

VT61, VT61T

DJ11 VT61 EXER
CZVTJBO

AH-9527B-MC
FICHE 1 OF 1

FEB 1981
COPYRIGHT © 76-80
MADE IN USA



The main body of the document consists of a grid of approximately 15 columns and 20 rows of data. Each cell in the grid contains a small, dense table or list of information, possibly representing a schedule or a set of parameters. The text is very small and difficult to read, but the layout is highly structured and repetitive.

.REM %

IDENTIFICATION

PRODUCT CODE: AC-9526B-MC
PRODUCT NAME: CZVTJBO DJ11 VT61 EXER
PRODUCT DATE: AUGUST, 1980
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.

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

COPYRIGHT (C) 1976,1980 BY DIGITAL EQUIPMENT CORPORATION.

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

| | | | |
|---------|-------|---------|---------|
| DIGITAL | PDP | UNIBUS | MASSBUS |
| DEC | DECUS | DECTAPE | |

TABLE OF CONTENTS

1. ABSTRACT
2. REQUIREMENTS (EQUIPMENT & MEMORY)
3. LOADING PROCEDURE
4. STARTING PROCEDURE
5. OPERATING PROCEDURE
6. ERRORS-GENERAL
7. RESTRICTIONS
8. MISCELLANEOUS
9. PROGRAM TESTS DESCRIPTION
10. CHANGE HISTORY

1. ABSTRACT

THIS PROGRAM IS AN ACCEPTANCE TEST FOR THE ENTIRE VT61 FAMILY OF TERMINALS. THE FUNCTIONAL TESTING IS BASED UPON A SET OF TERMINAL FUNCTIONS WHICH ARE COMMON THROUGHOUT THE ENTIRE FAMILY OF VT61 TYPE TERMINALS. THE FUNCTIONS AND THEIR DERIVED TESTING IS DESIGNED TO COMPLETELY CHECK (AT THE FUNCTIONAL LEVEL) THE TERMINAL MICRO-PROCESSOR AND ASSOCIATED RAMS. ALL TRANSMISSIONS TO THE VT61 WILL BE PRECEDED BY A SOM AND TERMINATED BY AEOM//.

THERE ARE TWO DISTINCT MODES IN WHICH THE PROGRAM CAN BE OPERATED. IN "AUTO" MODE UP TO 2 DJ11'S WITH UP TO 32 OPERATIONAL VT61'S WILL BE MAPPED AND ALL WILL BE TESTED SEQUENTIALLY. ALL TESTS WHICH DO NOT REQUIRE MANUAL INTERVENTION OR VISUAL SCREEN OBSERVATION (TESTS 1 THRU 20) WILL BE EXECUTED FOR EACH VT61 REPETITIVELY. ALL ERRORS WILL BE REPORTED ON THE SYSTEM CONSOLE (WHICH IS NOT TESTED EVEN IF IT IS A VT61).

IN MANUAL MODE CONSOLE ENTRY OF THE ADDRESSES AND TESTS IS REQUIRED. THE ADDRESSES AND TESTS CAN BE ENTERED IN A NON-SEQUENTIAL MANNER AND THE SUBSEQUENT EXECUTION WILL FOLLOW THE ENTRY SEQUENCE. THIS MODE MUST BE UTILIZED TO ENTER THE KEYBOARD TESTS, DATA LOOP TEST, AND PRINTER CONTROLLER TEST. SEQUENCE COMPLETION WILL EXIT TO THE RE-START POINT FOR THE MANUAL TEST.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP 11 FAMILY COMPUTER WITH 8K WORDS OF MEMORY, A CONSOLE, AND UP TO 32 VT61'S CONNECTED TO THE HOST COMPUTER VIA DJ11(S). VT61 MUST BE IN REMOTE; FULL DUPLEX AND AT LEAST 300 BAUD.

3. LOADING PROCEDURE

PROCEDURE FOR NORMAL BINARY PAPERTAPES SHOULD BE FOLLOWED.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP 11 FORMAT

SW15 = 1 HALT ON ERROR.
SW14 = 1 LOOP ON TEST
SW13 = 1 INHIBIT ERROR TYPEOUTS
SW11 = 1 INHIBIT ITERATIONS
SW10 = 1 BELL ON ERROR
SW9 = 1 LOOP ON ERROR
SW8 = 1 LOOP ON TEST IN SWR<7:0>

SPECIAL NOTE

IF THE COMPUTER UTILIZED IS A LSI 11 OR A COMPUTER WITHOUT A SWITCH REGISTER. THE PROGRAM WILL UTILIZE LOCATIONS 174 AND 176 AS A 'DISPLAY' REGISTER AND A 'SWITCH' REGISTER RESPECTIVELY. THE OPERATOR WILL BE RESPONSIBLE FOR THE LOADING OF THE 'SWITCH' REGISTER LOCATION PRIOR TO STARTING OR RESTARTING THE PROGRAM.

4.2 STARTING ADDRESSES

200 IS THE STARTING ADDRESS OF THE 'AUTO' ACCEPTANCE TEST
204 IS THE STARTING ADDRESS ON THE 'MANUAL' SELECT TEST.

5. OPERATING PROCEDURE

5.1 AUTO ACCEPTANCE MODE (SA = 200).

IN THIS MODE THE ONLY OPERATOR INTERVENTION REQUIRED IS SWR OPTION SELECTIONS SUCH AS LOOP ON TEST (SWR 11), BELL ON ERROR (SWR 0), ECT.. THE PROGRAM WILL, WITHOUT ANY EXTERNAL INTERVENTION, LOCATE THE DJ11(S)/LINES WITH VT61 TYPE UNITS ATTACHED AND SEQUENTIALLY TEST ALL UNITS REPETITIVELY WITH TESTS 1 THRU 20.

5.2 MANUAL UNIT/TEST SELECTION MODE (SA = 204)

THIS MODE REQUIRES THE OPERATOR TO ENTER THE ADDRESSES OF THE DJ11'S TO BE TESTED (FORMAT IS 17XXXX, ECT, -UP TO 2 ENTRIES). THE ENTRIES MUST BE SEPARATED BY COMMAS AND TERMINATED WITH A CARRIAGE RETURN. ENTERING AN ILLEGAL ADDRESS WILL RESULT IN A '?' BEING TYPED AND THE ADDRESS IGNORED! THE PROGRAM WILL THEN REQUEST THE LINES TO BE TESTED, IN BINARY FORMAT. TO TEST LINES ON TWO DJ11S, INSERT A -1(177777) AT THE END OF THE LINE LIST FOR EACH DJ11 AND TERMINATE THE ENTIRE LIST WITH A 0 WORD(000000).EXAMPLE- TEST LINE 1 OF 1ST DJ11 AND LINE 4 OF 2ND DJ11; ENTERED LIST WOULD BE 000002,177777,000020,177777,000000 C/R. THE OPERATOR MUST THEN, UPON PROGRAM REQUEST, ENTER A LIST OF TESTS TO BE EXECUTED IN THE SAME FORMAT AS THE ADDRESS ENTRY (I.E. YY,ZZ,C/R). PRECEEDING THE TERMINATING CARRIAGE RETURN WITH A 377 OCTAL WILL RESULT IN THE TESTS BEING REPETITIVELY EXECUTED FOR ALL ADDRESSES ENTERED.

SIMPLY DEPRESSING A CARRIAGE RETURN WHEN UNIT ADDRESSES ARE REQUESTED WILL RESULT IN THE MAPPING AND TESTING OF ALL GOOD DJ11(S)/LINES WITH OPERATIONAL VT61'S ATTACHED. HOWEVER, THE TEST LIST MUST STILL BE ENTERED VIA THE CONSOLE!! WHEN RUNNING THE EXERCISOR IN MANUAL MODE A CONTROL C (03 OCTAL) WILL RESULT IN THE TERMINATION OF TESTING AT THE END OF THE CURRENT SUBTEST.

6. ERRORS-GENERAL

6.1 NO OPERATIONAL VT61 ATTACHED

IF THE UNIT SELECTED (IN 'MANUAL' MODE) OR IN THE MAPPING OPERATION ('AUTO' MODE) DOES NOT RESULT IN A UNIT WHICH IS CAPABLE OF RESPONDING TO THE TEST THE MESSAGE 'NO VT61 RESPONDED TO ESCZ SEQ. AUTO RETRY IN 30 SEC'. WILL BE DISPLAYED ON THE CONSOLE EVERY 30 SECONDS UNTIL THE TEST IS STOPPED OR A UNIT RESPONDS.

6.2 EXCESSIVE 'FATAL' ERRORS FROM UNIT UNDER TEST

IF TEN FATAL ERRORS (INCOMPLETE TRANSMIT/RECIEVE CYCLES) OCCURS THE MESSAGE 'TESTING ABORTED-TOO MANY FATAL XMITS' WILL BE DISPLAYED AND THE TEST WILL EXIT TO THE INITIAL SETUP SEQUENCE OF THE REQUESTED MODE. IF THE TEST THEN LOCATES AN OPERATIONAL UNIT, IT WILL BEGIN TESTING IT.

6.3 COMMON ERROR MESSAGES

A. ESCAPE SEQUENCE ERROR (ERROR 1)

THIS ERROR MESSAGE IS RETURNED WHEN A SPECIFIC ESCAPE SEQUENCE DID NOT ELICIT THE EXPECTED RESPONSE FROM THE UNIT UNDER TEST. MESSAGE RETURNS TEST #, ERROR PROGRAM COUNT AND TWO WORDS WHICH CONTAIN UP TO 4 BYTES OF THE FAILING ESCAPE SEQUENCE (I.E. IF "TRANSMIT ALL" FAILED; THE ESC, O, V WOULD BE DISPLAYED IN THE FORMAT BYTE 1+2=015517, BYTE 3+4=000126).

B. RECEIVE STATUS ERROR (ERROR 2)

THIS ERROR MESSAGE IS RETURNED IF ANY OF BITS 12, 13, OR 14 ARE SET IN THE INTERFACE RECEIVE BUFFER REGISTER. DATA DISPLAYED IS THE ADDRESS OF THE CSR (CONTROL AND STATUS REGISTER) OF THE FAILING UNIT, THE CONTENTS OF THE FOREMENTIONED CSR, THE ERROR BITS FROM THE RECEIVE BUFFER REGISTER, AND THE CHARACTER WHICH WAS STORED WHEN THE ERRORS WERE DETECTED.

C. SOFTWARE STATUS (VSTAT) ERROR (ERROR 3)

THE LOCATION TAGGED "VSTAT" IS USED BY THE PROGRAM TO STORE DYNAMIC CONDITIONS RELATING TO THE UNIT UNDER TEST. THE BITS WHICH MAY CAUSE A SOFTWARE STATUS ERROR ARE:

| | |
|--------|--|
| BIT 15 | SET FOR XOFF, CLEARED FOR XON |
| BIT 14 | SET WHEN START OF MESSAGE RECEIVED |
| BIT 13 | SET WHEN END OF MESSAGE RECEIVED |
| BIT 12 | SET FOR A PERIPHERAL ABORT MESSAGE |
| BIT 10 | SET WHEN AN INTERFACE ERROR DETECTED |
| BIT 7 | SET WHEN AN XOFF WAS DETECTED AND THE TRANSMITTER WAS SHUT DOWN BY THE SOFTWARE. |
| BIT 1 | SET WHEN TRANSMIT COMPLETE |

THE ONLY BIT WHICH WILL UNCONDITIONALLY CAUSE THIS ERROR IS BIT 12 (PERIPHERAL ABORT) ALL OTHER BITS WILL BE SET AND RESET AND AN ERROR IS DEPENDENT UPON EXPECTED CONDITIONS (I.E. AFTER A COMPLETE TRANSMISSION BITS 1, 13 AND 14 MUST BE SET AND OTHERS MENTIONED RESET OR AN ERROR WILL BE REPORTED). DATA DISPLAYED IS THE PASS #, THE TEST #, EXPECTED STATUS AND ACTUAL STATUS.

D. VT61 HUNG ERROR (ERROR 11)

THIS ERROR MESSAGE IS DISPLAYED IF A COMPLETE TRANSMISSION(S) DOES NOT RESULT IN A SOM(S), AN EOM(S) AND TRANSMIT DONE. THIS ERROR IS A FATAL ERROR AND TEN OF THESE ERRORS WILL RESULT IN THE TEST ABORTING.

7. RESTRICTIONS

- A. IT IS IMPERATIVE THAT BOTH THE INTERFACE AND THE VT61 SHOULD BE PLACED IN FULL DUPLEX AND REMOTE (NOT LOCAL) MODE.
- B. UNIT TO BE TESTED CANNOT BE THE CONSOLE DEVICE.
- C. FOR THE AUTOMATIC TEST MAPPING OF THE DJ11'S, ALL ADDRESSES FOR THE UNITS TO BE TESTED MUST BE WITHIN THE STANDARD DEC ADDRESSES AND VECTORS. IF THIS IS NOT THE CASE, THE PROCEDURE OUTLINED IN SECTION 8-B MUST BE FOLLOWED BEFORE TESTING IS BEGUN.

8. MISCELLANEOUS

- A. EXECUTION TIME FOR THE AUTO SELECTION TESTS (TEST 1-20) WITH UNITS SET TO A BAUD RATE OF 9600 BAUD IS APPROXIMATELY 90 SECONDS.
- B. TO TEST A DEVICE (DJ11 WITH VT61 ATTACHED) AT NON-STANDARD ADDRESSES THE LOCATION 'STRTAB' CAN BE MODIFIED TO CONTAIN THE LOWEST OF THE NON-STANDARD ADDRESSES AND LOCATION 'ENDTAB' MODIFIED TO CONTAIN THE HIGHEST NON-STANDARD ADDRESS. ALL INTERFACES WITHIN THE NEW ADDRESSES WILL BE MAPPED AND TESTED IF THE PROPER RESPONSES ARE OBTAINED.
- C. TO CHANGE THE NUMBER OF FATAL ERRORS ALLOWED BEFORE TESTING IS ABORTED, LOCATION 'ALWCNT' (LOADED WITH 10) CAN BE MODIFIED TO THE DESIRED COUNT.
- D. ALL TESTS EXCEPT TEST 1 AND TEST 23 ARE RUN IN MAINTENANCE MODE, THEREFORE ALL TRANSMISSIONS FROM THE VT61 ARE EXPECTED TO BE PRECEDED BY A SOM AND TERMINATED WITH A EOM.

9. PROGRAM DESCRIPTION

9.0 INITIALIZATION

IN "AUTO" SEQUENCE MODE THIS SECTION OF THE TEST MAPS ALL DEVICES IN THE PRE-DETERMINED AREAS. DEVICES ARE THEN TESTED FOR INTERRUPT CAPABILITY VIA THE "MAINTENANCE" BIT AND ALL UNITS WHICH DO NOT OR CANNOT RESPOND ARE PURGED FROM THE TABLE. ALL UNITS ARE THEN ISSUED THE "ESCAPE Z" SEQUENCE AND THOSE WHICH DO NOT RESPOND, OR DO NOT RESPOND WITH THE PROPER "IDENT" ARE PURGED. ALL OPERATIONAL UNITS ARE STORED IN A TABLE(DLTBL) AND TESTED SEQUENTIALLY.

9.1 TEST 1 CHECK ALL COMMON ESCAPE SEQUENCES.

THIS TEST ISSUES ALL ESCAPE SEQUENCES AND INSURES THE VT61 HAS NOT FAILED DURING AN ESC SEQUENCE BY ISSUING A ESC Z TO FORCE A VT61 RESPONSE. THE PURPOSE OF THE TEST IS TO ATTEMPT TO INSURE THAT SUBSEQUENT TESTS WILL NOT RESULT IN A "HUNG" UNIT. DATA IS NOT EVALUATED.

ALL ERRORS ARE REPORTED AS ESCAPE SEQUENCE FAILURES(ERROR 1).

9.2 TEST 2 CHECK MAINTENANCE MODE.

ROUTINE TO INSURE ENTERING MAINTENANCE MODE CAUSES SOM AND EOM TO BE APPENDED TO ALL TRANSMITS FROM VT61 UNDER TEST. MAINTENANCE MODE IS ENTERED, THEN AN ESCAPE Z SEQUENCE IS ISSUED TO THE UNIT AND THE RESULTING RESPONSE FROM THE VT61 IS CHECKED FOR SOM/EOM.

ERROR 22 WILL BE ISSUED IF EITHER COMPONENT(SOM/EOM) IS MISSING.

9.3 TEST 3 CHECK DIRECT CURSOR ADDRESSING

THIS TEST INSURES THAT THE CURSOR WILL RESPOND TO DIRECT CURSOR ADDRESSING. THE UNIT IS RESET AND THE CURSOR POSITION IS VERIFIED TO BE HOME. THE CURSOR IS THEN MOVED TO ROW 23 COLUMN 80 AND THE POSITION IS AGAIN VERIFIED.

CURSOR POSITIONING ERRORS(ERROR 7) ARE REPORTED IF THE POSITIONS ARE INCORRECT.

9.4 TEST 4 CHECK LINEAR ADDRESSING MODE.

ROUTINE TO INSURE THE UNIT CAN ENTER LINEAR ADDRESSING MODE. 81 CHARACTERS ARE ISSUED TO THE UNIT UNDER TEST THEN THE CURSOR POSITION IS READ AND MUST BE ROW1, COL.0.

AN ESCAPE SEQUENCE ERROR (ERROR 1) IS ISSUED IF THE CURSOR IS NOT AT ROW1, COL.0

9.5 TEST 5 CHECK XON/XOFF FROM VT61

TEST TO INSURE OPERATION OF XON/XOFF COMMANDS FROM VT61. XOFF IS FORCED BY TRANSMITTING THE DATA ON LINE 23 WHILE SIMULTANEOUSLY FILLING THE SILO WITH NEW DATA. AFTER SENSING THE XOFF, THE TEST WAITS FOR THE TRANSMIT TO FINISH AND INSURES XON OCCURS BEFORE THE MAXIMUM TRANSFER TIME HAS ELAPSED. (30 SECONDS)

ERRORS ARE REPORTED IF THE FORMAT OF ERROR 3(VSTAT ERRORS) AND WILL REFLECT EITHER LACK OR EXCESS OF BIT 15.

9.6 TEST 6 CHECK XON/XOFF TO VT61

ROUTINE TO VERIFY OPERATION OF XOFF AND XON TO THE VT61. A FULL SCREEN TRANSMIT IS INITIATED AND A SERIES OF XOFFS AND XONS ARE ISSUED TO THE TERMINAL SEQUENTIALLY. ERRORS ARE REPORTED IF A XOFF DOES NOT STOP, OR A XON RESTART THE TRANSMISSION. TEST IS ENDED WHEN EOM IS SENSED.

ERRORS ARE REPORTED (ERROR 15 FOR XOFF FAILURE AND ERROR 16 FOR A XON FAILURE) AS SPECIFIC ERROR MESSAGES.

9.7 TEST 7 CHECK RAM AND COMMUNICATIONS PATHS

ROUTINE TO TEST VT61 RAM AND THE COMMUNICATION PATHS. THIS ROUTINE ISSUES A SERIES OF FULL SCREEN PATTERNS (77/100, 100/77, 52/125, INCREMENTING, AND REV. VIDEO INCREMENTING) TO THE VT61. THE FULL SCREEN IS THEN TRANSMITTED TO THE HOST AND AFTER EACH ITERATION RECEIVED DATA IS CHECKED AND ALL ERRORS (INCLUDING TRANSMISSION) ARE REPORTED.

ERRORS REPORTED COULD BE ERROR 2 FOR A RECEIVE STATUS ERROR, ERROR 4 FOR DATA ERRORS AND ERROR 5 FOR A RECEIVE BYTE COUNT ERROR.

9.10 TEST 10 CHECK TRANSMIT AND RECEIVE CHECKSUMS.

ROUTINE TO TEST THE ABILITY OF THE VT61 TO CALCULATE AND TRANSMIT CHECKSUMS OF BOTH TRANSMITTED AND RECEIVED DATA. SUBTEST 'A' TRANSMITS A FULL BUFFER UPDATING A CALCULATED CHECKSUM ON EACH CHARACTER TRANSMITTED. AN ESCAPE SEQUENCE REQUESTING THE RECEIVER CHECKSUM IS EMBEDDED AT THE END OF XMIT BUFFER AND THE RECEIVED CHECKSUM IS COMPARED TO THE CALCULATED. SUBTEST 'B' PERFORMS THE SAME TYPE OF CHECK ON THE VT61 TRANSMIT CHECKSUM, UTILIZING THE DATA SENT TO THE VT61 IN SUBTEST 'A', DURING A FULL SCREEN TRANSMIT.

ERROR 13 IS ISSUED (WITH CALCULATED AND RECEIVED CHECKSUM) IF A RECEIVE CHECKSUM ERROR IS DETECTED. ERROR 14 IS ISSUED (WITH SAME DATA AS ERROR 13) IF A VT61 TRANSMIT CHECKSUM ERROR IS DETECTED.

9.11 TEST 11 CHECK BASIC CURSOR COMMANDS

ROUTINE TO INSURE BASIC CURSOR COMMANDS RESULT IN CORRECT CURSOR MOVEMENT. COMMANDS ARE ISSUED IN THE SEQUENCE: RESET, CURSOR RIGHT, CURSOR DOWN, CURSOR LEFT, AND CURSOR UP. THE READ CURSOR POSITION COMMAND IS ISSUED AFTER EVERY MOVE CURSOR COMMAND AND RECEIVED POSITION IS COMPARED TO THE EXPECTED POSITION AND ANY ERRORS REPORTED.

AN ESCAPE SEQUENCE ERROR (ERROR 1) AND A CURSOR POSITIONING ERROR (ERROR 6) ARE ISSUED IF ANY FUNCTIONS ARE DETECTED TO FAIL.

9.12 TEST 12 CHECK READ CHARACTER AT CURSOR

ROUTINE TO INSURE THAT READ CHARACTER AT CURSOR FUNCTIONS CORRECTLY. COMMAND SEQUENCE IS: RESET, A, CURSOR LEFT, READ CHARACTER AT CURSOR. AN ERROR IS REPORTED IF THE CHARACTER RECEIVED IS NOT AN 'A'.

AN ESCAPE SEQUENCE ERROR (ERROR 1) AND A DATA COMPARE ERROR (ERROR 4) ARE ISSUED IF A FAILURE IS DETECTED.

9.13 TEST 13 CHECK REPLACE AND INSERT CHARACTER MODES

ROUTINE TO VERIFY OPERATION OF REPLACE AND INSERT MODE. INITIALLY ROW 0 IS WRITTEN TO 80 INCREMENTING CHARACTERS; ON THE FIRST PASS (REPLACE MODE) A CHARACTER(172) IS REPLACED AT THE HOME POSITION AND THE CHARACTERS AT ROW0, COL.0 AND ROW1, COL.0 ARE READ AND VERIFIED TO BE A '172' AND A 'NULL' RESPECTIVELY. ON THE SECOND PASS, INSERT MODE IS ENTERED AND THE RESULTING INSERTION (AT THE HOME POSITION) IS VERIFIED. ROW0, COL.0 SHOULD BE '172' AND ROW1, COL.0 SHOULD BE '161'.

- IF AN ERROR IS DETECTED IN EITHER MODE, THE APPROPRIATE ESCAPE SEQUENCE ERROR(ERROR 1) IS ISSUED.

9.14 TEST 14 CHECK VT61 SCROLL CAPABILITIES.

ROUTINE TO INSURE VT61 WILL SCROLL IF A LINE FEED IS ISSUED FROM ROW 23 OR A DATA INSERT FROM ROW 23 COL. 79. IN SUBTEST 'A', ROW 0 IS INITIALLY WRITTEN TO A 0 AND ROW 1 A 1. AFTER COMPLETION OF A LINE FEED (AND RESULTING SCROLL) ROW 00, COL.00 IS EXPECTED TO CONTAIN A 1. IN SUBTEST 'B', THE CURSOR IS PLACED AT ROW23, COL.79 AND A DATA CHARACTER 'A' IS ENTERED. THE CURSOR POSITION IS THEN READ AND SHOULD BE ROW23, COL.00. THE CHAR. AT HOME IS VERIFIED TO BE A NULL.

A SCROLL ERROR(ERROR 23) IS ISSUED IF EITHER FUNCTIONS FAIL TO ELICIT THE PROPER RESPONSE FROM THE UNIT UNDER TEST. THE ERROR PC WILL DISTINGUISH BETWEEN THE FAILING FUNCTIONS.

9.15 TEST 15 CHECK ALL SCREEN ADDRESSES.

THIS TEST INSURES THAT THE VT61 CURSOR CAN BE POSITIONED TO EVERY POSSIBLE ROW/COLUMN POSITION ON THE SCREEN. THIS IS TESTED BY FILLING THE COMPLETE SCREEN (EXCEPT ROW 23, COL.79 WHICH WILL CONTAIN A 'NULL') WITH THE CHARACTER 'A' AND THEN POSITIONING THE CURSOR (VIA DCA) TO EVERY POSITION AND THE 'A' AT THAT POSITION IS REPLACED WITH A SPACE(OCTAL 40). THE SCREEN IS THEN READ TO VERIFY THAT ONLY SPACES EXIST ON THE SCREEN. ALL POSITIONS CONTAINING NON-SPACES ARE REPORTED.

