05 = 104410
RIE = 000100
RING0 = 000001
RING1 = 000002
RING2 = 000004
RING3 = 000010
RLO = 000000
RL1 = 000400
RL2 = 001000
RL3 = 001400
RUN 001412
R6 =X000006
R7 =X000007
SAVACT 001410
SAVLIN 001374
SAVNO 001416
SAVNUM 001415
SAVPC 001404
SAV05 = 104407
SCOP1 = 104401
SERV.G 005436
SET 006732
SETFLG= 104406
SET1 006764
SEVEN = 000020
SEVENS= 000060
SHIFT = 104420
SILQAL= 020000
SILOEN= 010000
SIX = 000010
SIXS = 000050
SPACNT 004577
SPEED 007146
SPIN 010706
STACK = 001120
STFLG 007130
STKLMT= 177774
STOP 001446
SV05 004334
SWR 001304
SWREG 000176
SW0 = 000001
SW00 = 000001
SW01 = 000002
SW02 = 000004
SW03 = 000010

SW04 = 000020
SW05 = 000040
SW06 = 000100
SW07 = 000200
SW08 = 000400
SW09 = 001000
SW1 = 000002
SW10 = 002000
SW11 = 004000
SW12 = 010000
SW13 = 020000
SW14 = 040000
SW15 = 100000
SW2 = 000004
SW3 = 000010
SW4 = 000020
SW5 = 000040
SW6 = 000100
SW7 = 000200
SW8 = 000400
SW9 = 001000
S110 = 001000
S1200 = 003400
S134 = 001400
S150 = 002000
S1800 = 004000
S19200= 007400
S2000 = 004400
S2400 = 005000
S300 = 002400
S3600 = 005400
S4800 = 006000
S50 = 000000
S600 = 003000
S7200 = 006400
S75 = 000400
S9600 = 007000
TABLE2 007026
TBITVE= 000014
TBUF 007160
TCRO = 000001
TCR1 = 000002
TCR2 = 000004
TCR3 = 000010
TDATA 007136
TD0 001426
TD1 001430
TD2 001432
TD3 001434
TEIGHT 002106
TEMP 007660
TFIVE 002114
TIE = 040000

TKVEC = 000060
TLO = 000000
TL1 = 000400
TL2 = 001000
TL3 = 001400
TMTBL 002050
TPVEC = 000064
TRAPVE= 000034
TRDY = 100000
TRTVEC= 000014
TRO 001436
TR1 001440
TR2 001442
TR3 001444
TSEVEN 002110
TSIX 002112
TST1 010124
TST2 010500
TWOSTO= 000040
TYPDAT 005262
TYPE = 104402
TYPMSG 005152
T110 002054
T1200 002066
T134 002056
T150 002060
T1800 002070
T2000 002072
T2400 002074
T300 002062
T3600 002076
T4800 002100
T50 002050
T600 002064
T7200 002102
T75 002052
T9600 002104
VEC1 002272
WCHFLG 007126
WRDCNT 004574
WTBS.F 005250
XBEGIN 002450
XBX 005040
XCSR 002712
XERR 002734
XHEAD 006434
XMTCNT 001400
XMTLIN 001376
XPASS 002726
XSTATQ 006524
XTCR1 006754
XTSTN 005430
XVEC 002720

XX = 160210	\$DDW6 001220	\$ILLUP 006046	\$REGAD 001324	\$VECT1 001170
YY = 000500	\$DDW7 001222	\$INTAG 001301	\$REG0 001326	\$VECT2 001172
ZZ = 000020	\$DDW8 001224	\$ITEMB 001260	\$REG1 001330	\$XOFF = 000023
\$APTHD 001446	\$DDW9 001226	\$LFLG 003777	\$REG2 001332	\$XON = 000021
\$ATYC 003560	\$DEVCT 001130	\$LPADR 001252	\$REG3 001334	\$Y = 000023
\$ATY1 003534	\$DEVM 001176	\$LPERR 001254	\$REG4 001336	\$\$GET4= 000000
\$ATY3 003542	\$DOAGN 002706	\$MADR1 001152	\$REG5 001340	. = 013324
\$ATY4 003552	SE = 000002	\$MADR2 001156	\$RTNAD 002710	.ADVAN 004676
\$AUTOB 001300	\$ENDAD 002676	\$MADR3 001162	\$SAVR6 006052	.BUFSE 004766
\$BASE 001174	\$ENDCT 002662	\$MADR4 001166	\$SETUP= 000000	.CNVRT 004424
\$BDADR 001266	\$ENV 001140	\$MAIL 001120	\$SVPC = 013324	.CONVR 004420
\$BDDAT 001272	\$ENVM 001141	\$MAMS1 001150	\$SWR = 164000	.DCLAS 004644
\$CDW1 001200	\$EOP 002550	\$MAMS2 001154	\$SWREG 001142	.DELAY 004656
\$CDW2 001202	\$EOPCT 002654	\$MAMS3 001160	\$TESTN 001124	.DEVIC 004624
\$CHARC 003530	\$ERFLG 001247	\$MAMS4 001164	\$TIMES 001354	.ERRTA 011170
\$CMTAG 001244	\$ERMAX 001261	\$MBADR 001450	\$TKB 001312	.INSTE 004106
\$CM1 = 000006	\$ERROR 005010	\$MFLG 003776	\$TKS 001310	.INSTR 004002
\$CM2 = 000014	\$ERRPC 001262	\$MSGAD 001134	\$TMP0 001342	.INST1 004022
\$CM3 = 000006	\$ERRTB 001362	\$MSGLG 001136	\$TMP1 001344	.LPRSE 004726
\$CM4 = 000005	\$ERTTL 001256	\$MSGTY 001120	\$TMP2 001346	.MSG 004024
\$CPUOP 001146	\$ETABL 001140	\$SMTYP1 001151	\$TMP3 001350	.PARAM 004126
\$CRLF 001357	\$ETEND 001244	\$SMTYP2 001155	\$TMP4 001352	.PARMD 002742
\$DDW0 001204	\$FATAL 001122	\$SMTYP3 001161	\$TN = 000001	.PAWCH 006656
\$DDW1 001206	\$FFLG 004000	\$SMTYP4 001165	\$TPB 001316	.RES05 004366
\$DDW10 001230	\$FILLC 001322	\$SN = 000001	\$TPFLG 001323	.SAV05 004326
\$DDW11 001232	\$FILLS 001321	\$NULL 001320	\$TPS 001314	.SCOPI 003136
\$DDW12 001234	\$FLIP = 177777	\$PASS 001126	\$STM 001452	.SETFL 006536
\$DDW13 001236	\$FREE = 000002	\$PASTM 001454	\$STNM 001246	.SHIFT 004710
\$DDW14 001240	\$GDADR 001264	\$PWRAD 006042	\$TYPE 003200	.START 002116
\$DDW15 001242	\$GDDAT 001270	\$PWRDN 005702	\$TYPEC 003412	.TRPSR 004602
\$DDW2 001210	\$GET42 002666	\$PWRMG 006036	\$TYPEX 003532	.TRPTA 001742
\$DDW3 001212	\$HD = 000001	\$PWRUP 005754	\$UNIT 001132	.TYPE 003162
\$DDW4 001214	\$HIBTS 001446	\$QUES 001356	\$UNITM 001456	.\$X = 001446
\$DDW5 001216	\$ICNT 001250			

. ABS. 013324 000 OVR RO ABS GBL D

ERRORS DETECTED: 0

CVDZCB,CVDZCB=CVDZCB
RUN-TIME: 18 9 .7 SECONDS
RUN-TIME RATIO: 196/28=6.8
CORE USED: 37K (73 PAGES)