ALL ERRORS DETECTED WILL BE REPORTED AS DIRECT CURSOR ADDRESS ERROS(ERROR 7), AND WILL CONTAIN THE POSITION THE BAD DATA(NON-SPACE) WAS DETECTED AT.

9.16 TEST 16 CHECK LINE FEED AND CARRIAGE RETURN

ROUTINE TO INSURE PROPER OPERATION OF CARRIAGE RETURN AND LINE FEED DURING NORMAL MODE. INITIALLY THE CURSOR IS SET (VIA D.C.A.) TO ROW0, COL 20 AND A LINE FEED IS ISSUED. THE CURSOR POSITION IS THEN READ AND MUST BE ROW1, COL.20. A CARRIAGE RETURN IS THEN ISSUED AND CURSOR POSITION VERIFIED TO BE ROW1, COL0.

AN ESCAPE SEQUENCE ERROR(ERROR 1) AND A CURSOR POSITIONING ERROR(ERROR 6) WILL BE ISSUED IF AN ERROR IS DETECTED.

9.17 TEST 17 CHECK ERASE TO END OF SCREEN

ROUTINE TO VERIFY PROPER OPERATION OF ERASE TO END-OF-SCREEN. SCREEN IS WRITTEN TO 1920 INCREMENTING CHAR. ERASE TO END OF SCREEN IS THEN ISSUED AND THE ENTIRE SCREEN IS READ VERIFYING THAT IT IS ALL NULLS.

IF ANY NON-NULL POSITIONS ARE DETECTED, AND ESCAPE SEQUENCE ERROR (ERROR 1) AND A DATA ERROR(ERROR 4) WILL BE ISSUED.

9.20 TEST 20 CHECK SELF TEST, COPIER, AND ISSUE END OF PASS.

SELF TEST (ESC T) IS ISSUED TO THE UNIT UNDER TEST AND AN SELF TEST ERROR(ERROR 10) IS ISSUED IF THE UNIT CANNOT RESPOND TO AN "ESCAPE Z" SEQUENCE AFTER SELF TEST IS COMPLETE. IF SELF TEST IS SUCCESSFUL THE SCREEN IS WRITTEN TO 23 LINES OF INCREMENTING CHARACTERS AND 23 LINES OF INCREMENTING CHAR. IN REVERSE VIDEO. THE "IDENT" IS THEN CHECKED AND IF A COPIER IS PRESENT A COPY SCREEN COMMAND IS ISSUED (NOTE: THIS COMMAND WILL CAUSE THE UNIT TO BE "BUSY" AND NOT RESPOND TO ANY FURTHER COMMANDS UNTIL THE SCREEN HAS BEEN COMPLETELY COPIED.)

IF THE IDENT INDICATES A COPIER IS PRESENT AND THE COPY SCREEN IS INITIATED,BUT NOT COMPLETED, A "PERIPHERAL ABORT" (ERROR 20) ERROR IS ISSUED.

END OF AUTO-ACCEPTANCE TESTS

9.21 TEST 21 KEYBOARD ECHO TEST

ROUTINE TO ECHO THE KEYBOARD. KEYS FOR TAB, BELL, CARRIAGE AND LINE FEED ECHO A MNEMONIC, NON-DISPLAY CHAR. ECHO OCTAL EQUIVALENTS AND DISPLAY CHAR. ECHO THEMSELVES. (EXAMPLES- CHAR., SPACE, ESC, SPACE OR 037, SPACE.) A CONTROL C (003) WILL CAUSE A TEST EXIT.

9.22 TEST 22 TEST A LINE PRINTER(PRINTER CONTROLLER MODE)

ROUTINE TO UTILIZE THE VT61 AS A PRINTER CONTROLLER. AFTER TEST MESSAGE IS DISPLAYED, THE TEST WAITS FOR A C/R BEFORE ACTUALLY ENTERING TEST. A PATTERN OF INCREMENTING, ROLLING CHAR. WILL BE OUTPUTTED UNTIL A CONTROL C (003) IS RECEIVED.

IF THE LINE PRINTER IS DISABLED AFTER THE INITIALIZATION OF THE TEST, A 'PERIPHERAL ABORT' (ERROR 20) IS ISSUED.

9.23 TEST 23 UNIT SIMULATOR TEST

ROUTINE TO LOOP DATA/COMMANDS FROM THE VT61 BACK TO THE VT61. DATA TRANSMISSIONS RESULTING FROM A ESC SEQUENCE WILL ALSO BE LOOPED AND WILL ENTER THE SCREEN AT THE CURSOR POSITION. THIS TEST CAN BE USED TO SIMULATE, OR CREATE, SPECIFIC SCREEN PATTERNS AND OPERATIONS. A CONTROL C (003) EXITS TEST.

9.24 TEST 24 PRODUCTION KEYBOARD TEST

PRODUCTION KEYBOARD TEST. ALL KEYS MUST BE DEPRESSED IN THE SEQUENCE INDICATED ON THE SCREEN. ALL ERRORS OR MISTAKES ARE DISPLAYED IN OCTAL POSITIONAL FORMAT AND THE CORRECT KEY POSITION IN THE ROW IS DISPLAYED IN DECIMAL. THIS TEST IS RUN IN MAINTENANCE MODE, THEREFORE THE KEYS WILL ECHO THEIR POSITION, NOT THEIR INDICATED MNEMONIC. THE EXCEPTIONS ARE THE INDIVIDUAL TESTS FOR THE SHIF1 AND CONTROL FUNCTIONS. THESE TESTS ARE EXPLICITELY DEFINED BY MESSAGES TO THE OPERATOR. 10 ERRORS WILL CAUSE AN AUTOMATIC EXIT FROM TEST.

10.0 CHANGE HISTORY

NOTE: HISTORY STARTS WITH REV. B0

B0: ADDED CODE TO AUTOCHECK CPU TYPE, AND AUTOMATICALLY INSERT DELAY VALUES COMPATIBLE WITH CPU EXECUTION TIMING.

⌘

```
.NLIST MD,MC,CND
.LIST ME
.TITLE CZVTJBO DJ11 VT61 EXER
:*COPYRIGHT (C) 1975,1980
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MASS. 01754
:*
:*
:*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
:*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
```

642
643
644
645
646
647
648
649
650
651

652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707

```

;*
.SBTTL OPERATIONAL SWITCH SETTINGS
;*
SWITCH          USE
-----          -----
15             HALT ON ERROR
14             LOOP ON TEST
13             INHIBIT ERROR TYPEOUTS
12             INHIBIT TRACE TRAP
11             INHIBIT ITERATIONS
10             BELL ON ERROR
9              LOOP ON ERROR
8              LOOP ON TEST IN SWR<7:0>
.SBTTL BASIC DEFINITIONS

;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
.EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE     ;;BASIC DEFINITION OF SCOPE CALL

;*MISCELLANEOUS DEFINITIONS
HT= 11              ;;CODE FOR HORIZONTAL TAB
LF= 12              ;;CODE FOR LINE FEED
CR= 15              ;;CODE FOR CARRIAGE RETURN
CRLF= 200           ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776         ;;PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774     ;;STACK LIMIT REGISTER
PIRQ= 177772       ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570       ;;HARDWARE SWITCH REGISTER
DDISP= 177570      ;;HARDWARE DISPLAY REGISTER

;*GENERAL PURPOSE REGISTER DEFINITIONS
R0=  %0             ;;GENERAL REGISTER
R1=  %1             ;;GENERAL REGISTER
R2=  %2             ;;GENERAL REGISTER
R3=  %3             ;;GENERAL REGISTER
R4=  %4             ;;GENERAL REGISTER
R5=  %5             ;;GENERAL REGISTER
R6=  %6             ;;GENERAL REGISTER
R7=  %7             ;;GENERAL REGISTER
SP=  %6             ;;STACK POINTER
PC=  %7             ;;PROGRAM COUNTER

;*PRIORITY LEVEL DEFINITIONS
PR0=  0             ;;PRIORITY LEVEL 0
PR1=  40            ;;PRIORITY LEVEL 1
PR2=  100           ;;PRIORITY LEVEL 2
PR3=  140           ;;PRIORITY LEVEL 3
PR4=  200           ;;PRIORITY LEVEL 4
PR5=  240           ;;PRIORITY LEVEL 5
PR6=  300           ;;PRIORITY LEVEL 6
PR7=  340           ;;PRIORITY LEVEL 7

;* 'SWITCH REGISTER' SWITCH DEFINITIONS
SW15= 100000
  
```

C01100

000011
000012
000015
000200
177776
177774
177772
177570
177570

000000
000001
000002
000003
000004
000005
000006
000007
000006
000007

000000
000040
000100
000140
000200
000240
000300
000340

100000

```

708      040000      SW14= 40000
709      020000      SW13= 20000
710      010000      SW12= 10000
711      004000      SW11= 4000
712      002000      SW10= 2000
713      001000      SW09= 1000
714      000400      SW08= 400
715      000200      SW07= 200
716      000100      SW06= 100
717      000040      SW05= 40
718      000020      SW04= 20
719      000010      SW03= 10
720      000004      SW02= 4
721      000002      SW01= 2
722      000001      SW00= 1
723          .EQUIV SW09,SW9
724          .EQUIV SW08,SW8
725          .EQUIV SW07,SW7
726          .EQUIV SW06,SW6
727          .EQUIV SW05,SW5
728          .EQUIV SW04,SW4
729          .EQUIV SW03,SW3
730          .EQUIV SW02,SW2
731          .EQUIV SW01,SW1
732          .EQUIV SW00,SW0
733
734          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
735      100000      BIT15= 100000
736      040000      BIT14= 40000
737      020000      BIT13= 20000
738      010000      BIT12= 10000
739      004000      BIT11= 4000
740      002000      BIT10= 2000
741      001000      BIT09= 1000
742      000400      BIT08= 400
743      000200      BIT07= 200
744      000100      BIT06= 100
745      000040      BIT05= 40
746      000020      BIT04= 20
747      000010      BIT03= 10
748      000004      BIT02= 4
749      000002      BIT01= 2
750      000001      BIT00= 1
751          .EQUIV BIT09,BIT9
752          .EQUIV BIT08,BIT8
753          .EQUIV BIT07,BIT7
754          .EQUIV BIT06,BIT6
755          .EQUIV BIT05,BIT5
756          .EQUIV BIT04,BIT4
757          .EQUIV BIT03,BIT3
758          .EQUIV BIT02,BIT2
759          .EQUIV BIT01,BIT1
760          .EQUIV BIT00,BIT0
761
762          ;*BASIC 'CPU' TRAP VECTOR ADDRESSES
763      000004      ERRVEC= 4          ;;TIME OUT AND OTHER ERRORS
    
```

```

764      000010      RESVEC= 10      ;;RESERVED AND ILLEGAL INSTRUCTIONS
765      000014      TBITVEC=14      ;;'T' BIT
766      000014      TRTVEC= 14      ;;TRACE TRAP
767      000014      BPTVEC= 14      ;;BREAKPOINT TRAP (BPT)
768      000020      IOTVEC= 20      ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
769      000024      PWRVEC= 24      ;;POWER FAIL
770      000030      EMTVEC= 30      ;;EMULATOR TRAP (EMT) **ERROR**
771      000034      TRAPVEC=34      ;;'TRAP' TRAP
772      000060      TKVEC= 60      ;;TTY KEYBOARD VECTOR
773      000064      TPVEC= 64      ;;TTY PRINTER VECTOR
774      000240      PIRQVEC=240    ;;PROGRAM INTERRUPT REQUEST VECTOR
775      .SBTTL TRAP CATCHER
776
777      000000      .=0
778      ;;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
779      ;;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
780      ;;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
781      000174      .=174
782 000174 000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
783 000176 000000      SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
784      .SBTTL ACT11 HOOKS
785
786      ;;*****
787      ;HOOKS REQUIRED BY ACT11
788      $SVPC=.      ;SAVE PC
789      .=46
790 000046 011246      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
791      .=52
792 000052 000000      .WORD 0      ;;2)SET LOC.52 TO ZERO
793      .=$SVPC
794      .=200
795 000200 000137 002274      START:  JMP      AUTO      ;USE AUTO SELECTION OF UNITS
796 000204 000137 002326      MSTRT:  JMP      MANS      ;ALLOW OPERATOR SELECTION OF UNITS/TESTS
  
```

797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840

001100
001100 000000
001102 000
001103 000
001104 000000
001106 000000
001110 000000
001112 000000
001114 000
001115 001
001116 000000
001120 000000
001122 000000
001124 000000
001126 000000
001130 000000
001132 000000
001134 000
001135 000
001136 000000
001140 177570
001142 177570
001144 177560
001146 177562
001150 177564
001152 177566
001154 000
001155 002
001156 012
001157 000
001160 000000
001162 000000
001164 177607
001170 077
001171 015
001172 000012

000377

.SBTTL COMMON TAGS

```

*****
*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
*USED IN THE PROGRAM.
      . =1100
SCMTAG:      . =1100      ;; START OF COMMON TAGS
$PASS:      .WORD      0      ;; CONTAINS PASS COUNT
$STNM:      .BYTE      0      ;; CONTAINS THE TEST NUMBER
$ERFLG:     .BYTE      0      ;; CONTAINS ERROR FLAG
$ICNT:      .WORD      0      ;; CONTAINS SUBTEST ITERATION COUNT
$LPADR:     .WORD      0      ;; CONTAINS SCOPE LOOP ADDRESS
$LPERR:     .WORD      0      ;; CONTAINS SCOPE RETURN FOR ERRORS
$ERTTL:     .WORD      0      ;; CONTAINS TOTAL ERRORS DETECTED
$ITEMB:     .BYTE      0      ;; CONTAINS ITEM CONTROL BYTE
$ERMAX:     .BYTE      1      ;; CONTAINS MAX. ERRORS PER TEST
$ERRPC:     .WORD      0      ;; CONTAINS PC OF LAST ERROR INSTRUCTION
$GDADR:     .WORD      0      ;; CONTAINS ADDRESS OF 'GOOD' DATA
$BDADR:     .WORD      0      ;; CONTAINS ADDRESS OF 'BAD' DATA
$GDADR:     .WORD      0      ;; CONTAINS 'GOOD' DATA
$BDADR:     .WORD      0      ;; CONTAINS 'BAD' DATA
$RESV:     .WORD      0      ;; RESERVED--NOT TO BE USED
$AUTOB:     .BYTE      0      ;; AUTOMATIC MODE INDICATOR
$INTAG:     .BYTE      0      ;; INTERRUPT MODE INDICATOR
$SWR:       .WORD      DSWR      ;; ADDRESS OF SWITCH REGISTER
$DISP:     .WORD      DDISP      ;; ADDRESS OF DISPLAY REGISTER
$TKS:      177560      ;; TTY KBD STATUS
$TKB:      177562      ;; TTY KBD BUFFER
$TPS:      177564      ;; TTY PRINTER STATUS REG. ADDRESS
$TPB:      177566      ;; TTY PRINTER BUFFER REG. ADDRESS
$NULL:     .BYTE      0      ;; CONTAINS NULL CHARACTER FOR FILLS
$FILLS:    .BYTE      2      ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC:    .BYTE     12      ;; INSERT FILL CHARS. AFTER A 'LINE FEED'
$TPFLG:    .BYTE      0      ;; 'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
$TIMES:    0          ;; MAX. NUMBER OF ITERATIONS
$ESCAPE:   0          ;; ESCAPE ON ERROR ADDRESS
$BELL:     .ASCIZ    <207><377><377> ;; CODE FOR BELL
$QUES:     .ASCII    /?/      ;; QUESTION MARK
$CRIF:     .ASCII    <15>     ;; CARRIAGE RETURN
$LF:       .ASCIZ    <12>     ;; LINE FEED
*****
    
```


841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896

```
.SBTTL ERROR POINTER TABLE

;*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:

;*      EM      ;;POINTS TO THE ERROR MESSAGE
;*      DH      ;;POINTS TO THE DATA HEADER
;*      DT      ;;POINTS TO THE DATA
;*      DF      ;;POINTS TO THE DATA FORMAT

$ERRTB:
;GENERAL ESCAPE SEQUENCE ERROR MESSAGE
EM1      ;AN ESCAPE SEQUENCE TO VT61 FAILED.
DH1      ;TEST#,ERROR PC,2 SEQUENCE BYTES,2 SEQUENCE BYTES.
DT0
DF0

;RECEIVE STATUS ERROR MESSAGE
EM2      ;RECEIVE STATUS ERROR
DH2      ;ADDRESS,STATUS ,ERR. BITS,CHAR.
DT2
DF0

;RECIEVE SOFTWARE STATUS ERROR MESSAGE.
EM3      ;SOFTWARE (VSTAT) STATUS ERROR
DH3      ;PASS#,TEST#,GOOD STATUS,RECEIVED STATUS
DT4
DF6

;DATA ERROR
EM4      ;DATA EXPECTED DOES NOT MATCH RECEIVE DATA.
DH4      ;TEST#,REC.CNT.,EXPECTED DATA, RECEIVE DATA
DT5
DF0

;RECEIVE BYTE COUNT ERROR
EM5      ;BYTES EXPECTED DOES NOT EQUAL BYTES RECEIVED.
DH5      ;BYTES EXPECTED, BYTES RECEIVED
DT1
DF2

;GENERAL DIRECT CURSOR ADDRESS FAILURE
EM6      ;CURSOR POSITION ERROR
DH6      ;GD LINE, GD COL., BD LINE, BAD COL.
DT2
```

```
001174
001174 025032
001176 025117
001200 001424
001202 001444
001204 025162
001206 025212
001210 001454
001212 001444
001214 025251
001216 025312
001220 001502
001222 001545
001224 025361
001226 025425
001230 001514
001232 001444
001234 025474
001236 025553
001240 001436
001242 001452
001244 025604
001246 025637
001250 001454
```

| | | | | |
|-----|--------|--------|------|---|
| 897 | 001252 | 001476 | DF3 | |
| 898 | | | | |
| 899 | | | | :DIRECT CURSOR ADDRESS ERROR |
| 900 | | | | |
| 901 | 001254 | 025704 | EM7 | :DIRECT CURSOR ADDRESS ERROR |
| 902 | 001256 | 026004 | DH10 | :PASS#,TEST#,BD. ROW,BD. COL. |
| 903 | 001260 | 001502 | DT4 | |
| 904 | 001262 | 001545 | DF6 | |
| 905 | | | | |
| 906 | | | | |
| 907 | | | | :LAST TEST-SELF TEST FAILED |
| 908 | | | | |
| 909 | 001264 | 026206 | EM10 | :VT61 FAILED SELF-TEST FUNCTION |
| 910 | 001266 | 026563 | DH11 | :CSR, VECTOR |
| 911 | 001270 | 001436 | DT1 | |
| 912 | 001272 | 001450 | DF1 | |
| 913 | | | | |
| 914 | | | | :VT61 FAIL/HUNG ERROR MESSAGE |
| 915 | 001274 | 026043 | EM11 | :LAST TRANSMISSION TO VT61 CAUSED VT61 TO FAIL/HANG |
| 916 | 001276 | 025747 | DH7 | :PASS#,TEST#,ERROR PC |
| 917 | 001300 | 001466 | DT3 | |
| 918 | 001302 | 001536 | DF4 | |
| 919 | | | | |
| 920 | | | | :GENERAL TEST FAILURE-PRECEEDS DATA/POSITION ERROR |
| 921 | | | | |
| 922 | 001304 | 026131 | EM12 | :VT61 UNDERR TEST FAILED-ERROR DATA FOLLOWS |
| 923 | 001306 | 025747 | DH7 | :PASS#,TEST#,ERROR PC. |
| 924 | 001310 | 001466 | DT3 | |
| 925 | 001312 | 001536 | DF4 | |
| 926 | | | | |
| 927 | | | | :RECEIVE CHECKSUM ERROR |
| 928 | | | | |
| 929 | 001314 | 026362 | EM13 | :VT61 RECEIVER CHECKSUM ERROR |
| 930 | 001316 | 026247 | DH12 | :PASS#,TEST#,GD.CKSUM,BD CKSUM |
| 931 | 001320 | 001502 | DT4 | |
| 932 | 001322 | 001545 | DF6 | |
| 933 | | | | |
| 934 | | | | :TRANSMITTER CHECKSUM ERROR |
| 935 | | | | |
| 936 | 001324 | 026431 | EM14 | :VT61 TRANSMITTER CHECKSUM ERROR |
| 937 | 001326 | 026247 | DH12 | |
| 938 | 001330 | 001502 | DT4 | |
| 939 | 001332 | 001545 | DF6 | |
| 940 | | | | |
| 941 | | | | |
| 942 | | | | |
| 943 | | | | :XOFF FAILED TO HALT BLOCK XMIT |
| 944 | | | | |
| 945 | 001334 | 026656 | EM15 | :XOFF TO VT61 FAILED TO HALT BLOCK XMIT |
| 946 | 001336 | 027373 | DH13 | :PASS,TEST,VSTAT |
| 947 | 001340 | 001526 | DT6 | |
| 948 | 001342 | 001536 | DF4 | |
| 949 | | | | |
| 950 | | | | :XON FAILED TO RESTART BLOCK XMIT |
| 951 | | | | |
| 952 | 001344 | 026727 | EM16 | :XON TO VT61 FAILED TO RESTART BLOCK XMIT |

| | | | | | | | |
|------|--------|--------|--------|--------|---------|---------|--|
| 1009 | 001514 | 002266 | 001120 | 001124 | DT5: | .WORD | TSTNM,\$GDADR,\$GDDAT,\$BDDAT,0 |
| 1010 | 001522 | 001126 | 00000C | | | | |
| 1011 | 001526 | 001100 | 002266 | 001120 | DT6: | .WORD | \$PASS,TSTNM,\$GDADR,0 |
| 1012 | 001534 | 000000 | | | | | |
| 1013 | 001536 | 001 | 00C | 000 | DF4: | .BYTE | 1,0,0 |
| 1014 | 001541 | 000 | 00C | 001 | DF5: | .BYTE | 0,0,1,1 |
| 1015 | 001544 | 001 | | | | | |
| 1016 | 001545 | 001 | 000 | 000 | DF6: | .BYTE | 1,0,0,0 |
| 1017 | 001550 | 000 | | | | | |
| 1018 | | 001552 | | | | .EVEN | |
| 1019 | | | | | | | ; INSTRUCTION DEFINITIONS |
| 1020 | | 022626 | | | POP2SP | =22626 | |
| 1021 | | 024646 | | | PUSH2SP | =24646 | |
| 1022 | | | | | | | |
| 1023 | | | | | | | ::***** |
| 1024 | | | | | | | ; DEFINITION SOFTWARE STATUS(VSTAT) REGISTER BITS |
| 1025 | | | | | | | ::***** |
| 1026 | | | | | | | |
| 1027 | | 100000 | | | RXOFF | =100000 | ; SET FOR XOFF, CLEARED FOR XON |
| 1028 | | 040000 | | | RSOM | =040000 | ; SET FOR SOM (START OF MESSAGE). |
| 1029 | | 020000 | | | REOM | =020000 | ; SET FOR EOM (END OF MESSAGE). |
| 1030 | | 010000 | | | PABRT | =010000 | ; SET FOR A PERIPHERAL ABORT. |
| 1031 | | 004000 | | | RSTT | =004000 | ; SET FOR RECEIVE STATUS ERROR. |
| 1032 | | 002000 | | | CKSUM | =002000 | ; SET TO CALCULATE 61 REC. CHECKSUM |
| 1033 | | 001000 | | | FPL | =001000 | ; SET WHEN END OF LINE DETECTED |
| 1034 | | 000400 | | | ESC | =000400 | ; SET WHEN OCTAL 33 RECEIVED. |
| 1035 | | 000200 | | | XMKIL | =000200 | ; SET WHEN TRANSMIT KILLED. |
| 1036 | | 000100 | | | TXSUM | =000100 | ; SET TO CALCULATE 61 XMIT CHECKSUM |
| 1037 | | 000040 | | | REVID | =000040 | ; SET WHEN REVERSE VIDEO MODE RECEIVED. |
| 1038 | | 000020 | | | COMGP | =000020 | ; SET TO CONVERT REC. CHAR. BY -137. |
| 1039 | | 000010 | | | ILLNE | =000010 | ; SET FOR REC. INT. ON NON-SELECTED LINE. |
| 1040 | | 000004 | | | CURPOS | =000004 | ; SET WHEN CURSOR POS. RECEIVED |
| 1041 | | 000002 | | | TRMID | =000002 | ; SET WHEN TERMINAL I.D. RECEIVED. |
| 1042 | | 000001 | | | XMDNE | =000001 | ; SET UPON TRANSMIT COMPLETE |
| 1043 | | | | | | | |
| 1044 | | | | | | | ::***** |
| 1045 | | | | | | | ; DEFINITION OF DJ11 CONTROL BITS |
| 1046 | | | | | | | ::***** |
| 1047 | | | | | | | |
| 1048 | | 100000 | | | TRDY | =100000 | ; XMIT READY |
| 1049 | | 040000 | | | XENA | =040000 | ; XMIT INT. ENABLE |
| 1050 | | 000400 | | | XSCN | =000400 | ; XMIT SCAN ENABLE. |
| 1051 | | 000200 | | | RECDN | =000200 | ; RECEIVER DONE. |
| 1052 | | 000100 | | | RENA | =000100 | ; REC. INT. ENABLE. |
| 1053 | | 000020 | | | BCLR | =000020 | ; MOS CLEAR BUSY. |
| 1054 | | 000010 | | | MCLR | =000010 | ; MOS CLEAR. |
| 1055 | | 000004 | | | MAINT | =000004 | ; MAINTENANCE MODE. |
| 1056 | | 000001 | | | RSCN | =000001 | ; REC. SCAN ENABLE. |
| 1057 | | 040004 | | | TCOMB | =040004 | ; MAINT. MODE AND XMIT INT. ENABLE. |
| 1058 | | 000401 | | | SCAN | =000401 | ; REC. AND XMIT SCAN ENABLES. |
| 1059 | | 100000 | | | RRDY | =100000 | ; REG. 2, REC. BUFFER READY FLAG. |
| 1060 | | | | | | | |
| 1061 | | 003600 | | | TOTCH | =1920. | ; TOTAL CHARACTERS ON SCREEN |
| 1062 | | 003601 | | | TOTC1 | =1921. | ; TOTAL SCREEN +1 |
| 1063 | | | | | | | ::***** |
| 1064 | | | | | | | ; FOLLOWING ARE DJ11 ADDRESS, VECTOR AND LINE STORAGE TABLES |

```

1065      ;:*****
1066 001552 000004 VVECT: .BLKW 4 ;GOOD DJ11 VECTOR TABLE
1067 001562 000004 DJTBL: .BLKW 4 ;GOOD DJ11 ADDRESS TABLE
1068 001572 000020 INTAB: .BLKW 20 ;TABLE OF POSSIBLE DJ11 ADDRESSES
1069 001632 000050 DJLNE: .BLKW 40. ;TABLE OF DJ11 LINESL
1070
1071      ;:*****
1072      ;CURRENT POINTERS FOR ADDRESSES, VECTORS AND LINES
1073      ;:*****
1074 001752 000000 VECPT: .WORD ;VECTOR INDEX
1075 001754 000000 DJAPT: .WORD ;ADDRESS INDEX
1076 001756 000000 LNEPT: .WORD ;DJ11 LINE POINTER.
1077      ;ADDRESS TABLES FOR DJ11 INTERFACES
1078 001760 160010 STRTAB: .WORD 160010 ;BEGINNING OF FLOATING ADD.
1079 001762 164000 ENDTAB: .WORD 164000 ;END OF FLOATING ADD.
1080      ;:*****
1081      ;VT61 ADDRESSES IN TABLE REFLECT UNIT UNDER TEST
1082      ;:*****
1083 001764 000000 VJCSR: .WORD 0 ;DJ11 CONTROL AND STATUS.
1084 001766 000000 VRBUF: .WORD 0 ;RECEIVE DATA BUFFER
1085 001770 000000 VXTCR: .WORD 0 ;XMIT LINE CONTROL.
1086 001772 000000 VXBUF: .WORD 0 ;XMITTER DATA BUFFER
1087 001774 000000 VECT: .WORD 0 ;VECTOR FOR UNIT UNDER TEST
1088 001776 000000 TSTLNE: .WORD 0 ;DJ11 LINE UNDER TEST.
1089 002000 000000 OCTLNE: .WORD 0 ;OCT. EQUIV. OF TSTLNE (BIT8-11)
1090 002002 000000 CRCSR: .WORD 0 ;CONSOLE RECEIVE CSR
1091 002004 000000 CRBUF: .WORD 0 ;CONSOLE DATA BUFFER
1092
1093
1094      ;:*****
1095      ;TABLE OF VT61 COMMAND AND SEQUENCES
1096      ;:*****
1097
1098 .BEL =007
1099 002006 000007 BEL: .WORD 007 ;BELL
1100 .CARRT =015
1101 002010 000015 CARRT: .WORD 015 ;CARRIAGE RETURN
1102 .LNFED =012
1103 002012 000012 LNFED: .WORD 012 ;LINE FEED
1104 .TAB =011
1105 002014 000011 TAB: .WORD 011 ;TAB
1106      ;:*****
1107 002016 000001 .WORD 01 ;TABLE DELIMITER (ESCN)
1108      ;:*****
1109
1110 .CHOM =110
1111 002020 000110 CHOM: .WORD 110 ;HOME CURSOR H
1112
1113 .CRT =103
1114 002022 000103 CRT: .WORD 103 ;CURSOR RIGHT C
1115
1116 .CDWN =102
1117 002024 000102 CDWN: .WORD 102 ;CURSOR DOWN B
1118
1119 .CLFT =104
1120 002026 000104 CLFT: .WORD 104 ;CURSOR LEFT D
    
```


| | | | | | |
|------|--------|--------|-------------------|--|-----------------------------|
| 1121 | | | | | |
| 1122 | | 000101 | .CUP =101 | | |
| 1123 | 002030 | 000101 | CUP: .WORD 101 | | ;CURSOR UP A |
| 1124 | | | | | |
| 1125 | | 000112 | .EOS =112 | | |
| 1126 | 002032 | 000112 | EOS: .WORD 112 | | ;ERASE TO END OF SCREEN J |
| 1127 | | | | | |
| 1128 | | | | | |
| 1129 | | | | | |
| 1130 | 002034 | 000002 | .WORD 2 | | ;TABLE DELIMITER (ESCO) |
| 1131 | | | | | |
| 1132 | | | | | |
| 1133 | | | | | |
| 1134 | | 000101 | .EMAIN =101 | | |
| 1135 | 002036 | 000101 | EMAIN: .WORD 101 | | ;ENTER MAINTENANCE MODE A |
| 1136 | | 000141 | .DMAIN =141 | | |
| 1137 | 002040 | 000141 | DMAIN: .WORD 141 | | ;EXIT MAINTENANCE MODE SA |
| 1138 | | | | | |
| 1139 | | 000105 | .LKKB =105 | | |
| 1140 | 002042 | 000105 | LKKB: .WORD 105 | | ;LOCK KEYBOARD E |
| 1141 | | 000145 | .UNLKKB =145 | | |
| 1142 | 002044 | 000145 | UNLKKB: .WORD 145 | | ;UNLOCK KEYBOARD SE |
| 1143 | | | | | |
| 1144 | | 000103 | .DRECT =103 | | |
| 1145 | 002046 | 000103 | DRECT: .WORD 103 | | ;ENABLE LINEAR MODE C |
| 1146 | | | | | |
| 1147 | | 000133 | .CLRCK =133 | | |
| 1148 | 002050 | 000133 | CLRCK: .WORD 133 | | ;CLEAR RECEIVER CHECKSUM [|
| 1149 | | | | | |
| 1150 | | 000134 | .CLTCK =134 | | |
| 1151 | 002052 | 000134 | CLTCK: .WORD 134 | | ;CLEAR TRANSMITTER CHECKSUM |
| 1152 | | | | | |
| 1153 | | | | | |
| 1154 | | 000112 | .EEMP =112 | | |
| 1155 | 002054 | 000112 | EEMP: .WORD 112 | | ;ENABLE REVERSE VIDEO J |
| 1156 | | 000152 | .DEMP =152 | | |
| 1157 | 002056 | 000152 | DEMP: .WORD 152 | | ;DISABLE REVERSE VIDEO SJ |
| 1158 | | | | | |
| 1159 | | 000137 | .IABT =137 | | |
| 1160 | 002060 | 000137 | IABT: .WORD 137 | | ;INITIALIZE ABORT FLAG - |
| 1161 | | | | | |
| 1162 | | | | | |
| 1163 | 002062 | 000003 | .WORD 3 | | ;TABLE DELIMITER (ESCAPE P) |
| 1164 | | | | | |
| 1165 | | | | | |
| 1166 | | 000131 | .EAPNT =131 | | |
| 1167 | 002064 | 000131 | EAPNT: .WORD 131 | | ;ENABLE AUTO PRINT MODE Y |
| 1168 | | 000171 | .DAPNT =171 | | |
| 1169 | 002066 | 000171 | DAPNT: .WORD 171 | | ;DISABLE AUTO PRINT MODE SY |
| 1170 | | | | | |
| 1171 | | 000111 | .EINST =111 | | |
| 1172 | 002070 | 000111 | EINST: .WORD 111 | | ;ENABLE INSERT I |
| 1173 | | 000151 | .ERPL =151 | | |
| 1174 | 002072 | 000151 | ERPL: .WORD 151 | | ;ENABLE REPLACE SI |
| 1175 | | | | | |
| 1176 | | | | | |

| | | | | | |
|------|--------|--------|---------|--------------|------------------------------|
| 1177 | | | | | |
| 1178 | 002074 | 000004 | | | |
| 1179 | | | | | |
| 1180 | | | | | |
| 1181 | | 054433 | .DCRAD | =054433 | |
| 1182 | 002076 | 054433 | DCRAD: | .WORD 054433 | ;DIRECT CURSOR ADDRESSING |
| 1183 | | 067467 | .R23C79 | =067467 | |
| 1184 | 002100 | 067467 | R23C79: | .WORD 067467 | ;CURSOR TO LOWER RIGHT |
| 1185 | 002102 | 000000 | | .WORD 0 | |
| 1186 | | | | | |
| 1187 | 002104 | 047433 | RCUR: | .WORD 047433 | ;DIRECT CURSOR ADDRESSING |
| 1188 | | 000131 | .Y | =131 | |
| 1189 | | 000131 | .RDCUR | =00131 | |
| 1190 | 002106 | 000131 | RDCUR: | .WORD 00131 | ;READ CURSOR POSITION Y |
| 1191 | 002110 | 000000 | | .WORD 0 | |
| 1192 | | | | | |
| 1193 | | 000117 | .O | =117 | |
| 1194 | 002112 | 047433 | ESCO: | .WORD 047433 | ;ESCAPE O |
| 1195 | | 000126 | .XMTAL | =000126 | |
| 1196 | 002114 | 000126 | XMTAL: | .WORD 000126 | ;TRANSMIT ALL V |
| 1197 | 002116 | 000000 | | .WORD 0 | |
| 1198 | | | | | |
| 1199 | 002120 | 047433 | | .WORD 047433 | ;ESCAPE O |
| 1200 | | 000127 | .TCUCH | =127 | |
| 1201 | 002122 | 000127 | TCUCH: | .WORD 127 | ;XMIT CHARACTER AT CURSOR. W |
| 1202 | 002124 | 000000 | | .WORD 0 | |
| 1203 | | | | | |
| 1204 | 002126 | 047433 | | .WORD 047433 | ;ESCAPE O |
| 1205 | | 000135 | .TXRCK | =135 | |
| 1206 | 002130 | 000135 | TXRCK: | .WORD 135 | ;XMIT RECIEVER CHECKSUM] |
| 1207 | 002132 | 000000 | | .WORD 0 | |
| 1208 | | | | | |
| 1209 | 002134 | 047433 | | .WORD 047433 | ;ESCAPE O |
| 1210 | | 000136 | .TXTCK | =136 | |
| 1211 | 002136 | 000136 | TXTCK: | .WORD 136 | ;XMIT TRANSMITTER CHECKSUM |
| 1212 | 002140 | 000000 | | .WORD 0 | |
| 1213 | | | | | |
| 1214 | 002142 | 147433 | | .WORD 147433 | ;ESCAPE O |
| 1215 | | 000140 | .RABT | =140 | |
| 1216 | 002144 | 000140 | RABT: | .WORD 140 | ;READ THE ABORT FLAG. \ |
| 1217 | 002146 | 000000 | | .WORD 0 | |
| 1218 | | | | | |
| 1219 | | | | | |
| 1220 | 002150 | 177777 | | .WORD -1 | ;END OF TABLE TERMINATOR |
| 1221 | | | | | |
| 1222 | | | | | |
| 1223 | | | | | |
| 1224 | | | | | |
| 1225 | | | | | |
| 1226 | | | | | |
| 1227 | | 000127 | .EPNT | =127 | |
| 1228 | 002152 | 000127 | EPNT: | .WORD 127 | ;ENABLE PRINT MODE. W |
| 1229 | | 000130 | .DPNT | =130 | |
| 1230 | 002154 | 000130 | DPNT: | .WORD 130 | ;DISABLE PRINT MODE X |
| 1231 | | | | | |
| 1232 | | 000135 | .CPYSC | =135 | ;COPY SCREEN] |

| | | | | |
|------|--------|--------|---|--|
| 1233 | | 000136 | .ENAC =136 | ;ENABLE AUTO COPY MODE ESC ^ |
| 1234 | | 000137 | .DISAC =137 | ;DISABLE AUTO COPY MODE ESC - |
| 1235 | | 000150 | .PSCRN =000150 | ;PRINT THE SCREEN H/SH |
| 1236 | | | | |
| 1237 | | | | |
| 1238 | | | ::***** | |
| 1239 | | | ;ESCAPE CODE EQUIVALENCES AND IDENTIFIERS | |
| 1240 | | | ::***** | |
| 1241 | | | | |
| 1242 | | 000033 | .ESC =033 | ;PRIMARY ESCAPE CODE. |
| 1243 | | 000120 | .P =120 | |
| 1244 | 002156 | 050033 | ESCP: .WORD 050033 | ;ESCAPE P |
| 1245 | | 000124 | .TSTER =124 | |
| 1246 | 002160 | 000124 | TSTER: .WORD 124 | ;TEST TERMINAL(ESC O T) |
| 1247 | | 002076 | ESCYI =DCRAD | ;ESCYI EQUALS DCRAD/DCRADI |
| 1248 | | 000057 | SLSH =000057 | ;SLASH CODE FOR TERMINAL IDENT ESC. |
| 1249 | | 000106 | CKGP =106 | ;ENABLE REC.TO SUB 137 FROM ALL REC DATA |
| 1250 | | 000107 | NCKGP =107 | ;ENABLE NORMAL RECEIVED DATA. |
| 1251 | | 000171 | CPABRT =171 | ;COPIER ABORT |
| 1252 | | 000172 | PRABRT =172 | ;PRINTER ABORT |
| 1253 | | 000170 | NABRT =170 | ;NO ABORT SX |
| 1254 | 002162 | 000000 | IDENT: .WORD 0 | ;VT61 IDENT CODE |
| 1255 | | 002112 | ESCOI =ESCO | |
| 1256 | | 002156 | ESCPI =ESCP | |
| 1257 | | 002164 | FSCZI =ESCZ | |
| 1258 | | 055033 | .ESZ =055033 | |
| 1259 | 002164 | 055033 | ESZ: .WORD 055033 | ;OCTAL EQUIV. OF ESZ SEQUENCE |
| 1260 | | 000122 | .RESET =122 | |
| 1261 | 002166 | 000122 | RESET: .WORD 122 | ;VT61 INITIALIZE R |
| 1262 | | | | |
| 1263 | 002170 | 000033 | ESCN: .WORD 000033 | ;ESCAPE N-FLAG |
| 1264 | 002172 | 020041 | R01C00: .WORD 020041 | ;ROW1,COL. 0 |
| 1265 | 002174 | 032041 | R01C20: .WORD 032041 | ;ROW01,COLUMN 20 |
| 1266 | 002176 | 020066 | R22C00: .WORD 020066 | ;ROW22,COL.00 |
| 1267 | 002200 | 020054 | R12C00: .WORD 020054 | ;ROW 12,COLUMN 00 |
| 1268 | | 020067 | .R23C00 =020067 | |
| 1269 | 002202 | 020067 | R23C00: .WORD 020067 | ;ROW23,COL.00 |
| 1270 | | 025440 | .R00C11 =025440 | |
| 1271 | 002204 | 025440 | R00C11: .WORD 025440 | ;ROW,COL.11 |
| 1272 | | 032040 | .R00C20 =032040 | |
| 1273 | 002206 | 032040 | R00C20: .WORD 032040 | ;ROW 0,COLUMN 20 |
| 1274 | 002210 | 024040 | R00C08: .WORD 024040 | ;ROW 00,COLUMN 8 |
| 1275 | 002212 | 020040 | CUHME: .WORD 020040 | ;OCTAL EQUIV. OF CURSOR HOME. |
| 1276 | 002214 | 067440 | R00C80: .WORD 067440 | ;ROW 0, COLUMN 80. |
| 1277 | 002216 | 067067 | R23C78: .WORD 067067 | ;ROW 23,COL. 78. |
| 1278 | | 000040 | .R00 =40 | ;ROW 0 |
| 1279 | | 000041 | .R01 =41 | ;ROW 1 |
| 1280 | | 000054 | .R12 =54 | ;ROW 12 |
| 1281 | | 000066 | .R22 =66 | ;ROW 22 |
| 1282 | | 000067 | .R23 =67 | ;ROW 23 |
| 1283 | | 000040 | .C00 =40 | ;COLUMN 0 |
| 1284 | | 000043 | .C03 =43 | ;COL. 3 |
| 1285 | | 000050 | .C08 =50 | ;COL. 8 |
| 1286 | | 000053 | .C11 =53 | ;COL. 11 |
| 1287 | | 000064 | .C20 =64 | ;COL. 20 |
| 1288 | | 000065 | .C21 =65 | ;COL. 21 |

| | | | | | | | |
|------|--------|--------|---------|--------|------|--|---------------------------------|
| 1289 | | 000110 | .C40 | =110 | | ;COL. 40 | |
| 1290 | | 000157 | .C79 | =157 | | ;COL. 79 | |
| 1291 | | | | | | | |
| 1292 | | | | | | | |
| 1293 | | | | | | | |
| 1294 | | | | | | | |
| 1295 | | | | | | | |
| 1296 | | 000002 | SOM | =02 | | ;START OF MESSAGE | |
| 1297 | | 000004 | EOM | =04 | | ;END OF MESSAGE | |
| 1298 | | 000023 | XOFF | =23 | | ;TURN OFF TRANSMISSION | |
| 1299 | | 000021 | XON | =21 | | ;TURN ON TRANSMISSION | |
| 1300 | 002220 | 000000 | CHRD: | .WORD | 0 | ;STORAGE FOR SINGLE CH. READ | |
| 1301 | 002222 | 000000 | SVER1: | .WORD | | ;TEMP. STORAGE R1. | |
| 1302 | 002224 | 000000 | SVER2: | .WORD | | ;TEMP. STORAGE R2. | |
| 1303 | 002226 | 000000 | ZERO: | .WORD | 0 | ;MUST BE LEFT AS ZERO. | |
| 1304 | 002230 | 003000 | TYP6: | .WORD | 3000 | ;TYPE 6 OCTAL CHAR-NO ZEROS | |
| 1305 | 002232 | 000000 | TSTPTR: | .WORD | 0 | ;TEST POINTER IN MANUAL SELECT MODE | |
| 1306 | 002234 | 000000 | MODE: | .WORD | 0 | ;BYTE0=TESTING MODE,BYTE1=INTERFACE TYPE | |
| 1307 | 002236 | 000000 | FTLCNT: | .WORD | 0 | ;COUNT OF INCOMPLETE XMIT. | |
| 1308 | 002240 | 000012 | ALWCNT: | .WORD | 10. | ;# OF ALLOWABLE INCOMPLETE XMIT. | |
| 1309 | 002242 | 000001 | ONE: | .WORD | 1 | | |
| 1310 | 002244 | 000000 | TOADD: | .WORD | | | |
| 1311 | 002246 | 000000 | BUBCT: | .WORD | | | |
| 1312 | 002250 | 000000 | TPREG: | | 0 | | |
| 1313 | 002252 | 000000 | PRESC: | .WORD | | ;PRIMARY ESC COMMAND | |
| 1314 | 002254 | 000000 | ESSEQ: | .WORD | | ;SEQUENCE ASSEMBLY AREA | |
| 1315 | 002256 | 000000 | DLAY: | .WORD | | | |
| 1316 | 002260 | 000000 | ROSVE: | .WORD | | ;TEMP STORAGE FOR R0 ONLY. | |
| 1317 | 002262 | 000000 | VSTAT: | .WORD | 0 | | |
| 1318 | 002264 | 000000 | BLKM: | .WORD | 0 | ;FLAG LOCATION FOR BLOCK MODE XMIT. | |
| 1319 | 002266 | 000000 | TSTNM: | .WORD | 0 | ;DISPLAY STORAGE FOR TEST NUMBER. | |
| 1320 | | | | | | | |
| 1321 | 002270 | 000000 | PTYPE: | .WORD | 0 | ;PROCESSOR TYPE INDICATOR (REV. B0) | |
| 1322 | | | | | | ;IF 0, THEN 11/45 OR 11/70 | |
| 1323 | | | | | | ;IF -1, THEN 11/35 OR 11/40 | |
| 1324 | | | | | | ;IF 1, THEN 11/04 OR 11/34 | |
| 1325 | 002272 | 000000 | DLYCNT: | .WORD | 0 | ;USED IN WTBGND WAIT LOOP | |
| 1326 | | | | | | | |
| 1327 | | | | | | | |
| 1328 | | | | | | | |
| 1329 | | | | | | | |
| 1330 | | | | | | | |
| 1331 | | | | | | | |
| 1332 | | | | | | | |
| 1333 | | | | | | | |
| 1334 | 002274 | 005037 | 002234 | AUTO: | CLR | MODE | ;ZERO THE MODE SWITCH |
| 1335 | 002300 | 000137 | 012322 | | JMP | SETA | ;DO VECTOR SETUP |
| 1336 | 002304 | 004037 | 013106 | AUTOA: | JSR | RO,TRPVEC | ;GO FIND GOOD DJ11S |
| 1337 | 002310 | 004037 | 013214 | | JSR | RO,CDEV | ;CHECK DJ11S FOUND |
| 1338 | 002314 | 004037 | 013754 | | JSR | RO,INITA | ;INSURE VT61S ON DJ11 |
| 1339 | 002320 | 000137 | 002634 | | JMP | MODCK | ;VT61 PRESENT -BEGIN TESTING |
| 1340 | 002324 | 000767 | | | BR | AUTOA | ;NO VT61 FOUND LOOP IN CHECKING |
| 1341 | | | | | | | |
| 1342 | | | | | | | |
| 1343 | | | | | | | |
| 1344 | | | | | | | |

;MANUAL UNIT AND TEST SELECTION. UNITS CAN BE

```

1345                                     ;SELECTED VIA CONSOLE OR AUTO SELECTION CAN
1346                                     ;BE UTILIZED. TESTS ENTERED VIA CONSOLE WILL
1347                                     ;BE EXECUTED IN THE ORDER ENTERED.
1348
1349                                     ;:*****
1350
1351 002326 012737 000001 002234 MANS:  MOV #1,MODE          ;SET MODE TO MANUAL SELECT.
1352 002334 000137 012322          JMP SETA          ;GO SET UP CONSTANTS
1353 002340 104401 024605          MANS:  TYPE ,DMANA
1354 002344 004037 013106          JSR RO,TRPVEC    ;FIND GOOD DJ11'S
1355 002350 012703 001572          MOV #INTAB,R3
1356 002354 005002          BLDADD: CLR R2
1357
1358 002356 004037 021064          BLDADA: JSR RO,GTNUM ;GET A KEYBOARD INPUT
1359 002362 120127 000054          CMPB R1,#54     ;CHAR. = COMMA?
1360 002366 001004          BNE 1$         ;NO
1361 002370 004037 013042          JSR RO,TMNAD    ;YES-VERIFY THIS ADDRESS,
1362 002374 010223          MOV R2,(R3)+   ;STORE THIS ADDRESS
1363 002376 000766          BR BLDADD      ;AND LOOK FOR ANOTHER ADDRESS.
1364 002400 120137 002012          1$:  CMPB R1,LFED ;CHAR. = LINE FEED?
1365 002404 001025          BNE 3$         ;NO
1366 002406 005702          TST R2         ;ANY ENTRIES CREATED?
1367 002410 001414          BEQ 2$         ;NO USE AUTO SELECTION OF UNITS
1368 002412 004037 013042          JSR RO,TMNAD    ;YES-VERIFY THIS ADDRESS,
1369 002416 010223          MOV R2,(R3)+   ;STORE LAST ADDRESS,
1370 002420 013723 002226          MOV ZERO,(R3)+ ;AND SET A TERMINATOR IN TABLE.
1371 002424 004037 013214          JSR RO,CDEV     ;CHECK DJ11 ON VT 61 SELECTED
1372 002430 005737 001562          TST DJ1BL      ;ANY DJ11S GOOD?
1373 002434 001741          BEQ MANS       ;NO-BACK TO SQUARE ONE
1374 002436 000137 002466          JMP RLDLNE     ;YES- GO GET TESTS
1375 002442 004037 013214          2$:  JSR RO,CDEV  ;CHECK DJ11'S
1376 002446 004037 013754          JSR RO,INITA   ;VERIFY DJ11 HAVE VT61 ATTACHED
1377 002452 000137 002466          JMP BLDLNE     ;BEGIN LINE SELECTION
1378 002456 000730          BR MANS        ;NO UNIT FOUND-LOOP
1379 002460 004037 020762          3$:  JSR RO,OCTBIN ;KEEP BUILDING ADDRESS
1380 002464 000734          BR BLDADA
1381 002466 104401 024735          BLDLNE: TYPE ,DMANL ;TYPE ENTER LINES MESSAGE.
1382 002472 012703 001632          MOV #DJLNE,R3 ;SET FIRST LINE ADDRESS.
1383 002476 005002          BLDLNA: CLR R2
1384 002500 004037 021064          10$: JSR RO,GTNUM   ;GET A KEYBOARD INPUT
1385 002504 120127 000054          CMPB R1,#54   ;CHAR. = COMMA?
1386 002510 001002          BNE 1$        ;NO
1387 002512 010223          MOV R2,(R3)+  ;YES - STORE THIS ADDRESS
1388 002514 000770          BR BLDLNA     ;AND LOOK FOR ANOTHER LINE ENTRY.
1389 002516 120137 002012          1$:  CMPB R1,LFED ;CHAR. = LINE FEED?
1390 002522 001403          BEQ 2$        ;YES-SET TERMINATIONS AND EXIT.
1391 002524 004037 020762          JSR RO,OCTBIN ;NO-KEEP BUILDING ADDRESS
1392 002530 000763          BR 10$
1393 002532 010223          2$:  MOV R2,(R3)+ ;STORE LAST ADDRESS.
1394 002534 012723 177777          MOV #-1,(R3)+ ;STORE END OF ADD. TERMINATOR.
1395 002540 005013          CLR (R3)      ;STORE LAST LINE TERMINATOR
1396
1397 002542 104401 024705          BLDTST: TYPE ,DMANB ;TYPE 2ND PART OF MANUAL MESSAGE
1398 002546 012703 001572          MOV #INTAB,R3 ;USE INTAB AS TEST # STORAGE.
1399 002552 005004          CLR R4        ;CLEAR TEST COUNTER
1400 002554 005002          11$: CLR R2     ;CLEAR ASSEMBL WORD

```

```

1401 002556 004037 021064      10$: JSR    R0,GTNUM      ;GET A NUMERIC CHAR.
1402 002562 120127 000054      CMPB   R1,#54          ;CHAR.=COMMA?
1403 002566 001006                BNE    1$              ;NO
1404 002570 110223                MOVB   R2,(R3)+        ;YES STORE A TEST #
1405 002572 005204                INC    R4              ;AND INCREMENT TEST COUNT.
1406 002574 020437 000040      CMP    R4,32.         ;COUNT =32?
1407 002600 001415                BEQ    MODCK           ;YES ACCEPT NO MORE ENTRIES.
1408 002602 000764                BR     11$             ;NO KEEP LOOKING
1409 002604 120137 002012      1$:  CMPB   R1,LFED      ;CHAR. = LINE FEED?
1410 002610 001006                BNE    2$              ;NO
1411 002612 110223                MOVB   R2,(R3)+        ;LOAD THE LAST TEST
1412 002614 105013                CLRB   (R3)            ;AND INSERT TEST TABLE TERMINATOR
1413 002616 112737 000001 002234  MOVB   #1,MODE         ;SET MODE SWITCH TO MANUAL
1414 002624 000403                BR     MODCK           ;AND BEGIN TESTING.
1415
1416 002626 004037 020762      2$:  JSR    R0,OCTBIN     ;CONVERT CHAR.
1417 002632 000751                BR     10$
1418
1419
1420 ;:*****
1421 ;THIS ROUTINE LOOKS FOR THE OPERATIONAL MODE REQUESTED AND
1422 ;SELECTS THE NEXT UNIT TO BE TESTED.
1423
1424 ;MODE 0 = ACCEPTANCE TYPE TEST
1425 ;MODE 1 = OPERATOR SELECTION OF UNITS AND SEQUENCE OF TESTS.
1426 ;:*****
1427 002634 012737 001562 001754 MODCK: MOV    #DJTBL,DJAPT    ;INITIAL SETUP OF ADDRESS
1428 002642 012737 001552 001752      MOV    #VVECT,VECPT    ;AND VECTOR POINTERS.
1429 002650 012737 001632 001756      MOV    #DJLNE,LNEPT    ;LOAD LINE POINTER.
1430 002656 012701 001764                MODCO: MOV    #VJCSR,R1    ;LOAD ADDRESS DESTINATION
1431 002662 013702 001754                MOV    DJAPT,R2        ;LOAD CURRENT ADDRESS POINTER
1432 002666 017703 177060                MOV    @VECT,R3        ;LOAD CURRENT VECTOR POINTER
1433 002672 0C5712                TST    (R2)            ;ALL UNITS CHECKED?
1434 002674 001013                BNE    1$              ;NO - CONTINUE
1435 002676 005737 002234                TST    MODE            ;CHECK MODE
1436 002702 001002                BNE    10$
1437 002704 000137 002304                JMP    AUTOA           ;GO RESTART AUTO MODE
1438 002710 105777 177316      10$: TSTB   @TSTPTR      ;MANUAL LOOP REQUESTED?
1439 002714 100001                BPL    2$              ;NO
1440 002716 000746                BR     MODCK           ;YES-RESTART COMPLETE TEST.
1441 002720 000137 002340      2$:  JMP    MANSA          ;GO RESTART MANUAL MODE
1442 002724 004037 014314      1$:  JSR    R0,LDADD       ;NO-LOAD NEXT ADDRESSES
1443 002730 010237 001754                MOV    R2,DJAPT        ;SAVE ADDRESS POINTER.
1444 002734 010337 001774                MOV    R3,VECT        ;STORE VECT. OF UNIT UNDER TEST
1445 002740 012723 015256                MOV    #INTRC,(R3)+    ;YES - NOW SET UP RECEIVE VECTOR
1446 002744 012723 000340                MOV    #340,(R3)+     ;AND SET RECEIVER PSW TO 7
1447 002750 012723 016232                MOV    #INTXM,(R3)+   ;SET UP TRANSMIT VECTOR
1448 002754 012723 000340                MOV    #340,(R3)+     ;AND SET PSW TO 7.
1449 002760 017737 176772 001776 MODCA: MOV    @LNEPT,TSTLNE ;LOAD LINE TO UTILIZED.
1450 002766 023727 001776 177777      CMP    TSTLNE,#-1     ;THIS LINE REALLY A SEPARATOR?
1451 002774 001007                BNE    12$             ;NO-TEST IT.
1452 002776 062737 000002 001756      ADD    #2,LNEPT       ;YES-BUMP LINE POINTER ,UPDATE VECTOR
1453 003004 062737 000002 001752      ADD    #2,VECPT       ; POINTER AND GET NEXT ADDRESS.
1454 003012 000721                BR     MODCO
1455 003014 005046                12$: CLR    -(SP)        ;CLEAR THE PSW,LSI11 STYLE.
1456 003016 012746 003024                MOV    #100$,-(SP)
    
```

| | | | | | | | | |
|------|--------|--------|--------|--------|--------|-------|------------------|----------------------------------|
| 1457 | 003022 | 000002 | | | | RTI | | |
| 1458 | 003024 | 012737 | 032171 | 016174 | 100\$: | MOV | #RCRLB+477,REBUF | ;SET UP END OF BUFFER |
| 1459 | 003032 | 012737 | 032671 | 016502 | | MOV | #TCRLB+477,TEBUF | |
| 1460 | 003040 | 012737 | 031472 | 016172 | | MOV | #RCRLB,RBBUF | ;INITIALIZE REC.BUFFER. |
| 1461 | 003046 | 012737 | 032172 | 016500 | | MOV | #TCRLB,TBBUF | ;INITIALIZE TRANSMIT BUFFER. |
| 1462 | 003054 | 004037 | 017532 | | | JSR | RO,RESPTR | ;RESET INTERRUPT POINTERS. |
| 1463 | 003060 | 005037 | 002264 | | | CLR | BLKM | ;CLEAR BLOCK MODE FLAG. |
| 1464 | 003064 | 005037 | 002266 | | | CLR | TSTNM | ;CLEAR CURRENT TEST LOCATION. |
| 1465 | 003070 | 005037 | 022146 | | | CLR | XMZER | ;CLEAR ZERO TRANSMIT FLAG |
| 1466 | 003074 | 005037 | 002262 | | | CLR | VSTAT | ;CLEAR ALL INTERRUPT FLAGS |
| 1467 | 003100 | 004037 | 021004 | | | JSR | RO,CONVLN | ;CONVERT BINARY LINE # TO OCTAL. |
| 1468 | 003104 | 052777 | 000010 | 176652 | | BIS | #MCLR,@VJCSR | ;CLEAR SILO AND UARTS. |
| 1469 | 003112 | 000240 | | | | NOP | | |
| 1470 | 003114 | 013777 | 001776 | 176646 | | MOV | TSTLNE,@VXTCR | ;LOAD THE XMITTER LINE #. |
| 1471 | 003122 | 052777 | 000401 | 176634 | | BIS | #SCAN,@VJCSR | ;ALLOW REC. AND XMIT. SCANS. |
| 1472 | 003130 | 004037 | 016706 | | | JSR | RO,ZFLAG | ;ISSUE ESC Z TO VT61 |
| 1473 | 003134 | 012637 | 002162 | | | MOV | (SP)+,IDENT | ;POP STACK INTO IDENT |
| 1474 | 003140 | 100002 | | | | BPL | 11\$ | ;IF IDENT IS -1,CLEAR IT. |
| 1475 | 003142 | 005037 | 002162 | | | CLR | IDENT | |
| 1476 | 003146 | | | | 11\$: | | | |
| 1477 | 003146 | 012637 | 002220 | | | MOV | (SP)+,CHRD | ;POP STACK INTO CHRD |
| 1478 | 003152 | 001375 | | | | BNE | 11\$ | |
| 1479 | 003154 | 105037 | 002163 | | | CLRB | IDENT+1 | ;CLEAR ALL BUT IDENT BITS. |
| 1480 | 003160 | 104401 | 001171 | | | TYPE | ,\$CRLF | |
| 1481 | 003164 | 104401 | 026504 | | | TYPE | ,DVUNIT | ;ISSUE UNIT UNDER TEST MESSAGE |
| 1482 | 003170 | 013746 | 001764 | | | MOV | VJCSR,-(SP) | ;SAVE VJCSR FOR TYPEOUT |
| 1483 | | | | | | | | ;TYPE THE ADDRESS |
| 1484 | 003174 | 104403 | | | | TYPOS | | ;GO TYPE--OCTAL ASCII |
| 1485 | 003176 | 006 | | | | .BYTE | 6 | ;TYPE 6 DIGIT(S) |
| 1486 | 003177 | 001 | | | | .BYTE | 1 | ;TYPE LEADING ZEROS |
| 1487 | 003200 | 017746 | 176546 | | | MOV | @VECPT,-(SP) | ;SAVE @VECPT FOR TYPEOUT |
| 1488 | | | | | | | | ;TYPE THE VECTOR |
| 1489 | 003204 | 104403 | | | | TYPOS | | ;GO TYPE--OCTAL ASCII |
| 1490 | 003206 | 006 | | | | .BYTE | 6 | ;TYPE 6 DIGIT(S) |
| 1491 | 003207 | 000 | | | | .BYTE | 0 | ;SUPPRESS LEADING ZEROS |
| 1492 | 003210 | 013737 | 002000 | 002220 | | MOV | OCTLNE,CHRD | |
| 1493 | 003216 | 000337 | 002220 | | | SWAB | CHRD | |
| 1494 | 003222 | 013746 | 002220 | | | MOV | CHRD,-(SP) | ;SAVE CHRD FOR TYPEOUT |
| 1495 | | | | | | | | ;TYPE THE LINE |
| 1496 | 003226 | 104403 | | | | TYPOS | | ;GO TYPE--OCTAL ASCII |
| 1497 | 003230 | 006 | | | | .BYTE | 6 | ;TYPE 6 DIGIT(S) |
| 1498 | 003231 | 000 | | | | .BYTE | 0 | ;SUPPRESS LEADING ZEROS |
| 1499 | 003232 | 013746 | 002162 | | | MOV | IDENT,-(SP) | ;SAVE IDENT FOR TYPEOUT |
| 1500 | | | | | | | | ;TYPE THE IDENT |
| 1501 | 003236 | 104403 | | | | TYPOS | | ;GO TYPE--OCTAL ASCII |
| 1502 | 003240 | 006 | | | | .BYTE | 6 | ;TYPE 6 DIGITS |
| 1503 | 003241 | 000 | | | | .BYTE | 0 | ;SUPPRESS LEADING ZEROS |
| 1504 | 003242 | 104401 | 001171 | | | TYPE | ,\$CRLF | ;CARRIAGE RETURN AND LINE FEED |
| 1505 | 003246 | 032737 | 000001 | 002162 | | BIT | #BIT00,IDENT | ;UNIT HAVE A COPIER? |
| 1506 | 003254 | 001402 | | | | BEQ | 20\$ | ;NO |
| 1507 | 003256 | 104401 | 026631 | | | TYPE | ,DCOPYR | ;YES-ISSUE COPIER MESSAGE |
| 1508 | 003262 | 032737 | 000002 | 002162 | 20\$: | BIT | #BIT01,IDENT | ;UNIT HAVE A PRINTER? |
| 1509 | 003270 | 001402 | | | | BEQ | 21\$ | ;NO |
| 1510 | 003272 | 104401 | 026603 | | | TYPE | ,DPRTR | ;YES-ISSUE PRINTER MESSAGE. |
| 1511 | 003276 | 005037 | 002236 | | 21\$: | CLR | F1LCNT | ;CLEAR COUNT OF FATAL XMIT. |
| 1512 | 003302 | 062737 | 000002 | 001756 | | ADD | #2,LNEPT | ;UPDATE LINE POINTER. |


```

1513 003310 012737 032674 032672      MOV      #ABBUF,ABUFP      ;RESET THE REC. DATA POINTER
1514 003316 052777 000100 176440      BIS      #RENA,@VJCSR     ;SET THE REC. INT. ENABLE FOR TESTS
1515 003324 105737 002234                TSTB    MODE              ;CHECK TESTING MODE
1516 003330 001403                BEQ     ASTRT             ;AUTO MODE
1517 003332 012737 001572 002232      MOV      #INTAB,TSTPTR   ;LOAD THE INITIAL TEST NUMBER
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529 003340      ASTRT:
1530
1531 003340 000004      TST1:  SCOPE
1532 003342 012737 000001 001160      MOV      #1,$TIMES       ;;DO 1 ITERATION
1533 003350 012737 003356 001106      MOV      #ESTST,$LPADR   ;;SET SCOPE LOOP ADDRESS
1534
1535 003356 012701 002006      ESTST:  MOV      #BEL,R1      ;POINT TO FIRST COMMAND
1536 003362 042777 000100 176374      BIC      #RENA,@VJCSR     ;CLEAR REC. INT. ENABLE
1537 003370 113737 001102 002266      MOVB    $TSTNM,TSTNM     ;LOAD THE TEST NUMBER.
1538 003376 005037 002252                CLR     PRESC
1539 003402 005004                CLR     R4
1540 003404      ZERST:
1541 003404 013746 002226      MOV      ZERO,-(SP)      ;;PUSH ZERO ON STACK
1542 003410 012702 002252      MOV      #PRESC,R2      ;SET UP SEQUENCE ADDRESS
1543 003414 012103      GCMD:  MOV      (R1)+,R3     ;LOAD THE COMMAND
1544 003416 001405                BEQ     1$              ;IF CHAR. ZERO,MUST BE XMIT TERMINATOR
1545 003420 100537                BMI     ESTEX           ;TABLE EXPENDED - EXIT TEST.
1546 003422 120327 000004      CMPB    R3,#4            ;IS COMMAND ACTUALLY A DELIMITER?
1547 003426 103444                BLO    DELIM           ;YES, GO UPDATE FUNCTIONS
1548 003430 001473                BEQ     SPTN           ;NO, ITS A '10' - SPECIAL CASE.
1549 003432 005704      1$:    TST      R4              ;SEE IF FLAG INDICATING SEQ.
1550 003434 100474                BMI     SEQ4           ;4 IS SET. - YES EXIT
1551 003436 010337 002254      2$:    MOV      R3,ESSEQ    ;PUSH THE SEQUENCE TO BE TESTED
1552 003442      INXMT:
1553 003442 013746 002254      MOV      ESSEQ,-(SP)     ;;PUSH ESSEQ ON STACK
1554 003446 005704                TST     R4              ;DOES THIS SEQUENCE REQUIRE
1555 003450 001402                BEQ     3$              ;ADDITIONAL ESC?
1556 003452 013746 002252      MOV      PRESC,-(SP)    ;;PUSH PRESC ON STACK
1557
1558 003456 004037 014606      3$:    JSR      R0,TEC     ;GO TRANSMIT THIS SEQUENCE.
1559
1560 003462 005704      4$:    TST      R4              ;IN I/O SEQUENCES?
1561 003464 100011                BPL     40$            ;NO
1562 003466 012737 000054 020514      MOV      #4.,DCOUNT     ;YES,SET UP TO DELAY 1+ SEC.
1563 003474 004037 020452                JSR     R0,DELAY
1564 003500 032777 100000 176260      BIT      #RRDY,@VRBUF   ;CLEAR THE SILO
1565 003506 001374                BNE     .-6
1566
1567 003510 004037 016706      40$:   JSR      R0,ZFLAG      ;ISSUE ESC Z SEQUENCE-GET IDENT
1568 003514      5$:

```

| | | | | | | | | |
|------|--------|--------|--------|--------|--------|-------|---------------|---|
| 1569 | 003514 | 012637 | 002220 | | | MOV | (SP)+,CHRD | ::POP STACK INTO CHRD |
| 1570 | 003520 | 123737 | 002220 | 002162 | | CMPB | CHRD,IDENT | ::HAVE WE POPPED THE IDENT? |
| 1571 | 003526 | 001045 | | | | BNE | T1ERR | ::NO-ERROR CONDITION |
| 1572 | 003530 | | | | POPIT: | | | |
| 1573 | 003530 | 012637 | 002220 | | | MOV | (SP)+,CHRD | ::POP STACK INTO CHRD |
| 1574 | 003534 | 001375 | | | | BNE | -4 | |
| 1575 | 003536 | 000722 | | | | BR | ZERST | ::GET NEXT COMMAND |
| 1576 | | | | | | | | |
| 1577 | 003540 | 120327 | 000001 | | DELIM: | CMPB | R3,#1 | |
| 1578 | 003544 | 001407 | | | | BEQ | 1\$ | ::FIRST DELIMITER - SET ESCN |
| 1579 | | | | | | | | |
| 1580 | 003546 | 120327 | 000002 | | | CMPB | R3,#BIT01 | ::SECOND DELIMITER - SET ESCO |
| 1581 | 003552 | 001412 | | | | BEQ | 2\$ | |
| 1582 | | | | | | | | |
| 1583 | 003554 | 120327 | 000003 | | | CMPB | R3,#3 | ::THIRD DELIMITER - SET ESCP |
| 1584 | 003560 | 001413 | | | | BEQ | 3\$ | |
| 1585 | 003562 | 000714 | | | | BR | GCMD | ::INVALID CHARACTER - GET ANOTHER |
| 1586 | | | | | | | | |
| 1587 | 003564 | 012704 | 000001 | | 1\$: | MOV | #1,R4 | ::SET FIRST DELIMITER FLAG. |
| 1588 | 003570 | 013737 | 002170 | 002252 | | MOV | ESCN,PRES | ::INSERT ESCN. |
| 1589 | 003576 | 000706 | | | | BR | GCMD | |
| 1590 | | | | | | | | |
| 1591 | 003600 | 013737 | 002112 | 002252 | 2\$: | MOV | ESCO,@#PRES | ::INSERT ESCO |
| 1592 | 003606 | 000702 | | | | BR | GCMD | |
| 1593 | | | | | | | | |
| 1594 | 003610 | 013737 | 002156 | 002252 | 3\$: | MOV | ESCP,@#PRES | ::INSERT ESCP |
| 1595 | 003616 | 000676 | | | | BR | GCMD | |
| 1596 | | | | | | | | |
| 1597 | 003620 | 012704 | 177777 | | SPTN: | MOV | #-1,R4 | ::SET FLAG INDICATING I/O |
| 1598 | 003624 | 000673 | | | | BR | GCMD | ::SEQUENCES. |
| 1599 | | | | | | | | |
| 1600 | 003626 | 005703 | | | SEQ4: | TST | R3 | ::CHECK IF COMMAND = 0 |
| 1601 | 003630 | 001704 | | | | BEQ | INXMT | ::YES, COMPLETE SEQUENCE ASSEMBLED |
| 1602 | 003632 | 110322 | | | | MOVB | R3,(R2)+ | ::NO - KEEP ASSEMBLING |
| 1603 | 003634 | 000303 | | | | SWAB | R3 | ::POSITION HIGH ORDER BIT |
| 1604 | 003636 | 110322 | | | | MOVB | R3,(R2)+ | ::AND ASSEMBLE IT |
| 1605 | 003640 | 000665 | | | | BR | GCMD | ::GET ANOTHER BYTE |
| 1606 | | | | | | | | |
| 1607 | 003642 | 004037 | 017110 | | T1ERR: | JSR | RO,CLREG | |
| 1608 | 003646 | 013737 | 002252 | 001124 | | MOV | PRES,\$GDDAT | ::AND INSERT IN ERROR |
| 1609 | 003654 | 000337 | 001124 | | | SWAB | \$GDDAT | ::REASSEMBLE FAILING SEQUENCES |
| 1610 | 003660 | 013737 | 002254 | 001126 | | MOV | ESSEQ,\$BDDAT | |
| 1611 | 003666 | 105737 | 002255 | | | TSTB | ESSEQ+1 | ::IF UPPER BYTE IS CLEAR DO NOT SWAP |
| 1612 | 003672 | 001402 | | | | BEQ | 1\$ | |
| 1613 | 003674 | 000337 | 001126 | | | SWAB | \$BDDAT | ::MESSAGE 1 |
| 1614 | 003700 | 104001 | | | 1\$: | ERROR | 1 | ::ISSUE ERROR MESSAGE |
| 1615 | 003702 | 005237 | 002236 | | | INC | FTLCNT | ::INCREMENT FATAL XMIT COUNT. |
| 1616 | 003706 | 023737 | 002236 | 002240 | | CMP | FTLCNT,ALWCNT | ::FATAL XMIT EXCEEDED ALLOWED? |
| 1617 | 003714 | 103003 | | | | BHIS | FTEXT | ::YES-EXIT. |
| 1618 | 003716 | 000704 | | | | BR | POPIT | ::CLEAR THE STACK AND TRY ANOTHER COMMAND |
| 1619 | 003720 | | | | ESTEX: | | | |
| 1620 | 003720 | 012637 | 002220 | | | MOV | (SP)+,CHRD | ::POP STACK INTO CHRD |
| 1621 | 003724 | 052777 | 000100 | 176032 | FTEXT: | BIS | #RENA,@VJCSR | ::SET THE REC. INT. ENABLE FOR TESTS |
| 1622 | | | | | | | | |
| 1623 | | | | | | | | |
| 1624 | | | | | | | | |

::*****
 ;ROUTINE TO INSURE ENTERING MAINTENANCE MODE CAUSES SOM AND

```

1625                                     ;EOM TO BE APPENDED TO ALL TRANSMITS FROM VT61 UNDER TEST.
1626                                     ;MAINTENANCE MODE IS ENTERED , THEN AN ESCAPE Z SEQUENCE
1627                                     ;IS ISSUED TO THE UNIT AND THE RESULTING TRANSMISSION IS
1628                                     ;CHECKED OF SOM/EOM.
1629                                     ;:*****
1630                                     ;:*****
1631                                     ;:*****
1632 003732 000004 TST2: SCOPE
1633 003734 012737 000005 001160 MOV #5,$TIMES ;;DO 5 ITERATIONS
1634 003742 012737 003750 001106 MOV #CKMNT,$LPADR ;;SET SCOPE LOOP ADDRESS
1635
1636 003750 004037 016510 CKMNT: JSR R0,RESETV ;RESET THE UNIT AND SETMAINT. MODE.
1637 003754 112777 000002 012522 MOVB #SOM,@TBUF ;XMIT THE START OF MESSAGE.
1638 003762 004037 017406 JSR R0,XMIT1
1639 003766 113777 002164 012510 MOVB ESCZ,@TBUF
1640 003774 004037 017406 JSR R0,XMIT1 ;SEND AN IDENT REQUEST.
1641 004000 113777 002165 012476 MOVB ESCZ+1,@TBUF
1642 004006 004037 017406 JSR R0,XMIT1
1643 004012 112777 000004 012464 MOVB #EOM,@TBUF ;XMIT END OF MESSAGE.
1644 004020 004037 017406 JSR R0,XMIT1
1645 004024 005037 002256 CLR DLAY ;SET UP SOM DELAY OF 100M.S.
1646 004030 032737 040000 002262 1$: BIT #RSOM,VSTAT ;RECEIVED THE START OF MESSAGE?
1647 004036 001003 BNE CKEOM ;YES-GO LOOK FOR EOM.
1648 004040 005337 002256 DEC DLAY ;NO-RUN TIMEOUT DELAY
1649 004044 001371 BNE 1$ ;AND KEEP LOOKING.
1650
1651 004046 012701 000062 CKEOM: MOV #50.,R1 ;SET MAX DELAY FOR 500 M.S.
1652 004052 032737 020000 002262 1$: BIT #REOM,VSTAT ;RECEIVED END OF MESSAGE?
1653 004060 001007 BNE 10$ ;YES-CHECK FOR BOTH RECEIVED.
1654 004062 012737 000001 020514 MOV #1,DCOUNT ;DELAY FOR 10 M..S.
1655 004070 004037 020452 JSR R0,DELAY
1656 004074 005301 DEC R1 ;AND KEEP LOOKING.
1657 004076 001365 BNE 1$
1658 004100 032737 040000 002262 10$: BIT #RSOM,VSTAT ;RECEIVED SOM?
1659 004106 001404 BEQ 2$ ;NO ISSUE ERROR
1660 004110 032737 020000 002262 BIT #REOM,VSTAT ;RECEIVED EOM?
1661 004116 001007 BNE EXMNT ;YES, NO ERRORS-EXIT.
1662 004120 012737 006001 001124 2$: MOV #6001,$GDDAT ;LOAD ERROR WITH EXPECTED
1663 004126 013737 002262 001126 MOV VSTAT,$BDDAT ;AND ACTUAL STATUS.
1664 004134 104022 ERROR 22
1665
1666 004136 000240 EXMNT: NOP
1667
1668 ;:*****
1669 ;THIS TEST INSURES THAT THE CURSOR WILL RESPOND
1670 ;TO DIRECT CURSOR ADDRESSING. THE UNIT IS RESET AND THE CURSOR
1671 ;POSITION IS VERIFIED TO BE HOME . THE CURSOR IS THEN MOVED
1672 ;TO POSITION ROW 23 COLUMN 80 AND THE POSITION IS AGAIN
1673 ;VERIFIED. ERRORS ARE REPORTED IF THE POSITIONS ARE INCORRECT.
1674 ;:*****
1675
1676 ;:*****
1677 ;:*****
1678 004140 000004 TST3: SCOPE
1679 004142 012737 000005 001160 MOV #5,$TIMES ;;DO 5 ITERATIONS
1680 004150 012737 004156 001106 MOV #CURS1,$LPADR ;;SET SCOPE LOOP ADDRESS
    
```

```

1681
1682 004156 013701 016500 CURS1: MOV TBBUF,R1 ;USE R1 AS XMIT BUFFER POINTER.
1683 004162 004037 016510 JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
1684 004166 013721 002112 MOV ESCOI,(R1)+ ;CLFT. RESET, READ CURSOR
1685 004172 113721 002106 MOV RDCUR,(R1)+ ;POSITION, CURSOR LEFT.
1686 004176 012737 000003 016506 MOV #3,XMCNT ;XMIT 3 BYTES
1687
1688 004204 004037 017132 JSR R0,VMREC ;XMIT AND RECEIVE.
1689 004210 000402 BR 10$ ;NORMAL EXIT.
1690 004212 104011 ERROR 11 ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
1691 004214 000446 BR 2$ ;EXIT TEST.
1692 004216 013701 031472 10$: MOV RCRLB,R1 ;GET THE CURRENT CURSOR POSITION.
1693 004222 020137 002212 CMP R1,CUHME ;CURSOR REALLY HOME?
1694 004226 001405 BEQ 1$ ;YES EXIT
1695 004230 104012 ERROR 12 ;VT61 FAILURE MESSAGE
1696 004232 013746 002212 MOV CUHME,-(SP) ;PUSH CUHME ON STACK
1697 004236 004037 017612 JSR R0,CURER ;GO LOAD AND ISSUE CURSOR ERROR
1698
1699 004242 013701 016500 1$: MOV TBBUF,R1 ;LOAD XMIT BUFFER WITH
1700 004246 013721 002076 MOV DCRAD,(R1)+
1701 004252 013721 002100 MOV R23C79,(R1)+ ;CURSOR TO ROW 23, COL.79
1702 004256 013721 002112 MOV ESCOI,(R1)+ ;READ CURSOR POSITION
1703 004262 013721 002106 MOV RDCUR,(R1)+ ;IT AND CURSOR RIGHT
1704 004266 012737 000007 016506 MOV #7,XMCNT ;XMIT 7 BYTES.
1705 004274 004037 017132 JSR R0,VMREC ;XMIT AND RECEIVE
1706 004300 000402 BR 20$ ;NORMAL EXIT.
1707 004302 104011 ERROR 11 ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
1708 004304 000412 BR 2$ ;EXIT TEST.
1709 004306 012701 031472 20$: MOV #RCRLB,R1
1710
1711 004312 023711 002100 CMP R23C79,(R1) ;CHECK CURSOR POSITION TO LOWER RT.
1712 004316 001405 BEQ 2$ ;OK, EXIT
1713 004320 104012 ERROR 12 ;VT61 FAILURE MESSAGE
1714 004322 013746 002100 MOV R23C79,-(SP) ;PUSH R23C79 ON STACK
1715 004326 004037 017612 JSR R0,CURER ;LOAD AND ISSUE CURSOR ERROR .
1716 004332 000240 2$: NOP
1717
1718 ;:*****
1719 ;ROUTINE TO INSURE THE UNIT CAN ENTER LINEAR ADDRESSING
1720 ;MODE. 81 CHARACTERS ARE ISSUED TO THE UNIT UNDER TEST
1721 ;THEN THE CURSOR POSITION IS READ AND MUST BE ROW1,COL.0.
1722 ;:*****
1723 ;:*****
1724 004334 000004 TST4: SCOPE
1725 004336 012737 000005 001160 MOV #5,$TIMES ;:DO 5 ITERATIONS
1726 004344 012737 004352 001106 MOV #CKLIN,$LPADR ;:SET SCOPE LOOP ADDRESS
1727
1728 004352 004037 016510 CKLIN: JSR R0,RESETV ;RESET THE UNIT-SET MAINT AND LINEAR MODES
1729 004356 013701 016500 MOV TBBUF,R1
1730 004362 012703 000120 MOV #80.,R3
1731 004366 004037 020516 JSR R0,BLDINC ;LOAD XMIT BUFFER WITH 80 CHAR AND
1732 004372 013721 002104 MOV RCUR,(R1)+
1733 004376 013721 002106 MOV RDCUR,(R1)+ ;READ CURSOR POSINION.
1734 004402 012737 000123 016506 MOV #83.,XMCNT
1735 004410 004037 017132 JSR R0,VMREC ;XMIT THE BUFFER.
1736 004414 000402 BR 1$
  
```

```

1737 004416 104011          ERROR 11          ;LAST XMIT CAUSED UNIT TO HANG.
1738 004420 000421          BR LINXT          ;EXIT TEST
1739 004422 023777 002172 011542 1$: CMP R01C00,@RBBUF ;CURSOR AT ROW1,COL. 0?
1740 004430 001415          BEQ LINXT          ;YES-EXIT
1741 004432 013737 002112 001124  MOV ESCO,$GDDAT
1742 004440 000337 001124  SWAB $GDDAT
1743 004444 013737 002046 001126  MOV DRECT,$BDDAT ;ISSUE ESC SEQUENCE AND CURSOR
1744 004452 104001          ERROR 1
1745 004454 013746 002172  MOV R01C00,-(SP) ;;PUSH R01C00 ON STACK
1746 004460 004037 017612  JSR RO,CURER
1747 004464 000240          LINXT: NOP
1748
1749
1750 ;:*****
1751 ;TEST TO INSURE OPERATION OF XON/XOFF COMMANDS
1752 ;FROM VT61. XOFF IS FORCED BY TRANSMITTING LINE 23 WHILE SIMUL-
1753 ;TANEOUSLY FILLING THE SILO WITH DATA. AFTER SENSING
1754 ;THE XOFF, THE TEST WAITS FOR THE TRANSMIT TO FINISH AND
1755 ;INSURES XON OCCURS BEFORE THE MAX. TRANSFER TIME HAS ELAPSED.
1756 ;(30 SECONDS)
1757 ;:*****
1758
1759 ;:*****
1760 004466 000004          TST5: SCOPE
1761 004470 012737 000010 001160  MOV #10,$TIMES ;;DO 10 ITERATIONS
1762 004476 012737 004504 001106  MOV #BASC3,$LPADR ;;SET SCOPE LOOP ADDRESS
1763 004504 013701 016500          BASC3: MOV TBBUF,R1 ;R1 = 1ST XMIT BUFFER ADDRESS.
1764 004510 012737 001001 002264  MOV #1001,BLKM ;SET XMIT TO SOM- DATA -EOM.
1765 004516 005037 002262          CLR VSTAT
1766 004522 004037 016510          JSR RO,RESETV ;RESET THE UNIT AND WAIT FOR XON.
1767 004526 013721 002076          MOV DCRAD,(R1)+
1768 004532 013721 002202          MOV R23C00,(R1)+ ;CURSOR TO ROW 23, COL.0
1769 004536 013721 002112          MOV ESCO,(R1)+
1770 004542 013721 002114          MOV XMTAL,(R1)+ ;TRANSMIT THE LINE.
1771 004546 012703 000050          MOV #40.,R3
1772 004552 004037 020516          JSR RO,BLDINC ;40 CHAR. OF INCREMENTING CHAR.
1773 004556 012737 000057 016506  MOV #47.,XMCNT ;SET UP TO XMIT 47 BYTES
1774 004564 052777 040000 175172  BIS #XENA,@VJCSR ;TRANSMIT ENABLES
1775 004572 012703 000050          MOV #40.,R3 ;MAXIMUM DELAY EQUAL 400 M.S.
1776 004576 012737 000001 020514 2$: MOV #1,DCount
1777 004604 004037 020452          JSR RO,DELAY ;DELAY FOR 10 MILLISEC.
1778 004610 032737 100000 002262  BIT #RXOFF,VSTAT ;CHECK FOR XOFF
1779 004616 001007          BNE 3$ ;FOUND IT EXIT THIS SECTION.
1780 004620 005303          DEC R3 ;DELAYED 400 M.S.?
1781 004622 001365          BNE 2$ ;NO-KEEP LOOKING FOR XOFF.
1782 004624 104012          ERROR 12 ;GENERAL VT61 FAILURE MESSAGE
1783 004626 012746 100000          MOV #100000,-(SP) ;;PUSH #100000 ON STACK
1784 004632 004037 016746          JSR RO,CKSFT ;GO REPORT ERROR
1785 004636
1786 004636 012746 000001          3$: MOV #XMDNE,-(SP) ;;PUSH #XMDNE ON STACK
1787 ;:*****
1788 004642 005737 002270          TST PTYPE ;PROCESSOR TYPE?
1789 004646 001407          BEQ 8$ ;BR IF 11/45 OR 11/70
1790 004650 100403          BMI 7$ ;BR IF 11/35 OR 11/40
1791 004652 012746 000031          MOV #25.,-(SP) ;LOAD FOR 11/04 OR 11/34
1792 004656 000405          BR 10$ ;
    
```

(REV. B0)

| | | | | | | | | | |
|------|--------|--------|--------|--------|----------|-------|-----------------|--|---|
| 1793 | 004660 | 012746 | 000062 | | 7\$: | MOV | #50.,-(SP) | | :LOAD FOR 11/35 OR 11/40 |
| 1794 | 004664 | 000402 | | | | BR | 10\$ | | :LOAD FOR 11/45 OR 11/70 |
| 1795 | 004666 | 012746 | 000144 | | 8\$: | MOV | #100.,-(SP) | | |
| 1796 | 004672 | 004037 | 022150 | | 10\$: | JSR | RO,WTBGND | | |
| 1797 | | | | | ;***** | | | | |
| 1798 | 004676 | 000411 | | | | BR | EXIT3 | | :TIMEOUT-EXIT TEST. |
| 1799 | 004700 | 127727 | 025766 | 000021 | | CMPB | @ABUFP,#XON | | :RECEIVED A XON? |
| 1800 | 004706 | 001405 | | | | BEQ | EXIT3 | | :YES-NO ERROR-EXIT |
| 1801 | | | | | | | | | |
| 1802 | 004710 | 104012 | | | | ERROR | 12 | | :GENERAL VT61 FAILURE MESSAGE |
| 1803 | 004712 | 012746 | 000001 | | | MOV | #000001, -(SP) | | :PUSH #000001 ON STACK |
| 1804 | 004716 | 004037 | 016746 | | | JSR | RO,CKSFT | | |
| 1805 | 004722 | 004037 | 017532 | | EXIT3: | JSR | RO,RESPTR | | :RESET INTERRUPT POINTERS. |
| 1806 | | | | | | | | | |
| 1807 | | | | | ;***** | | | | |
| 1808 | | | | | | | | | :ROUTINE TO VERIFY OPERATION OF XOFF AND XON TO THE VT61. |
| 1809 | | | | | | | | | :A FULL SCREEN TRANSMIT IS INITIATED AND A SERIES OF XOFF AND |
| 1810 | | | | | | | | | :XON ARE ISSUED TO THE TERMINAL SEQUENTIALLY. |
| 1811 | | | | | | | | | :ERRORS ARE REPORTED IF XOFF DOES NOT STOP,OR XON RESTART |
| 1812 | | | | | | | | | :THE TRANSMISSION. TEST IS ENDED WHEN EOM IS SENSED. |
| 1813 | | | | | ;***** | | | | |
| 1814 | | | | | | | | | |
| 1815 | | | | | ;***** | | | | |
| 1816 | 004726 | 000004 | | | TST6: | SCOPE | | | |
| 1817 | 004730 | 012737 | 000001 | 001160 | | MOV | #1,\$TIMES | | :DO 1 ITERATION |
| 1818 | 004736 | 012737 | 004744 | 001106 | | MOV | #ONOF61,\$LPADR | | :SET SCOPE LOOP ADDRESS |
| 1819 | | | | | | | | | |
| 1820 | 004744 | 004037 | 016510 | | ONOF61: | JSR | RO,RESETV | | :RESET THE UNIT AND WAIT FOR XON. |
| 1821 | 004750 | 042737 | 077577 | 002262 | | BIC | #77577,VSTAT | | :CLEAR THE FLAGS |
| 1822 | 004756 | 013746 | 002226 | | | MOV | ZERO, -(SP) | | :PUSH ZERO ON STACK |
| 1823 | 004762 | 013746 | 002114 | | | MOV | XMTAL, -(SP) | | :PUSH XMTAL ON STACK |
| 1824 | 004766 | 013746 | 002112 | | | MOV | ESCO, -(SP) | | :PUSH ESCO ON STACK |
| 1825 | 004772 | 004037 | 014606 | | | JSR | RO,TEVC | | |
| 1826 | 004776 | 012737 | 000010 | 020514 | ONOFFLP: | MOV | #10,DCOUNT | | :ALLOW 100 M.S. FOR OPERATION |
| 1827 | 005004 | 004037 | 020452 | | | JSR | RO,DELAY | | :TO BEGIN. |
| 1828 | 005010 | 112777 | 000023 | 011466 | | MOV | #XOFF,@TBUFP | | |
| 1829 | 005016 | 004037 | 017406 | | | JSR | RO,XMIT1 | | :SEND A XOFF TO VT61. |
| 1830 | 005022 | 012704 | 000036 | | | MOV | #30.,R4 | | |
| 1831 | 005026 | 013705 | 032672 | | OFFLP: | MOV | ABUFP,R5 | | :ALLOW 300M.S. FOR XMIT TO CEASE |
| 1832 | 005032 | 012737 | 000001 | 020514 | | MOV | #1,DCOUNT | | |
| 1833 | 005040 | 004037 | 020452 | | | JSR | RO,DELAY | | |
| 1834 | 005044 | 023705 | 032672 | | | CMP | ABUFP,R5 | | |
| 1835 | 005050 | 001406 | | | | BEQ | ONOFFA | | :XMIT STOPPED-GO RESTART IT. |
| 1836 | 005052 | 005304 | | | | DEC | R4 | | |
| 1837 | 005054 | 001364 | | | | BNE | OFFLP | | :COUNTER NO EQUAL 300 MS-LOOP |
| 1838 | 005056 | 013737 | 002262 | 001120 | | MOV | VSTAT,\$GDADR | | :UNIT DID NOT RESPOND TO XOFF |
| 1839 | 005064 | 104015 | | | | ERROR | 15 | | :ISSUE ERROR |
| 1840 | | | | | | | | | |
| 1841 | 005066 | 112777 | 000021 | 011410 | ONOFFA: | MOV | #XON,@TBUFP | | :SEND A XON TO THE VT61. |
| 1842 | 005074 | 004037 | 017406 | | | JSR | RO,XMIT1 | | |
| 1843 | 005100 | 012704 | 000036 | | | MOV | #30.,R4 | | :SET UP FOR 300MS DELAY. |
| 1844 | 005104 | 032737 | 020000 | 002262 | ONLFP: | BIT | #REOM,VSTAT | | :EOM RECEIVED? |
| 1845 | 005112 | 001020 | | | | BNE | ONOFFXT | | :YES-EXIT |
| 1846 | 005114 | 013705 | 032672 | | | MOV | ABUFP,R5 | | |
| 1847 | 005120 | 012737 | 000001 | 020514 | | MOV | #1,DCOUNT | | |
| 1848 | 005126 | 004037 | 020452 | | | JSR | RO,DELAY | | :ALLOW 300 MS FOR XMIT TO RESTART |

```

1849 005132 023705 032672      CMP      ABUFP,R5
1850 005136 001317      BNE      ONOFLP      ;IT RESTARTED-GO STOP IT.
1851 005140 005304      DEC      R4
1852 005142 001360      BNE      ONLP        ;NOT YET 300 MS LOOP.
1853 005144 013737 002262 001120  MOV      VSTAT,$GDADR ;XMIT DIT NOT RESTART-ISSUE
1854 005152 104016      ERROR    16          ;ERROR AND EXIT
1855 005154 000240      ONOFLT: NOP
1856
1857      ;:*****
1858      ;ROUTINE TO TEST VT61 RAM AND THE COMMUNICATION PATHS.
1859      ;THIS ROUTINE ISSUES A SERIES OF PATTERNS(77/100,100/77,
1860      ;52/125,INCREMENTING,AND REV. VIDEO INCREMENTING) TO THE VT61.
1861      ;THE SCREEN IS THEN TRANSMITTED TO THE HOST AND AFTER EACH
1862      ;ITERATION RECEIVED DATA IS CHECKED AND ALL ERRORS(INCLUDING
1863      ;TRANSMISSION) ARE REPORTED.
1864      ;:*****
1865
1866      ;:*****
1867 005156 000004      TST7:  SCOPE
1868 005160 012737 000001 001160      MOV      #1,$TIMES    ;;DO 1 ITERATION
1869 005166 012737 005174 001106      MOV      #MEM1,$LPADR ;;SET SCOPE LOOP ADDRESS
1870
1871 005174 004037 016510      MEM1:  JSR      R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
1872 005200 005005      CLR      R5           ;CLEAR PATTERN OFFSET.
1873 005202 016504 005710      MEMA:  MOV      MPATT(R5),R4 ;LOAD PATTERN TO BE TRANSMITTED
1874 005206 004037 017532      JSR      R0,RESPTR    ;RESET POINTERS
1875 005212 042737 077577 002262      BIC      #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1876 005220 112777 000002 011256      MOVB     #SOM,@TBUFP  ;XMIT THE START OF MESSAGE.
1877 005226 004037 017406      JSR      R0,XMIT1
1878 005232 012702 003600      MOV      #TOTCH,R2   ;LOAD A COUNT OF SCREEN
1879 005236 005302      MEMB:  DEC      R2      ;DECREMENT XMIT COUNT
1880 005240 001414      BEQ      10$         ;COUNT = ZERO?
1881
1882 005242 004037 005656      12$:   JSR      R0,PATGN     ;NO-GENERATE NEXT BYTE TO XMIT.
1883 005246 110477 011232      MOVB     R4,@TBUFP   ;LOAD THE CHARACTER.
1884 005252 004037 017406      JSR      R0,XMIT1   ;NO-XMIT ANOTHER BYTE.
1885 005256 023737 002236 002240      CMP      FTLCNT,ALWCNT ;EXCEEDED FATAL ERROR COUNT?
1886 005264 103764      BLO      MEMB        ;NO-CHECK IF ANOTHER TRANSMISSION REQUIRED.
1887 005266 000137 005730      JMP      MEMXT       ;YES-GO ABORT TEST.
1888 005272 112777 000004 011204  10$:   MOVB     #EOM,@TBUFP ;XMIT END OF MESSAGE.
1889 005300 004037 017406      JSR      R0,XMIT1
  
```

| | | | | | | | | |
|------|--------|--------|--------|--------|------|-------|--------------|-------------------------------------|
| 1890 | 005304 | 004037 | 017532 | | | JSR | RO,RESPTR | :RESET INTERRUPT POINTERS. |
| 1891 | | | | | | | | |
| 1892 | 005310 | 013701 | 016500 | | | MOV | TBBUF,R1 | :LOAD XMIT BUFFER WITH |
| 1893 | 005314 | 013721 | 002170 | | | MOV | ESCN,(R1)+ | |
| 1894 | 005320 | 013721 | 002020 | | | MOV | CHOM,(R1)+ | :CURSOR HOME |
| 1895 | 005324 | 013721 | 002164 | | | MOV | ESCZI,(R1)+ | :ESCAPE Z |
| 1896 | 005330 | 013721 | 002112 | | | MOV | ESCO,(R1)+ | |
| 1897 | 005334 | 013721 | 002114 | | | MOV | XMTAL,(R1)+ | :TRANSMIT ALL |
| 1898 | 005340 | 013711 | 002012 | | | MOV | LNFEED,(R1) | :LINE FEED. |
| 1899 | 005344 | 012737 | 000010 | 016506 | | MOV | #8.,XMCNT | :SET UP TO XMIT 8 BYTES |
| 1900 | 005352 | 004037 | 017132 | | | JSR | RO,XMREC | :XMIT,WAIT FOR REC. EOM |
| 1901 | 005356 | 000402 | | | | BR | 1\$ | :NORMAL EXIT |
| 1902 | 005360 | 104011 | | | | ERROR | 11 | :LAST TRANSMIT CAUSED VT61 TO HANG |
| 1903 | 005362 | 000562 | | | | BR | MEMXT | :EXIT TEST |
| 1904 | 005364 | 042737 | 077577 | 002262 | 1\$: | BIC | #77577,VSTAT | :CLEAR ALL FLAGS BUT XOFF AND XMKIL |
| 1905 | 005372 | 005002 | | | | CLR | R2 | :CLEAR RECEIVE COUNTER. |

| | | | | | | | |
|------|--------|--------|--------|--------|----------|---------------|--|
| 1906 | 005374 | 016504 | 005710 | | MOV | MPATT(R5),R4 | :LOAD PATTERN |
| 1907 | 005400 | 012703 | 032472 | | MOV | #TCRLB+300,R3 | :SET UT ERROR STORAGE |
| 1908 | 005404 | 013701 | 016172 | | MOV | RBBUF,R1 | :SET UP RECEIVE POINTER |
| 1909 | 005410 | 005037 | 002256 | | CLR | DLAY | :SET UP TIME OUT DELAY |
| 1910 | 005414 | 013737 | 016172 | 016176 | MOV | RBBUF,RBUFP | :RESET RECEIVE POINTER |
| 1911 | 005422 | 023701 | 016176 | | 1\$: CMP | RBUFP,R1 | :RECEIVED A CHAR? |
| 1912 | 005426 | 001013 | | | BNE | MEMD | :YES-GO CHECK IT. |
| 1913 | 005430 | 032737 | 020000 | 002262 | BIT | #REOM,VSTAT | :HAVE WE RECEIVED EOM? |
| 1914 | 005436 | 001033 | | | BNE | CKDAT | :YES, GO CHECK FOR DATA ERRORS |
| 1915 | 005440 | 005337 | 002256 | | DEC | DLAY | :RUN TIME OUT DELAY. |
| 1916 | 005444 | 001366 | | | BNE | 1\$ | :NOT EXPIRED-KEEP LOOKING. |
| 1917 | 005446 | 005237 | 002236 | | INC | FTLCNT | :TRANSMISSION FAILED-INCR. FATAL COUNT |

| | | | | | | |
|------|--------|--------|--------|------------|----------------|-------------------------------------|
| 1918 | 005452 | 104011 | | ERROR | 11 | |
| 1919 | 005454 | 000525 | | BR | MEMXT | |
| 1920 | 005456 | 005202 | | MEMD: INC | R2 | ; DATA IN. INCREMENT COUNTER |
| 1921 | 005460 | 004037 | 005656 | JSR | R0, PATGN | ; GET GOOD CHARACTER, PUT IN R4 AND |
| 1922 | 005464 | 122705 | 000010 | CMPB | #10, R5 | ; CHECKING REV. VIDEO DATA? |
| 1923 | 005470 | 001002 | | BNE | 1\$ | ; NO-DO NOT MODIFY |
| 1924 | 005472 | 052704 | 000200 | BIS | #BIT07, R4 | ; YES-FORCE BIT 7. |
| 1925 | 005476 | 121104 | | 1\$: CMPB | (R1), R4 | ; COMPARE DATA |
| 1926 | 005500 | 001743 | | BEQ | MEMC | |
| 1927 | 005502 | 020227 | 003600 | CMP | R2, #TOTCH | ; COMPARING LAST CHAR? |
| 1928 | 005506 | 001740 | | BEQ | MEMC | ; YES-NEVER COUNT AS A ERROR. |
| 1929 | | | | | | |
| 1930 | 005510 | 020327 | 032542 | CMP | R3, #TCRLB+350 | ; STORED 20 ERRORS? |
| 1931 | 005514 | 103335 | | BHIS | MEMC | ; YES-STORE NO MORE. |
| 1932 | 005516 | 110423 | | MOVB | R4, (R3)+ | ; STORE THE GOOD DATA. |
| 1933 | 005520 | 111123 | | MOVB | (R1), (R3)+ | ; STORE THE BAD DATA. |
| 1934 | 005522 | 010223 | | MOV | R2, (R3)+ | ; STORE THE RECEIVE COUNT. |
| 1935 | 005524 | 000731 | | BR | MEMC | |
| 1936 | 005526 | 022703 | 032472 | CKDAT: CMP | #TCRLB+300, R3 | |
| 1937 | 005532 | 001415 | | BEQ | CKMEM | |
| 1938 | 005534 | 012701 | 032472 | MOV | #TCRLB+300, R1 | ; LOAD FIRST ERROR ADDRESS. |
| 1939 | 005540 | 004037 | 017110 | 1\$: JSR | R0, CLREG | ; CLEAR ERROR REGISTERS |
| 1940 | 005544 | 112137 | 001124 | MOVB | (R1)+, \$GDDAT | ; LOAD THE GOOD DATA. |
| 1941 | 005550 | 112137 | 001126 | MOVB | (R1)+, \$BDDAT | ; LOAD THE ERROR BUFFER |
| 1942 | 005554 | 012137 | 001120 | MOV | (R1)+, \$GDADR | ; LOAD RECEIVE COUNT |
| 1943 | 005560 | 104004 | | ERROR | 4 | ; ISSUE DATA ERROR MESSAGE. |
| 1944 | 005562 | 020103 | | CMP | R1, R3 | ; ISSUED ALL ERRORS? |
| 1945 | 005564 | 103765 | | BLO | 1\$ | ; NO-CONTINUE |
| 1946 | | | | | | |
| 1947 | 005566 | 020227 | 003600 | CKMEM: CMP | R2, #TOTCH | ; DID WE XFER 1920 TIMES? |
| 1948 | 005572 | 001406 | | BEQ | 1\$ | ; YES - GO CHECK STATUS |

```

1949 005574 012737 003600 001124      MOV    #TOTCH,$GDDAT  ;NO, PUT GOOD COUNT IN GDDAT
1950 005602 010237 001126      MOV    R2,$BDDAT     ;AND ACTUAL COUNT IN BDDAT.
1951 005606 104005      ERROR  5             ;ISSUE COUNT ERROR.
1952
1953 005610      1$:
1954 005610 012746 060000      MOV    #60000,-(SP)  ;;PUSH #60000 ON STACK
1955 005614 004037 016746      JSR    R0,CKSFT
1956 005620 062705 000002      ADD    #2,R5         ;INCREMENT PATTERN POINTER
1957 005624 005765 005710      TST    MPATT(R5)     ;TEST NEXT PATTERN
1958 005630 001437      BEQ    MEMXT         ;ZERO-END OF TEST EXIT.
1959 005632 100007      BPL    2$           ;NOT INCRMENTING PATTERN.
1960 005634 122705 000010      CMPB  #10,R5        ;SET REVERSE VIDEO?
1961 005640 001004      BNE    2$           ;NO.
1962 005642 012703 005724      MOV    #SETREV,R3   ;YES-ENTER REVERSE VIDEO
1963 005646 004037 017472      JSR    R0,LDXMIT    ;AND RE-ISSUE INCREMENTING PATTERN.
1964 005652 000137 005202      2$: JMP    MEMA        ;NOT ZERO, GO EXERCISE IT.
1965
1966 005656 042704 000200      PATGN: BIC   #200,R4  ;CLEAR REV. VIDEO BIT IF SET.
1967 005662 005704      TST    R4           ;CHECK R4 FOR PATTERN
1968 005664 100402      BMI    1$          ;IF MINUS, DO INCREMENTING.
1969 005666 000304      SWAB  R4           ;OTHERWISE SWAP BYTES AND
1970 005670 000200      RTS    R0          ;EXIT.
1971 005672 105204      1$: INCB  R4        ;ADD ONE TO INCREMENTING
1972 005674 120427 000177      CMPB  R4,#177     ;HAVE WE EXCEEDED LIMIT
1973 005700 103402      BLO    2$         ;NO, EXIT
1974 005702 016504 005710      MOV    MPATT(R5),R4 ;YES, RESET PATTERN AND
1975 005706 000200      2$: RTS    R0     ;EXIT.
1976
1977      .EVEN
1978 005710 037500      MPATT =.           ;PATTERN 77,100
1979 005712 040077      .WORD 037500      ;PATTERN 100,77
1980 005714 025125      .WORD 040077      ;PATTERN 52,125
1981 005716 100040      .WORD 025125      ;PATTERN INCREMENTING
1982 005720 100040      .WORD 100040      ;PATTERN INCREMENTING-REV. VIDEO.
1983 005722 000000      .WORD 0           ;PATTERN TABLE TERMINATOR
1984      .SEQUENCE TO ENTER REVERSE VIDEO.
1985 005724 033 117 112 SETREV: .BYTE .ESC,.O,.EEMP,0
1986 005727 000
1987 005730 000240      MEMXT: NOP
1988
1989      ;;*****
1990
1991      ;ROUTINE TO TEST THE ABILITY OF THE VT61 TO CALCULATE
1992      ;AND TRANSMIT CHECKSUMS OF BOTH TRANSMITTED AND RECEIVED
1993      ;DATA. SUBTEST A TRANSMITS A FULL BUFFER UPDATING A CALCULATED
1994      ;CHECKSUM ON EACH CHARACTER TRANSMITTED. AN ESCAPE SEQUENCE
1995      ;REQUESTING THE RECEIVER CHECKSUM IS EMBEDDED AT THE END OF
1996      ;XMIT BUFFER AND THE RECEIVED CHECKSUM IS COMPARED TO THE
1997      ;CALCULATED. SUBTEST B PERFORMS THE SAME TYPE OF CHECK ON
1998      ;THE VT61 TRANSMIT CHECKSUM,UTILIZING THE DATA SENT TO THE VT61
1999      ;IN SUBTEST A,DURING A FULL SCREEN TRANSMIT.
2000
2001      ;;*****
2002
2003      ;;*****
2004 005732 000004      TST10: SCOPE
  
```

```

2005 005734 012737 000003 001160      MOV      #3,$TIMES      ;;DO 3 ITERATIONS
2006 005742 012737 005750 001106      MOV      #CKSUMA,$LPADR ;;SET SCOPE LOOP ADDRESS
2007
2008 005750 004037 016510      CKSUMA: JSR      R0,RESETV      ;RESET THE UNIT AND WAIT FOR XON.
2009 005754 004037 017532      JSR      R0,RESPTR      ;RESET INTERRUPT POINTERS
2010 005760 012737 001001 002264      MOV      #1001,BLKM      ;SET XMIT TO SOM- DATA -EOM.
2011 005766 012703 006446      MOV      #ITSUMA,R3      ;DIS. RECT. MODE AND CLEAR CHECKSUM
2012 005772 004037 017472      JSR      R0,LDXMIT
2013 005776 042737 077577 002262      BIC      #77577,VSTAT      ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2014 006004 013701 016500      MOV      TBBUF,R1      ;LOAD XMIT BUFFER WITH
2015 006010 012703 006473      MOV      #315.,R3
2016 006014 004037 020516      JSR      R0,BLDINC      ;314 INCREMENTING CHAR.
2017 006020 113721 002170      MOV      ESCN,(R1)+
2018 006024 113721 002020      MOV      CHOM,(R1)+      ;CURSOR HOME
2019 006030 113721 002112      MOV      ESCO,(R1)+
2020 006034 113721 002113      MOV      ESCO+1,(R1)+
2021 006040 113711 002130      MOV      TXRCK,(R1)      ;TRANSMIT RECEIVER CHECKSUM.
2022 006044 005004      CLR      R4      ;CLEAR CHECKSUM REGISTER
2023 006046 012705 000004      MOV      #EOM,R5      ;PRELOAD CHECKSUM REG. WITH
2024 006052 004037 021174      JSR      R0,CALCK      ;EOM FROM PRIOR XMIT.
2025 006056 052737 002000 002262      BIS      #CKSUM,VSTAT      ;REQUEST CHECKSUM CALCULATIONS.
2026 006064 012737 000500 016506      MOV      #320.,XMCNT      ;SETUP TO XMIT 320 BYTES
2027 006072 052777 040000 173664      BIS      #XENA,@VJCSR      ;ENABLE XMIT INTERRUPTS
2028 006100 012746 020000      MOV      #REOM,-(SP)      ;;PUSH #REOM ON STACK
2029
2030 006104 005737 002270      *****
2031 006110 001407      TST      PTYPE      ;PROCESSOR TYPE?
2032 006112 100403      BEQ      2$      ;IF 0, THEN 11/45 OR 11/70
2033 006114 012746 000005      BMI      1$      ;IF -1, THEN 11/35 OR 11/40
2034 006120 000405      MOV      #5,-(SP)      ;;PUSH #5 ON STACK
2035 006122      BR      3$
2036 006122 012746 000012      1$:      MOV      #10.,-(SP)      ;;PUSH #10. ON STACK
2037 006126 000402      BR      3$
2038 006130      2$:
2039 006130 012746 000024      MOV      #20.,-(SP)      ;;PUSH #20. ON STACK
2040 006134 004037 022150      3$:      JSR      R0,WTBGND      ;LOOK FOR EOM.
2041 *****
2042 006140 000546      BR      CKEXT      ;ERROR EXIT IF NOT FOUND
2043 006142 127704 010024      CMPB    @RBBUF,R4      ;COMPARE CHECKSUMS
2044 006146 001414      BEQ      CKSUMB      ;GOOD GO TO SUBTEST B
2045 006150 004037 017110      JSR      R0,CLREG      ;BAD COMPARE
2046 006154 110437 001124      MOV      R4,$GDDAT      ;LOAD CALCULATED CHECKSUM
2047 006160 117737 010006 001126      MOV      @RBBUF,$BDDAT      ;AND VT61 RECEIVER CHECKSUM
2048 006166 104013      ERROR    13      ;ISSUE ERROR
2049 006170 012746 060001      MOV      #60001,-(SP)      ;;PUSH #60001 ON STACK
2050 006174 004037 016746      JSR      R0,CKSFT      ;ERROR.
2051
2052 006200 042737 077577 002262      CKSUMB: BIC      #77577,VSTAT      ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2053 006206 005004      CLR      R4      ;CLEAR CHECKSUM REGISTER
2054 006210 052737 000100 002262      BIS      #TXSUM,VSTAT      ;SET UP FOR XMIT CHECKSUM GENERATION.
2055 006216 012737 001001 002264      MOV      #1001,BLKM      ;SET XMIT TO SOM- DATA -EOM.
2056 006224 013701 016500      MOV      TBBUF,R1
2057 006230 004037 021242      JSR      R0,LDBUF      ;LOAD THE BUFFER WITH:
2058 006234      033      117      134      .BYFE .ESC,.O,.CLTCK,.ESC,.O,.XMTAL,.ESC,.O,.TXTCK,O
2059 006237      033      117      126
2060 006242      033      117      136
    
```

(REV. B0)

```

2061 006245 000
2062 006246 012737 000011 016506 MOV #9.,XMCNT ;SET UP TO XMIT 9 BYTES
2063 006254 052777 040000 173502 BIS #XENA,@VJCSR ;ALLOW XMIT INTERRUPTS
2064 006262 012746 000001 MOV #XMDNE,-(SP) ;;PUSH #XMDNE ON STACK
2065 ;***** (REV. B0)
2066 006266 005737 002270 TST PTYPE ;PROCESSOR TYPE?
2067 006272 001407 BEQ 2$ ;IF 0, THEN 11/45 OR 11/70
2068 006274 100403 BMI 1$ ;IF -1, THEN 11/35 OR 11/40
2069 006276 012746 000005 MOV #5,-(SP) ;;PUSH #5 ON STACK
2070 006302 000405 BR 3$ ;
2071 006304 1$:
2072 006304 012746 000012 MOV #10.,-(SP) ;;PUSH #10. ON STACK
2073 006310 000402 BR 3$ ;
2074 006312 2$:
2075 006312 012746 000024 MOV #20.,-(SP) ;;PUSH #20. ON STACK
2076 006316 004037 022150 JSR R0,WTBGND ;LOOK FOR XMIT DONE.
2077 ;*****
2078 006322 000455 BR CKEXT ;TIME OUT - EXIT TEST.
2079 006324 005037 002256 CKSRC: CLR DLAY ;SET UP TIME OUT DELAY
2080 006330 013702 032672 MOV ABUFP,R2 ;RESET THE RECEIVER FLAG
2081 006334 023702 032672 1$: CMP ABUFP,R2 ;RECEIVED A CHAR?
2082 006340 001007 BNE 2$ ;YES-GO CHECK IT.
2083 006342 005337 002256 DEC DLAY ;RUN TIME OUT DELAY.
2084 006346 001372 BNE 1$ ;
2085 006350 005237 002236 INC FTLCNT ;TIMED OUT-INCREMENT FATAL XMIT COUNT
2086 006354 104011 ERROR 11 ;ISSUE HUNG MESSAGE AND EXIT.
2087 006356 000437 BR CKEXT ;
2088 006360 122777 000004 024304 2$: CMPB #EOM,@ABUFP ;RECEIVED EOM CHAR?
2089 006366 001356 BNE CKSRC ;
2090 006370 042737 020000 002262 BIC #REOM,VSTAT ;CLEAR THE EOM FLAG
2091 006376 032737 020000 002262 BIT #REOM,VSTAT ;NOW WAIT FOR LAST EOM FLAG
2092 006404 001774 BEQ -6 ;FROM XMIT TRANSMITTER CHECKSUM.
2093 006406 120477 007560 CMPB R4,@RBBUF ;COMPARE 61 TO HOST CHECKSUM.
2094 006412 001421 BEQ CKEXT ;EQUAL - EXIT TEST
2095 006414 004037 017110 JSR R0,CLREG ;
2096 006420 110437 001124 MOV R4,$GDDAT ;LOAD THE HOST CALCULATED CHECKSUM
2097 006424 117737 007542 001126 MOV @RBBUF,$BDDAT ;LOAD THE VT61 TRANSMITTED CHECKSUM
2098 006432 104014 ERROR 14 ;ISSUE VT61 XMIT CHECKSUM ERROR
2099 006434 012746 060001 MOV #60001,-(SP) ;;PUSH #60001 ON STACK
2100 006440 004037 016746 JSR R0,CKSFT ;CHECK FOR STATUS ERROR
2101 006444 000404 BR CKEXT ;
2102
2103 006446 033 117 103 ITSUMA: .BYTE .ESC,.O,.DRECT,.ESC,.O,.CLRCK,0,0
2104 006451 033 117 133
2105 006454 000 000
2106
2107 006456 004037 017532 CKEXT: JSR R0,RESPTR
2108
2109 ;*****
2110 ;ROUTINE TO INSURE BASIC CURSOR COMMANDS
2111 ;RESULT IN CORRECT CURSOR MOVEMENT. COMMANDS
2112 ;ARE ISSUED IN THE SEQUENCE: RESET, CURSOR RIGHT,
2113 ;CURSOR DOWN, CURSOR LEFT, AND CURSOR UP. THE READ
2114 ;CURSOR POSITION COMMAND IS ISSUED AFTER EVERY
2115 ;CURSOR COMMAND AND CURRENT IS COMPARED TO GOOD
2116 ;AND ANY ERRORS REPORTED.

```

```

2117 ;:*****
2118
2119 ;:*****
2120 006462 000004          TST11: SCOPE
2121 006464 012737 000005 001160      MOV    #5,$TIMES      ;;DO 5 ITERATIONS
2122 006472 012737 006500 001106      MOV    #CURS1A,$LPADR ;;SET SCOPE LOOP ADDRESS
2123
2124 006500 013701 016500      CURS1A: MOV    TBBUF,R1      ;LOAD XMIT BUFFER ADDRESS
2125 006504 004037 016510      JSR    RO,RESETV      ;RESET THE UNIT AND WAIT FOR XON.
2126 006510 004037 021242      JSR    RO,LDBUF       ;LOAD THE BUFFER WITH:
2127 006514          033      103      033      .BYTE  .ESC,.CRT,.ESC,.O,.RDCUR,.ESC,.CDWN,.ESC
2128 006517          117      131      033
2129 006522          102      033
2130 006524          117      131      033      .BYTE  .O,.RDCUR,.ESC,.CLFT,.ESC,.O,.RDCUR
2131 006527          104      033      117
2132 006532          131
2133 006533          033      101      033      .BYTE  .ESC,.CUP,.ESC,.O,.RDCUR,.BEL,O
2134 006536          117      131      007
2135 006541          000
2136 006542 012737 000024 016506      MOV    #20.,XMCNT     ;SET TO XMIT 20 CHARACTERS
2137 006550 012737 000004 017400      MOV    #4,RECITT      ;SET RECEIVE ITERATION TO 4
2138 006556 012737 032272 017402      MOV    #TCRLB+100,WDSTOR;SET UP WORD STORAGE POINTER
2139 006564 004037 017132      JSR    RO,XMREC       ;XMIT ,AND WAIT FOR REC.DONE
2140 006570 000402
2141 006572 104011      BR     11$            ;NORMAL EXIT
2142 006574 000436      ERROR  11            ;LAST XMIT CAUSED /T61 TO HANG.
2143 006576 012701 006662      BR     CUR1XT        ;EXIT TEST
2144 006602 012702 032272      11$:  MOV    #GDCURP,R1     ;R1=GOOD POSITION TABLE
2145 006606 012703 002022      MOV    #TCRLB+100,R2  ;R2=ACTUAL CURSOR PGSITION
2146                                MOV    #CRT,R3        ;R3=CURSOR COMMAND TABLE
2147 006612 021112      12$:  CMP    (R1),(R2)     ;COMPARE GOOD TO ACTUAL
2148 006614 001415      BEQ   2$             ;OK-GO UPDATE POINTERS.
2149 006616 113737 002170 001125      MOVB  ESCN,$GDDAT+1
2150 006624 111337 001124      MOVB  (R3),$GDDAT    ;LOAD COMMAND IN ESC ERROR
2151 006630 005037 001126      CLR   $BDDAT
2152 006634 104001      ERROR  1            ;AND ISSUE IT
2153 006636 011237 031472      MOV   (R2),RCRLB     ;LOAD BAD CURSOR POSITION
2154 006642 011146      MOV   (R1),-(SP)     ;;PUSH (R1) ON STACK
2155 006644 004037 017612      JSR   RO,CURER       ;LOAD AND ISSUE CURSRO ERROR MESSAGE
2156 006650 022122      2$:  CMP   (R1)+,(R2)+   ;INCREMENT POSITION POINTERS.
2157 006652 022337 002030      CMP   (R3)+,CUP      ;CHECK FOR COMMAND TERM.(CUP).
2158 006656 001355      BNE  12$            ;NOT AT TERMINATOR-COMPARE AGAIN
2159 006660 000404      BR   CUR1XT         ;EXIT TEST
2160
2161
2162 006662 020440      GDCURP: .WORD 20440    ;ROW 0, COL. 1
2163 006664 020441      .WORD 20441        ;ROW 1, COL. 1
2164 006666 020041      .WORD 20041        ;ROW 1, COL. 0
2165 006670 020040      .WORD 20040        ;ROW 0, COL. 0
2166 006672 000240      CUR1XT: NOP
2167
2168 ;:*****
2169 ;ROUTINE TO INSURE THAT READ CHARACTER AT CURSOR
2170 ;FUNCTIONS CORRECTLY. COMMAND SEQUENCE IS: RESET, A, CURSOR
2171 ;LEFT, READ CHARACTER AT CURSOR.
2172 ;AN ERROR IS REPORTED IF THE LAST READ IS NOT AN 'A'.

```

```

2173 ;:*****
2174 ;:*****
2175 ;:*****
2176 006674 000004 TST12: SCOPE
2177 006676 012737 000005 001160 MOV #5,$TIMES ;;DO 5 ITERATIONS
2178 006704 012737 006712 001106 MOV #CURS1B,$LPADR ;;SET SCOPE LOOP ADDRESS
2179
2180 006712 013701 016500 CURS1B: MOV TBBUF,R1
2181 006716 004037 016510 JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
2182 006722 012721 000101 MOV #101,(R1)+ ;A
2183 006726 113721 002170 MOVB ESCN,(R1)+
2184 006732 113721 002026 MOVB CLFT,(R1)+ ;CURSOR LEFT
2185 006736 013721 002112 MOV ESCOI,(R1)+
2186 006742 013711 002122 MOV TCUCH,(R1) ;TRANSMIT CH. AT CURSOR
2187 006746 012737 000006 016506 MOV #6,XMCNT ;SET UP TO XMIT 6 CHARACTERS
2188 006754 004037 017132 JSR R0,XMREC ;XMIT STRING AND WAIT FOR EOM.
2189 006760 000402 BR 10$ ;NORMAL EXIT
2190 006762 104011 ERROR 11 ;LAST XMIT CAUSED VT61 TO HANG/FAIL
2191 006764 000430 BR 2$ ;EXIT TEST
2192 006766 127727 007200 000101 10$: CMPB @RBBUF,#101 ;CHARACTER READ=A
2193 006774 001424 BEQ 2$ ;YES-NEXT SUBTEST
2194 006776 013737 002112 001124 MOV ESCOI,$GDDAT
2195 007004 000337 001124 SWAB $GDDAT ;REASSEMBLE ESC DATA
2196 007010 005037 001126 CLR $BDDAT
2197 007014 113737 002122 001127 MOVB TCUCH,$BDDAT+1 ;LOAD FAILING ESC SEQUENCE
2198 007022 104001 ERROR 1 ;AND ISSUE IT
2199 007024 004037 017110 JSR R0,CLREG
2200 007030 112737 000101 001124 MOVB #101,$GDDAT ;LOAD GOOD CH. AND CH.
2201 007036 117737 007130 001126 MOVB @RBBUF,$BDDAT
2202 007044 104004 ERROR 4 ;READ AND ISSUE THEM.
2203
2204 007046 000240 2$: NOP ;END OF TEST
2205 ;:*****
2206 ;:ROUTINE TO VERIFY OPERATION OF REPLACE AND INSERT MODE.
2207 ;:INITIALLY ROW 0 IS WRITTEN TO 80 INCREMENTING CHAR.
2208 ;:ON THE FIRST PASS(REPLACE MODE) A CHARACTER IS REPLACED
2209 ;:AT HOME AND THE CHAR. AT ROW0,COL.0(172) AND ROW1,COLO(NULL)
2210 ;:ARE VERIFIED. ON THE SECOND PASS, INSERT MODE IS ENTERED
2211 ;:AND THE RESULTING INSERTION(AT HOME) IS VERIFIED.ROW0,COLO
2212 ;:SHOULD BE 172 AND ROW1,COLO SHOULD BE 161.
2213 ;:*****
2214 ;:*****
2215 ;:*****
2216 007050 000004 TST13: SCOPE
2217 007052 012737 000005 001160 MOV #5,$TIMES ;;DO 5 ITERATIONS
2218 007060 012737 007066 001106 MOV #INRPL,$LPADR ;;SET SCOPE LOOP ADDRESS
2219
2220 007066 004037 016510 INRPL: JSR R0,RESETV ;RESET THE UNIT
2221 007072 013701 016500 MOV TBBUF,R1
2222 007076 005201 INC R1 ;LEAVE ROOM IN BUFFER FOR SOM.
2223 007100 012703 000120 MOV #80.,R3 ;CREATE A LINE OF 80 INCREMENTING
2224 007104 004037 020516 JSR R0,BLDINC ;CHAR. ON THE SCREEN.
2225 007110 105011 CLRB (R1)
2226 007112 013703 016500 MOV TBBUF,R3
2227 007116 004037 017472 JSR R0,LDXMIT
2228 007122 005005 CLR R5 ;USE R5 AS TEST INDEXER.
  
```

```

2229 007124 012737 000002 017400 INAG· MOV #2,RECITT ;SET UP TO RECEIVE 2 CHAR.
2230 007132 012737 032372 017404 MOV #TCRLB+200,BYSTOR ;SET UP STORAGE AREA.
2231 007140 013701 016500 MOV TBBUF,R1
2232 007144 004037 021242 JSR RO,LDBUF ;LOAD THE BUFFER WITH:
2233 007150 033 110 172 .BYTE .ESC,.CHOM,172,.ESC,.CHOM,.ESC,.O,.TCUCH
2234 007153 033 110 033
2235 007156 117 127
2236 007160 033 102 033 .BYTE .ESC,.CDWN,.ESC,.O,.TCUCH,0
2237 007163 117 127 000
2238 007166 012737 000015 016506 MOV #13,XMCNT ;SET UP TO XMIT 13 CAHR.
2239 007174 004037 017132 JSR RO,XMREC
2240 007200 000402 BR 1$ ;NORMAL EXIT
2241 007202 104011 ERROR 11 ;LAST XMIT CAUSED UNIT TO HANG.
2242 007204 000433 BR INRXT ;EXIT TEST.
2243 007206 026537 007264 032372 1$: CMP TDATA(R5),TCRLB+200 ;COMPARE GOOD TO REC.DATA.
2244 007214 001407 BEQ 2$ ;GOOD-LOOP OR EXIT.
2245 007216 016537 007256 001126 MOV TFUNCTION(R5),$BDDAT
2246 007224 013737 002156 001124 MOV ESCP,$GDDAT ;LOAD ESCAPE SEQ. ERROR.
2247 007232 104001 ERROR 1
2248 007234 005725 2$: TST (R5)+ ;INCREMENT INDEXER.
2249 007236 020527 000004 CMP R5,#4 ;THRU WITH TEST?
2250 007242 001414 BEQ INRXT ;YES-EXIT.
2251 007244 012703 007270 MOV #ENSRT,R3 ;NO-SECOND PASS- ENTER
2252 007250 004037 017472 JSR RO,LDXMIT ;INSERT MODE AND DO AGAIN.
2253 007254 000723 BR INAG
2254
2255 007256 000151 000111 177777 TFUNCTION: .WORD .ERPL,.EINST,-1
2256 007264 172 000 172 TDATA: .BYTE 172,0,172,160
2257 007267 160
2258 007270 033 120 111 ENSRT: .BYTE .ESC,.P,.EINST,0
2259 007273 000
2260 007274 000240 INRXT: NOP
2261
2262 ;:*****
2263 ;ROUTINE TO INSURE VT61 WILL SCROLL IF A LINE FEED
2264 ;IS ISSUED FROM ROW 23 OR A DATA ENTRY FROM ROW23,COL. 79.
2265 ;IN SUBTEST A, ROW 0 IS INTIALLY WRITTEN TO A 0 AND ROW 1
2266 ;A 1. AFTER COMPLETION OF A LINE FEED(AND RESULTING SCROLL)
2267 ;ROW 00,COL.00 IS EXPECTED TO CONTAIN A 1.
2268 ;IN SUBTEST B, THE CURSOR IS PLACED AT ROW23,COL.79
2269 ;AND A DATA CHARACTER "A" IS ENTERED. THE CURSOR
2270 ;POSITION IS THEN READ AND SHOULD BE ROW23,COL.00. THE
2271 ;CHARACTER AT HOME IS VERIFIED TO BE A NULL.
2272 ;:*****
2273
2274 ;:*****
2275 007276 000004 TST14: SCOPE
2276 007300 012737 000005 001160 MOV #5,$TIMES ;:DO 5 ITERATIONS
2277 007306 012737 007314 001106 MOV #CKSCRA,$LPADR ;:SET SCOPE LOOP ADDRESS
2278
2279 007314 004037 016510 CKSCRA: JSR RO,RESETV ;RESET THE UNIT.
2280 007320 013701 016500 MOV TBBUF,R1
2281 007324 004037 021242 JSR RO,LDBUF ;LOAD THE XMIT BUFFER WITH:
2282 007330 060 033 102 .BYTE 60,.ESC,.CDWN,.ESC,.CLFT,61,.ESC,.Y,.R23,.COU
2283 007333 033 104 061
2284 007336 033 131 067
  
```



```

2285 007341 040
2286 007342 012 033 110 .BYTE .LNFED,.ESC,.CHOM,.ESC,.O,.TCUCH,.BEL,O
2287 007345 033 117 127
2288 007350 007 000
2289 007352 012737 000020 016506 MOV #16.,XMCNT ;SET UP TO XMIT 16 BYTES.
2290 007360 004037 017132 JSR RO,XMREC ;NORMAL EXIT
2291 007364 000402 BR 1$ ;LAST XMIT CAUSED UNIT TO HANG.
2292 007366 104011 ERROR 11 ;EXIT TEST.
2293 007370 000452 BR GDSCRL ;CHARACTER AT HOME A 1?
2294 007372 127727 006574 000061 1$: CMPB @RBBUF,#61 ;YES-NEXT TEST
2295 007400 001401 BEQ CKSCRB ;NO-ISSUE NO SCROLL ERROR.
2296 007402 104023 ERROR 23 ;SET UP FOR TWO REC. LOOPS.
2297 007404 012737 000002 017400 CKSCRB: MOV #2,RECITT ;SET UP CURSOR POSITION STROAGE.
2298 007412 012737 032372 017402 MOV #TCRLB+200,WDSTOR
2299 007420 013701 016500 MOV TBBUF,R1 ;LOAD XMIT BUFFER WITH:
2300 007424 004037 021242 JSR RO,LDBUF
2301 007430 033 131 067 .BYTE .ESC,.Y,.R23,.C79,101,.ESC,.O,.RDCUR
2302 007433 157 101 033
2303 007436 117 131
2304 007440 033 110 033 .BYTE .ESC,.CHOM,.ESC,.O,.TCUCH,O
2305 007443 117 127 000
2306 007446 012737 000015 016506 MOV #13.,XMCNT ;SET UP TO XMIT 13 BYTES.
2307 007454 004037 017132 JSR RO,XMREC ;XMIT AND WAIT FOR RECEIVED DONE.
2308 007460 000402 BR 1$ ;LAST XMIT CAUSED /T61 TO HANG.
2309 007462 104011 ERROR 11 ;ERROR EXIT
2310 007464 000414 BR GDSCRL ;NULL RECEIVED?
2311 007466 127737 006500 002226 1$: CMPB @RBBUF,ZERO ;YES-EXIT TEST
2312 007474 001410 BEQ GDSCRL ;NO-ISSUE NO SCROLL ERROR.
2313 007476 104023 ERROR 23 ;LOAD RECEIVED CURSOR POSITION.
2314 007500 013777 032372 006464 MOV TCRLB+200,@RBBUF ;:PUSH R23C00 ON STACK
2315 007506 013746 002202 MOV R23C00,-(SP) ;GO ISSUE CURSOR ERROR.
2316 007512 004037 017612 JSR RO,CURER
2317 007516 000240 GDSCRL: NOP
2318
2319
2320 ;:*****
2321 ;THIS TEST INSURES THAT THE VT61 CURSOR CAN BE
2322 ;POSITIONED TO VERY POSSIBLE ROW/COLUMN POSITON
2323 ;ON THE SCREEN. THIS IS TESTED BY FILLING THE
2324 ;COMPLETE SCREEN WITH A CHARACTER(A) AND THEN
2325 ;POSITONING THE CURSOR (VIA DCA) TO EVERY POSITION
2326 ;AND THE "A" AT THAT POSITION IS REPLACED WITH A SPACE.
2327 ;THE SCREEN IS THEN READ TO VERIFY THAT ONLY SPACES
2328 ;EXIST ON THE SCREEN. ALL POSITIONS CONTAINING
2329 ;NON-SPACES ARE REPORTED.
2330 ;:*****
2331 ;:*****
2332 ;:*****
2333 ;:*****
2334 007520 000004 TST15: SCOPE
2335 007522 012737 000001 001160 MOV #1,$TIMES ;:DO 1 ITERATION
2336 007530 012737 007536 001106 MOV #CURS2,$LPADR ;:SET SCOPE LOOP ADDRESS
2337
2338 007536 042737 077577 002262 CURS2: BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2339 007544 004037 016510 JSR RO,RESETV ;RESET THE UNIT AND WAIT FOR XON.
2340 007550 012702 003600 MOV #TOTCH,R2 ;LOAD A COUNT OF SCREEN(1920).
  
```

```

2341 007554 112777 000002 006722      MOVB  #SOM,@TBUF  ;XMIT THE START OF MESSAGE.
2342 007562 004037 017406      JSR   R0,XMIT1
2343 007566 005302      1$:  DEC   R2        ;DECREMENT XMIT COUNT
2344 007570 001413      BEQ   10$        ;COUNT = ZERO?
2345
2346 007572 112777 000101 006704      MOVB  #101,@TBUF  ;LOAD THE CHARACTER(A).
2347 007600 004037 017406      JSR   R0,XMIT1   ;NO-XMIT ANOTHER BYTE.
2348 007604 023737 002236 002240      CMP   FTLCNT,ALWCNT ;EXCEEDED FATAL ERROR COUNT?
2349 007612 103765      BLO   1$        ;NO-CHECK IF XMIT COMPLETE.
2350 007614 000137 010254      JMP   C2XT      ;YES-GO ABORT TEST.
2351 007620 004037 017532      10$: JSR   R0,RESPTR ;RESET INTERRUPT POINTERS.
2352 007624 013737 002216 020014      MOV   R2,C78,LNRW ;SET UP 1ST ADDRESS
2353 007632 013701 016500      MOV   TBUF,R1   ;LOAD XMIT BUFFER WITH
2354 007636 013721 002076      MOV   DCRAD,(R1)+
2355 007642 010102      MOV   R1,R2     ;R2 POINTS TO CURSOR ADD. IN BUFFER
2356 007644 013721 002216      MOV   R2,C78,(R1)+ ;CURSOR TO LOWER RIGHT -1.
2357 007650 112721 000040      MOVB  #40,(R1)+  ;SPACE
2358 007654 012737 000005 016506      MOV   #5,XMCNT  ;SET UP TO XMIT 5 CHARACTERS
2359 007662 042737 077577 002262      BIC   #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2360 007670 052777 040000 172066      BIS   #XENA,@VJCSR ;XMIT INTERRUPTS.
2361 007676 012746 000001      MOV   #XMDNE,-(SP) ;:PUSH #XMDNE ON STACK
2362      ;***** (REV B0)
2363 007702 005737 002270      TST   PTYPE     ;PROCESSOR TYPE?
2364 007706 001407      BEQ   51$      ;IF 0 THEN 11/45 OR 11/70
2365 007710 100403      BMI   50$      ;IF -1 THEN 11/35 OR 11/40
2366 007712 012746 000001      MOV   #1,-(SP) ;:PUSH #1 ON STACK
2367 007716 000405      BR    53$
2368 007720      50$:
2369 007720 012746 000003      MOV   #3,-(SP) ;:PUSH #3 ON STACK
2370 007724 000402      BR    53$
2371 007726      51$:
2372 007726 012746 000006      MOV   #6,-(SP) ;:PUSH #6 ON STACK
2373 007732 004037 022150      53$: JSR   R0,WTBGND ;LOOK FOR XMIT DONE
2374      ;*****
2375 007736 000546      BR    C2XT     ;NOT FOUND-ERROR EXIT
2376 007740 021237 002212      CMP   (R2),CUHME ;DELETED TO HOME?
2377 007744 001405      BEQ   3$      ;YES
2378 007746 004037 017710      JSR   R0,CMPOS  ;NO-GET NEXT POSITION TO BE DELETED
2379 007752 013712 020014      MOV   LNRW,(R2) ;LOAD IT IN XMIT BUFFER
2380 007756 000736      BR    2$      ;AND DELETE IT.
2381 007760 004037 017532      3$: JSR   R0,RESPTR ;RESET INTERRUPT POINTERS
2382 007764 013737 002212 020014      MOV   CUHME,LNRW ;LOAD INITIAL CHECK POSITION(HOME)
2383 007772 013701 016500      MOV   TBUF,R1   ;LOAD XMIT BUFFER WITH
2384 007776 010102      MOV   R1,R2     ;STORE ERRORS IN XMIT BUFFER
2385 010000 042737 077577 002262      BIC   #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2386 010006 012737 001001 002264      MOV   #1001,BLKM ;SET XMIT TO SOM- DATA -EOM.
2387 010014 013721 002170      MOV   ESCN,(R1)+
2388 010020 013721 002020      MOV   CHGM,(R1)+ ;CURSOR HOME
2389 010024 013721 002112      MOV   ESCO,(R1)+
2390 010030 013721 002114      MOV   XMTAL,(R1)+ ;TRANSMIT ALL
2391 010034 012737 000005 016506      MOV   #5,XMCNT
2392 010042 052777 040000 171714      BIS   #XENA,@VJCSR ;SET XMIT ENABLE
2393 010050 012746 000001      MOV   #XMDNE,-(SP) ;:PUSH #XMDNE ON STACK
2394      ;***** (REV. B0)
2395 010054 005737 002270      TST   PTYPE     ;PROCESSOR TYPE
2396 010060 001407      BEQ   61$      ;IF 0 THEN 11/45 OR 11/70
  
```

```

2397 010062 100403          BMI 60$          ;IF -1 THEN 11/35 OR 11/40
2398 010064 012746 000001    MOV #1,-(SP)      ;;PUSH #1 ON STACK
2399 010070 000405          BR 63$           ;
2400 010072          60$:          MOV #3,-(SP)      ;;PUSH #3 ON STACK
2401 010072 012746 000003    BR 63$           ;
2402 010076 000402          61$:          MOV #6,-(SP)      ;;PUSH #6 ON STACK
2403 010100          63$:          JSR R0,WTBGND     ;LOOK FOR SOM OR XMIT DONE
2404 010100 012746 000006    ;*****
2405 010104 004037 022150    BR C2XT         ;NOT FOUND-ERROR EXIT
2406 010110 000461          4$:          MOV RBUF,R1      ;SET UP RECEIVE FLAG
2407 010112 013701 016176    CLR DLAY        ;SET UP TIME OUT DELAY
2408 010116 005037 002256    40$:         CMP R1,RBUF     ;CHARACTER RECEIVED?
2409 010122 020137 016176    BLO 41$        ;YES-GO CHECK IT.
2410 010126 103411          BIT #REOM,VSTAT ;LOOK FOR END OF MESSAGE
2411 010130 032737 020000 002262 BNE C2CK       ;FOUND IT, EXIT TEST
2412 010136 001025          DEC DLAY        ;RUN TIME OUT DELAY.
2413 010140 005337 002256    BNE 40$        ;AND LOOK FOR RECEIVED CH.
2414 010144 001366          ERROR 11       ;LAST XMIT CAUSED VT61 TO HANG.
2415 010146 104011          BR C2CK        ;GO SEE IF ANY ERRORS STORED.
2416 010150 000420          41$:         MOV RBUF,RBUF    ;RESET RECEIVE POINTER
2417 010152 013737 016172 016176 CMPB @RBUF,#40  ;CHAR EQUAL A SPACE?
2418 010156 127727 006006 000040 BNE 6$         ;NOT A SPACE-MUST BE ERROR-STORE IT
2419 010160 001003          5$:          JSR R0,CPPOS    ;UPDATE CURSOR POSITION
2420 010170 004037 017752    BR 4$         ;
2421 010174 000746          6$:          CMP #TCRLB+20.,R2 ;STORED 10 ERRORS?
2422 010176 022702 032216    BLOS 5$        ;YES-IGNORE ANY FURTHER ERRORS.
2423 010202 101772          MOV LNRW,(R2)+ ;STORE FAILING CURSOR POSITION
2424 010204 013722 020014    BR 5$         ;
2425 010210 000767          C2CK:       CMP R2,TBBUF    ;ANY ERRORS STORED?
2426 010212 020237 016500    BEQ C2XT      ;NO EXIT TEST
2427 010216 001416          MOV TBBUF,R1  ;USE R1 AS ERROR POINTER
2428 010220 013701 016500    1$:          CMP (R1),R23C79 ;CURSOR TO LOWER RIGHT?
2429 010224 021137 002100    BEQ C2XT     ;YES-NOT AN ERROR.
2430 010230 001411          ERROR 12     ;NO-ISSUE ERROR MESSAGES
2431 010232 104012          MOV #20040,-(SP) ;PUSH #20040 ON STACK
2432 010234 012746 020040    MOV (R1)+,@RBUF ;LOAD FAILING POS.
2433 010236 012177 005726    JSR R0,CURER  ;ISSUE CURSOR ERROR
2434 010240 004037 017612    CMP R1,R2     ;DONE WITH ERRORS?
2435 010244 004037 017612    BLO 1$       ;NO, DUMP ANOTHER.
2436 010250 020102          C2XT:       NOP      ;EXIT TEST
2437 010252 103764          ;*****
2438 010254 000240          ;ROUTINE TO INSURE PROPER OPERATION OF CARRIAGE RETURN
2439          ;AND LINE FEED DURING NORMAL MODE. INITIALLY THE CURSOR IS
2440          ;SET(VIA D.C.A.) TO ROW0,COL 20 AND A LINE FEEL IS ISSUED
2441          ;THE CURSOR POSITION IS THEN READ AND MUST BE ROW1,COL20.
2442          ;A CARRIAGE RETURN IS THEN ISSUED AND CURSOR POSITION VERIFIED
2443          ;TO BE ROW1,COLO.
2444          ;*****
2445          ;*****
2446          ;*****
2447          ;*****
2448          ;*****
2449          ;*****
2450          ;*****
2451          ;*****
2452 010256 000004    TST16: SCOPE
  
```

```

2453 010260 012737 000005 001160      MOV      #5,$TIMES      ;;DO 5 ITERATIONS
2454 010266 012737 010274 001106      MOV      #NWLN,$LPADR  ;;SET SCOPE LOOP ADDRESS
2455
2456 010274 004037 016510      NWLN:   JSR      RO,RESETV ;RESET THE UNIT AND ENTER MAINT.MODE
2457 010300 013701 016500      MOV      TBBUF,R1
2458 010304 004037 021242      JSR      RO,LDBUF      ;LOAD XMIT BUFFER WITH-
2459 010310      033      131      040      .BYTE    .ESC,.Y,.R00,.C20
2460 010313      064
2461 010314      012      033      117      .BYTE    .LNFED,.ESC,.O,.RDCUR,.BEL,O
2462 010317      131      007      000
2463 010322 012737 000011 016506      MOV      #9.,XMCNT     ;SETUP TO XMIT 9 CHARACTERS
2464 010330 004037 017132      JSR      RO,XMREC      ;GO DO IT
2465 010334 000402      BR       30$          ;NORMAL EXIT.
2466 010336 104011      ERROR    11          ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
2467 010340 000454      BR       4'          ;EXIT TEST
2468 010342 023777 002174 005622 30$:   CMP      R01C20,@RBBUF ;CHECK CURSOR POS. S/B ROW 1, COL 20.
2469 010350 001412      BEQ     3$
2470 010352 005037 001124      CLR     $GDDAT
2471 010356 013737 002012 001126      MOV     LNFED,$BDDAT
2472 010364 104001      ERROR    1          ;ISSUE IT
2473 010366 013746 002174      MOV     R01C20,-(SP)  ;;PUSH R01C20 ON STACK
2474 010372 004037 017612      JSR     RO,CURER      ;SETUP AND ISSUE CURSOR ERROR
2475 010376 013701 016500      3$:   MOV     TBBUF,R1
2476 010402 013721 002010      MOV     CARRT,(R1)+   ;LOAD XMIT BUFFER WITH
2477 010406 013721 002112      MOV     ESCOI,(R1)+   ;CARRIAGE RETURN, READ CURSOR
2478 010412 013721 002106      MOV     RDCUR,(R1)+  ;POSITION
2479 010416 012737 000004 016506      MOV     #4,XMCNT     ;SET UP TO TRANSMIT 4 CHARACTERS
2480 010424 004037 017132      JSR     RO,XMREC      ;GO DO IT
2481 010430 000402      BR     40$          ;NORMAL EXIT.
2482 010432 104011      ERROR    11          ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
2483 010434 000416      BR     4$          ;EXIT TEST
2484 010436 023777 002172 005526 40$:   CMP     R01C00,@RBBUF ;CHECK CURSOR POS. S/B ROW1, COL 0.
2485 010444 001412      BEQ     4$          ;EXIT TEST IF GOOD.
2486 010446 005037 001124      CLR     $GDDAT
2487 010452 013737 002010 001126      MOV     CARRT,$BDDAT
2488 010460 104001      ERROR    1          ;ISSUE IT
2489 010462 013746 002172      MOV     R01C00,-(SP) ;;PUSH R01C00 ON STACK
2490 010466 004037 017612      JSR     RO,CURER      ;SET UP AND ISSUE CURSOR ERROR
2491 010472 000240      4$:   NOP
2492
2493
2494      ;:*****
2495
2496      ;ROUTINE TO VERIFY PROPER OPERATION OF ERASE TO END-OF-
2497      ;SCREEN. SCREEN IS WRITTEN TO 1920 INCREMENTING CHAR.
2498      ;ERASE TO END OF SCREEN IS THEN ISSUED AND THE
2499      ;ENTIRE SCREEN IS READ VERIFYING THAT IT IS ALL NULLS.
2500      ;:*****
2501
2502      ;:*****
2503      TST17: SCOPE
2504 010474 000004      MOV     #3,$TIMES     ;;DO 3 ITERATIONS
2505 010476 012737 000003 001160      MOV     #ERSE,$LPADR  ;;SET SCOPE LOOP ADDRESS
2506 010504 012737 010512 001106
2507
2508 010512 004037 016510      ERSE:   JSR     RO,RESETV ;RESET THE UNIT -SET MAINT. MODE.
    
```

```

2509 010516 005077 005450 CLR @RBBUF ;CLEAR THE CHECK LOCATION.
2510 010522 004037 020544 JSR RO,DATSC ;FILL THE SCREEN.
2511 010526 013701 016500 MOV TBBUF,R1
2512 010532 004037 021242 JSR RO,LDBUF ;LOAD XMIT BUFFER WITH:
2513 010536 033 110 033 .BYTE .ESC,.CHOM,.ESC,.EOS,.ESC,.O,.XMTAL,O
2514 010541 112 033 117
2515 010544 126 000
2516 010546 113737 002170 001125 MOVB ESCN,$GDDAT+1
2517 010554 113737 002032 C01124 MOVB EOS,$GDDAT ;LOAD ERROR WITH ERASE TO EOS
2518 010562 005037 001126 CLR $BDDAT
2519 010566 005077 005400 CLR @RBBUF
2520 010572 012737 000007 016506 MOV #7,XMCNT ;SET UP TO XMIT 7 BYTES
2521 010600 004037 017132 JSR RO,XMREC ;XMIT AND WAIT FOR REC. DONE
2522 010604 000402 BR 5$
2523 010606 104011 ERROR 11 ;ESC ERROR
2524 010610 000413 BR ERSXT ;EXIT TEST
2525 010612 127737 005354 002226 5$: CMPB @RBBUF,ZERO ;VT61 XMITTED SOM/EOM ONLY?
2526 010620 001407 BEQ ERSXT ;YES-EXIT TEST.
2527 010622 104001 ERROR 1 ;NO-ERASE TO END OF SCREEN
2528 010624 004037 017110 JSR RO,CLREG ;GO CLEAR ERROR STORAGE
2529 010630 117737 005336 001126 MOVB @RBBUF,$BDDAT
2530 010636 104004 ERROR 4 ;ISSUE DATA ERROR
2531 010640 000240 ERSXT: NOP
    
```

```

2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
    ;:*****
    ;:ROUTINE TO SET UP END OF PASS INDICATION.
    ;:SELF TEST(ESC O T) IS ISSUED TO THE UNIT UNDER TEST
    ;:AND AN ERROR IS ISSUED IF THE UNIT CANNOT RESPOND AFTER
    ;:SELF TEST IS COMPLETE. IF SELF TEST IS SUCCESSFUL THE
    ;:SCREEN IS WRITTEN TO 23 LINES OF INCREMENTING CHARACTERS
    ;:AND 23 LINES OF INCREMENTING CHAR. IN REVERSE VIDEO.
    ;:THE IDENT IS THEN CHECKED AND IF A COPIER IS PRESENT A
    ;:COPY SCREEN COMMAND IS ISSUED(NOTE: THIS COMMAND WILL CAUSE
    ;:THE UNIT TO BE 'BUSY' AND NOT RESPOND TO ANY FURTHER COMMANDS
    ;:UNTIL THE SCREEN HAS BEEN COMPLETELY COPIED.)
    ;:*****
    
```

```

2549 010642 000004 TST20: SCOPE
2550 010644 012737 000001 001160 MOV #1,$TIMES ;:DO 1 ITERATION
2551 010652 012737 010660 001106 MOV #LSTST,$LPADR ;:SET SCOPE LOOP ADDRESS
2552
2553 010660 LSTST:
2554 010660 013746 002226 MOV ZERO,-(SP) ;:PUSH ZERO ON STACK
2555 010664 013746 002160 MOV TSTER,-(SP) ;:PUSH TSTER ON STACK
2556 010670 013746 002112 MOV ESCO,-(SP) ;:PUSH ESCO ON STACK
2557 010674 004037 014606 JSR RO,TESC ;:TRANSMIT IT.
2558 010700 004037 016626 JSR RO,GETON ;:GO LOOK FOR A XON.
2559 010704 000407 BR 1$ ;:VT61 RESPONDED-NOT HUNG
2560 010706 013737 001764 001124 MOV VJCSR,$GDDAT ;:LOAD THE ADDRESS
2561 010714 013737 001774 001126 MOV VECT,$BDDAT ;:LOAD THE VECTOR
2562 010722 104010 ERROR 10 ;:REPORT SELF TEST FAILURE
2563 010724 004037 016510 1$: JSR RO,RESETV ;:RESET AND SET MAINT. MODE.
2564 010730 005037 002246 CLR BUBCT ;:SET UP HALF-SCREEN FLAG.
    
```

```

2565 010734 042737 077577 002262 2$: BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2566 010742 013701 016500 MOV TBBUF,R1 ;SET UP BEG. OF XMIT BUFFER
2567 010746 012703 000500 MOV #320.,R3 ;FILL BUFFER WITH INCREMENTING CHAR.
2568 010752 004037 020516 JSR RO,BLDINC
2569 010756 012737 001001 002264 MOV #1001,BLKM ;SET XMIT TO SOM- DATA -EOM.
2570 010764 012737 001700 016506 MOV #960.,XMCNT ;SEND 12 LINE TO VT61
2571 010772 052777 040000 170764 BIS #XENA,@VJCSR ;ENABLE XMIT INTERRUPTS
2572 011000 012746 000001 MOV #XMDNE,-(SP) ;;PUSH #XMDNE ON STACK
2573 ***** (REV. B0)
2574 011004 005737 002270 TST PTYPE ;PROCESSOR TYPE
2575 011010 001407 BEQ 5$ ;IF 0 THEN 11/45 OR 11/70
2576 011012 100403 BMI 4$ ;IF -1 THEN 11/35 OR 11/40
2577 011014 012746 000031 MOV #25.,-(SP) ;;PUSH #25. ON STACK
2578 011020 000405 BR 6$
2579 4$:
2580 011022 012746 000062 MOV #50.,-(SP) ;;PUSH #50. ON STACK
2581 011026 000402 BR 6$
2582 5$:
2583 011030 012746 000144 MOV #100.,-(SP) ;;PUSH #100. ON STACK
2584 011034 004037 022150 6$: JSR RO,WTBGND ;LOOK FOR XMDNE
2585 *****
2586 011040 000430 BR ENDSEL ;NOT FOUND-EXIT.
2587 011042 005737 002246 TST BUBCT ;DONE WITH SCREEN?
2588 011046 001007 BNE 3$ ;YES-EXIT
2589 011050 012703 005724 MOV #SETREV,R3 ;NO-ISSUE ENTER REVERSE VIDEO
2590 011054 004037 017472 JSR RO,LDXMIT ;ESCAPE SEQUENCE.
2591 011060 005237 002246 INC BUBCT ;INCREMENT SCREEN HALF FLAG.
2592 011064 000723 BR 2$ ;AND ISSUE SECOND HALF IN REV. VIDEO.
2593 011066 032737 000001 002162 3$: BIT #BIT00,IDENT ;IDENT = COPIER?
2594 011074 001412 BEQ ENDSEL ;NO
2595 011076 013746 002226 MOV ZERO,-(SP) ;;PUSH ZERO ON STACK
2596 011102 012746 000135 MOV #.CPYSC,-(SP) ;;PUSH #.CPYSC ON STACK
2597 011106 013746 002170 MOV ESCN,-(SP) ;;PUSH ESCN ON STACK
2598 011112 004037 014606 JSR RO,TESC
2599 011116 004037 021250 JSR RO,CKABRT ;CHECK FOR A PERIPHERAL ABORT.
2600 011122 105737 002234 ENDSEL: TSTB MODE ;IF IN MAN MODE DO NOT ENTER EOP.
2601 011126 001402 BEQ ENDPS
2602 011130 000137 003340 JMP ASTRT
2603 011134 042777 000100 170622 ENDPS: BIC #RENA,@VJCSR ;CLEAR REC.INT. BEFORE NEXT UNIT SELECT.
2604 .SBTTL END OF PASS ROUTINE
2605 *****
2606 ;*INCREMENT THE PASS NUMBER ($PASS)
2607 ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
2608 ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
2609 ;*IF SW12=1 INHIBIT TRACE TRAP
2610 ;*IF THERES A MONITOR GO TO IT
2611 ;*IF THERE ISN'T JUMP TO MODCA
2612
2613
2614 SEOP:
2615 011142 SCOPE
2616 011142 000004 CLR $TSTNM ;;ZERO THE TEST NUMBER
2617 011150 005037 001102 CLR $TIMES ;;ZERO THE NUMBER OF ITERATIONS
2618 011154 005237 001100 INC $PASS ;;INCREMENT THE PASS NUMBER
2619 011160 042737 100000 001100 BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
2620 011166 005327 DEC (PC)+ ;;LOOP?
    
```

| | | | | | | | |
|------|--------|--------|--------|--------|--|---------------------------------------|--|
| 2621 | 011170 | 000001 | | | | \$EOPCT: .WORD 1 | |
| 2622 | 011172 | 003031 | | | | BGT \$DOAGN | :::YES |
| 2623 | 011174 | 012737 | | | | MOV (PC)+,@(PC)+ | :::RESTORE COUNTER |
| 2624 | 011176 | 000001 | | | | \$ENDCT: .WORD 1 | |
| 2625 | 011200 | 011170 | | | | \$EOPCT | |
| 2626 | 011202 | 104401 | 011325 | | | TYPE \$ENDMG | :::TYPE 'END PASS #' |
| 2627 | 011206 | 013746 | 001100 | | | MOV \$PASS,-(SP) | :::SAVE \$PASS FOR TYPEOUT |
| 2628 | 011212 | 104405 | | | | TYPDS | :::GO TYPE--DECIMAL ASCII WITH SIGN |
| 2629 | 011214 | 104401 | 011322 | | | TYPE \$ENULL | :::TYPE A NULL CHARACTER |
| 2630 | 011220 | 013700 | 000042 | | | \$GET42: MOV @#42,R0 | :::GET MONITOR ADDRESS |
| 2631 | 011224 | 001414 | | | | BEQ \$DOAGN | :::BRANCH IF NO MONITOR |
| 2632 | 011226 | 005046 | | | | CLR -(SP) | :::INSURE THE 'T' BIT IS CLEAR |
| 2633 | 011230 | 012746 | 011236 | | | MOV #SCLR.T,-(SP) | :::SETUP FOR AN RTI OR RTJ |
| 2634 | 011234 | 000426 | | | | BR \$RTRN | :::GO DO AN RTI OR RTJ TO LOAD THE PSW |
| 2635 | | | | | | | :::WITH A CLEARED 'T' BIT |
| 2636 | 011236 | | | | | \$CLR.T: | |
| 2637 | 011236 | 013700 | 000042 | | | MOV @#42,R0 | :::INSURE R0 CONTAINS THE MONITORS |
| 2638 | 011242 | 001405 | | | | BEQ \$DOAGN | :::RETURN ADDRESS |
| 2639 | 011244 | 000005 | | | | RESET | :::CLEAR THE WORLD |
| 2640 | 011246 | 004710 | | | | \$ENDAD: JSR PC,(R0) | :::GO TO MONITOR |
| 2641 | 011250 | 000240 | | | | NOP | :::SAVE ROOM |
| 2642 | 011252 | 000240 | | | | NOP | :::FOR |
| 2643 | 011254 | 000240 | | | | NOP | :::ACT11 |
| 2644 | 011256 | | | | | \$DOAGN: | |
| 2645 | 011256 | 104400 | | | | TRAP | :::PUSH OLD PSW AND PC ON STACK |
| 2646 | 011260 | 042716 | 000020 | | | BIC #20,(SP) | :::CLEAR THE 'T' BIT |
| 2647 | 011264 | 032777 | 010000 | 167646 | | BIT #BIT12,@SWR | :::RUN WITH TRACE TRAP? |
| 2648 | 011272 | 001005 | | | | ONE 1\$ | :::BR IF NO |
| 2649 | 011274 | 005137 | 011320 | | | COM \$TBIT | :::IS IT TIME FOR TRACE TRAP |
| 2650 | 011300 | 100402 | | | | BMI 1\$ | :::BR IF NO |
| 2651 | 011302 | 052716 | 000020 | | | BIS #20,(SP) | :::SET TRACE TRAP |
| 2652 | 011306 | 012746 | 011314 | | | 1\$: MOV #SLOOP,-(SP) | :::JUMP TO START OF IEST |
| 2653 | 011312 | 000002 | | | | \$RTRN: RTI | :::RETURN--THIS IS CHANGED TO |
| 2654 | | | | | | | :::AN 'RTT' IF 'RTT' IS A LEGAL |
| 2655 | | | | | | | :::INSTRUCTION |
| 2656 | 011314 | | | | | \$LOOP: | |
| 2657 | 011314 | 000137 | | | | JMP @(PC)+ | :::RETURN |
| 2658 | 011316 | 002760 | | | | \$RTNAD: .WORD MODCA | |
| 2659 | 011320 | 000000 | | | | \$TBIT: .WORD 0 | :::'T' BIT STATE INDICATOR |
| 2660 | 011322 | 377 | 377 | 000 | | \$ENULL: .BYTE -1,-1,0 | :::NULL CHARACTER STRING |
| 2661 | 011325 | 015 | 042412 | 042116 | | \$ENDMG: .ASCIIZ <15><12>/END PASS #/ | |
| 2662 | 011332 | 050040 | 051501 | 020123 | | | |
| 2663 | 011340 | 000043 | | | | | |
| 2664 | | | | | | | :::***** |
| 2665 | | | | | | | :ROUTINE TO ECHO THE KEYBOARD. KEYS FOR TAB,BELL,CARRIAGE |
| 2666 | | | | | | | :AND LINE FEED ECHO A MNEMONIC, NON-DISPLAY CHAR. ECHO OCTAL |
| 2667 | | | | | | | :EQUIVALENTS AND DISPLAY CHAR. ECHO THEMSELVES. |
| 2668 | | | | | | | :(EXAMPLES-CHAR.,SPACE,ESC,SPACE OR 037,SPACE.) A |
| 2669 | | | | | | | :CONTROL C (003) WILL CAUSE A TEST EXIT. |
| 2670 | | | | | | | :::***** |
| 2671 | | | | | | | :::***** |
| 2672 | | | | | | | :::***** |
| 2673 | | | | | | | :::***** |
| 2674 | 011342 | 000004 | | | | I\$T21: SCOPE | |
| 2675 | 011344 | 012737 | 000001 | 001160 | | MOV #1,\$TIMES | :::DO 1 ITERATION |
| 2676 | 011352 | 012737 | 011360 | 001106 | | MOV #KYBD,\$LPADR | :::SET SCOPE LOOP ADDRESS |

```

2677
2678 011360 004037 017532          KYBD: JSR   RO,RESPTR
2679 011364 012702 027474          MOV   #DKYBD,R2      ;LOAD MESSAGE ADDRESS INR2
2680 011370 004037 020612          JSR   RO,DSMES      ;DISPLAY KEYBOARD MESSAGE
2681 011374 012703 030062          MOV   #DCNTZ,R3    ;ISSUE CONTROL C EXIT MESSAGE
2682 011400 004037 017472          JSR   RO,LDXMIT
2683 011404 012703 011642          MOV   #EXMAIN,R3
2684 011410 004037 017472          JSR   RO,LDXMIT    ;ISSUE EXIT MAINTENANCE MODE.
2685 011414 042737 077577 002262 KYSTRT: BIC  #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2686 011422 105777 021244          TSTB @ABUFP        ;SEE IF A CHAR. RECEIVED
2687 011426 001001          BNE  11$          ;YES-GO PROCESS IT
2688 011430 000001          WAIT          ;WAIT FOR A CH.
2689 011432 117701 021234          11$: MOVB @ABUFP,R1    ;GET RAW RECEIVED DATA
2690 011436 004037 022074          JSR   RO,EXTST     ;CHECK FOR EXIT CONDITIONS
2691 011442 000402          BR   10$          ;NO EXIT -CONTINUE.
2692 011444 000137 003340          JMP  ASTRT        ;EXIT TEST 4
2693 011450 105077 021216          10$: CLRB @ABUFP    ;CLEAR CHAR FROM BUFFER
2694 011454 032737 000400 002262 BIT   #ESC,VSTAT  ;CHAR.=ESC(033)?
2695 011462 001405          BEQ  12$          ;NO
2696 011464 005037 016200          CLR  ESAMB       ;YES - RESET ESC ASSEMBLY FLAG
2697 011470 012703 027467          MOV  #DESC,R3    ;LOAD ESC MESSAGE ADDRESS
2698 011474 000454          BR   KYBXMT
2699 011476 120127 000041          12$: CMPB R1,#41   ;CHAR. LESS THAN 41 OR
2700 011502 103415          BLO  2$          ;HIGHER THAN 176, GO ECHO
2701 011504 120127 000176          CMPB R1,#176    ;OCTAL EQUIVALENT
2702 011510 101012          BHI  2$
2703 011512 110177 004766          MOVB R1,@TBUFP  ;LOAD CHAR. IN XMIT BUFF.
2704 011516 004037 017406          JSR  RO,XMIT1    ;GO XMIT IT
2705 011522 112777 000040 004754 MOVB #40,@TBUFP ;LOAD A SPACE
2706 011530 004037 017406          JSR  RO,XMIT1    ;AND XMIT IT.
2707 011534 000727          BR   KYSTRT
2708 011536 120137 002006          2$:  CMPB R1,BEL   ;CHAR.=BELL?
2709 011542 001003          BNE  3$
2710 011544 012703 027743          MOV  #DBELL,R3  ;LOAD BELL MESSAGE ADDRESS
2711 011550 000426          BR   KYBXMT
2712 011552 120137 002014          3$:  CMPB R1,TAB   ;CHAR. =TAB?
2713 011556 001003          BNE  4$
2714 011560 012703 027724          MOV  #DTAB,R3  ;YES-ECHO 'TAB'
2715 011564 000420          BR   KYBXMT
2716 011566 123701 002010          4$:  CMPB R1,CARRT,R1 ;CHAR.=CARRIAGE RETURN?
2717 011572 001003          BNE  5$
2718 011574 012703 027731          MOV  #DCR,R3   ;YES - ECHO 'C/R'.
2719 011600 000412          BR   KYBXMT
2720 011602 120137 002012          5$:  CMPB R1,LFED   ;CHAR.=LINE FEED?
2721 011606 001003          BNE  6$
2722 011610 012703 027736          MOV  #DLF,R3   ;NO CHECK FOR CONTROL Z
2723 011614 000404          BR   KYBXMT    ;YES - ECHO 'L/F'.
2724 011616 004037 020706          6$:  JSR  RO,BINQCT ;CONVERT BINARY TO OCTAL
2725 011622 012703 002222          MOV  #SVR1,R3
2726 011626 042737 077577 002262 KYBXMT: BIC  #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2727 011634 004037 017472          JSR  RO,LDXMIT  ;GO XMIT BUFFER
2728 011640 000665          BR   KYSTRT    ;WAIT FOR NEXT CHAR.
2729
2730          ;SEQUENCE TO EXIT MAINTENANCE MODE.
2731 011642 033 117 141 EXMAIN: .BYTE .ESC,.O,.DMAIN,0
2732 011645 000

```



```

2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744 011646 000004
2745 011650 012737 000001 001160
2746 011656 012737 011664 001106
2747
2748 011664 012702 030126
2749 011670 004037 020612
2750 011674 012703 011642
2751 011700 004037 017.72
2752 011704 004037 021942
2753 011710
2754 011710 013746 002226
2755 011714 013746 002152
2756 011720 013746 002170
2757 011724 004037 014606
2758 011730 013701 016500
2759 011734 012705 000041
2760 011740 042737 077577 002262
2761 011746 013701 016500
2762 011752 012703 000204
2763 011756 004037 020522
2764 011752 013721 002010
2765 011766 013721 002012
2766 011772 012737 000206 016506
2767 012000 052777 040000 167756
2768 012006 032737 000001 002262
2769 012014 001774
2770 012016 004037 022074
2771 012022 000402
2772 012024 000137 003340
2773 012030 004037 021250
2774 012034 122705 000177
2775 012040 001337
2776 012042 000734
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788 012044 000004
  
```

```

;*****
;ROUTINE TO UTILIZE THE VT61 AS A PRINTER CONTROLLER.
;AFTER TEST MESSAGE IS DISPLAYED, THE TEST WAITS
;FOR A C/R BEFORE ACTUALLY ENTERING TEST. A PATTERN
;OF INCREMENTING, ROLLING CHAR. WILL BE OUTPUTTED UNTIL A
;CONTROL C(003) IS RECEIVED.
;*****
;*****
TST2: SCOPE
      MOV #1,$TIMES ;;DO 1 ITERATION
      MOV #TPRNT,$LPADR ;;SET SCOPE LOOP ADDRESS
TPRNT: MOV #DPRNT,R2 ;LOAD PRINTER MESSAGE ADDRESS
      JSR RO,DSMES ;AND ISSUE IT
      MOV #EXMAIN,R3
      JSR RO,LDXMIT ;ISSUE EXIT MAINTENANCE MODE.
      JSR RO,GTCR ;GO SET CARRIAGE RETURN
3$:   MOV ZERO,-(SP) ;;PUSH ZERO ON STACK
      MOV EPNT,-(SP) ;;PUSH EPNT ON STACK
      MOV ESCN,-(SP) ;;PUSH ESCN ON STACK
      JSR RO,TEC
      MOV TBBUF,R1 ;LOAD R1 WITH XMIT BUFFER
4$:   MOV #41,R5 ;R5=1ST CHAR
5$:   BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
      MOV TBBUF,R1
      MOV #132.,R3 ;R3= LINE WIDTH
      JSR RO,BLDINA ;GO BUILD A SLIDING PATTERN.
      MOV CARRT,(R1)+ ;LOAD A C/R AND L/F
      MOV LNFED,(R1)+
      MOV #134.,XMCNT ;SET UP TO XMIT BY BYTES.
      BIS #XENA,@VJCSR
      BIT #XMDNE,VSTAT ;WAIT FOR XMIT DONE
      BEQ #-6
      JSR RO,EXTST ;CHECK FOR EXIT REQUEST.
      BR 6$ ;NO-CONTINUE
      JMP ASTRT ;YES-EXIT TEST!!
6$:   JSR RO,CKABRT ;CHECK FOR A PERIPHERAL ABORT.
      CMPB #177,R5 ;EXCEEDED PATT. LIMIT?
      BNE 5$ ;NO
      BR 4$ ;YES RESET IT
;*****
;ROUTINE TO LOOP DATA/COMMANDS FROM THE VT61 BACK TO
;THE VT61. DATA TRANSMISSIONS RESULTING FROM A ESC
;SEQUENCE WILL ALSO BE LOOPED AND WILL ENTER THE SCREEN
;AT THE CURSOR POSITION.THIS TEST CAN BE USED TO SIMULATE,
;OR CREATE, SPECIFIC SCREEN PATTERNS AND OPERATIONS.
;*****
;*****
TST3: SCOPE
  
```

```

2789 012046 012737 000001 001160      MOV    #1,$TIMES      ;;DO 1 ITERATION
2790 012054 012737 012062 001106      MOV    #LPTST,$LPADR  ;;SET SCOPE LOOP ADDRESS
2791
2792 012062 004037 017532      LPTST: JSR    RO,RESPTR  ;RESET POINTERS
2793 012066 012702 027751      MOV    #DLOOP,R2     ;LOAD LOOP MESSAGE ADDRESS
2794 012072 004037 020612      JSR    RO,DSMES       ;DISPLAY IT
2795 012076 012703 011642      MOV    #EXMAIN,R3
2796 012102 004037 017472      JSR    RO,LDXMIT      ;ISSUE EXIT MAINTENANCE MODE.
2797 012106 004037 021644      JSR    RO,LOOP        ;GO LOOP VT61
2798 012112 000137 003340      JMP    ASTRT          ;ENTER MAN MODE VIA SCOPE ROUTINE.
2799
2800      ;:*****
2801
2802      ;PRODUCTION KEYBOARD TEST. ALL KEYS MUST BE DEPRESSED
2803      ;IN THE SEQUENCE INDICATED ON THE SCREEN. ALL ERRORS
2804      ;OR MISTAKES ARE DISPLAYED IN OCTAL POSITIONAL FORMAT AND THE
2805      ;CORRECT KEY POSITION IN THE ROW IS DISPLAYED IN DECIMAL.
2806      ;THIS TEST IS RUN IN MAINTENANCE MODE, THEREFORE THE KEYS
2807      ;WILL ECHO THEIR POSITION, NOT THEIR INDICATED MNEMONIC. 10
2808      ;ERRORS WILL CAUSE AN AUTOMATIC EXIT FROM TEST.
2809      ;:*****
2810
2811      ;:*****
2812 012116 000004      TST24: SCOPE
2813 012120 012737 000001 001160      MOV    #1,$TIMES      ;;DO 1 ITERATION
2814 012126 012737 012134 001106      MOV    #PDKBD,$LPADR  ;;SET SCOPE LOOP ADDRESS
2815
2816
2817 012134 012702 030344      PDKBD: MOV    #DKBD ,R2
2818 012140 004037 020612      JSR    RO ,DSMES      ;DISPLAY KEYBOARD TEST MESSAGE.
2819 012144 005037 002246      CLR    BUBCT          ;CLEAR ERROR COUNT LOCATION.
2820 012150 005005      CLR    R5
2821
2822 012152 016504 012276      DOAROW: MOV   DTTBL(R5),R4 ;SET UP 'GOOD' CHAR. POINTER
2823 012156 016503 012250      MOV   MSTBL(R5),R3
2824 012162 001414      BEQ   FEXIT           ;MESSAGE WAS ZERO-EXIT.
2825 012164 100421      BMI   CLMAIN          ;IF MESSAGE IS -1,CLEAR MAINT. MODE.
2826 012166 004037 017472      JSR   RO,LDXMIT       ;ISSUE 'ROW OR FUNCTION' MESSAGE.
2827 012172 004037 021376      JSR   RO,CKKBD        ;GO CHECK IT.
2828 012176 123727 002246 000012      CMPB  BUBCT,#10.     ;TEN ERROR EXIT?
2829 012204 103401      BLO   1$              ;NO-CONTINUE.
2830 012206 000402      BR    FEXIT           ;YES-EXIT TEST.
2831 012210 005725      1$:   TST  (R5)+        ;INCREMENT OFFSET.
2832 012212 000757      BR    DOAROW          ;NO-DO NEXT ROW/FUNCTION.
2833 012214 012702 030113      FEXIT: MOV   #DEXT,R2
2834 012220 004037 020612      JSR   RO,DSMES        ;ISSUE EXIT MESSAGE
2835 012224 000137 003340      JMP   ASTRT
2836 012230 012703 012244      CLMAIN: MOV  #RSMIN,R3 ;SET UP TO EXIT MAINT. MODE.
2837 012234 004037 017472      JSR   RO,LDXMIT
2838 012240 005725      TST  (R5)+           ;INCREMENT OFFSET.
2839 012242 000743      BR    DOAROW          ;NOW TEST CONTROL AND SHIFT FUNCTIONS.
2840 012244      033      117      141  RSMIN: .BYTE .ESC,.O,.DMAIN,0
2841 012247      000
2842
2843
2844      ;TABLE OF MESSAGE ADDRESSES.
  
```

```

2845
2846 012250 030547 030654 030711 MSTBL: .WORD DTOP,DSEC,DTHRD,DBOT
2847 012256 031040
2848 012260 031116 031142 177777 .WORD DSPCE,DKPD,-1,DCONT,DLSHFT,DRSHFT,0
2849 012266 030770 030475 030601
2850 012274 000000
2851
2852 012276 031343 031364 031404 DT1BL: .WORD ROW1,ROW2,ROW3,ROW4,SPCB
2853 012304 031422 031444
2854 012310 031446 000000 031440 .WORD KYPD,0,CNTRA,SHFTA,SHFTA
2855 012316 031442 031442
2856
2857 ;:*****
2858 ;:SUBROUTINE TO ALLOW SETUP FROM MULTIPLE ENTRIES
2859 ;:*****
2860
2861 012322 SETA:
2862 .SBTTL INITIALIZE THE COMMON TAGS
2863 ;:CLEAR THE COMMON TAGS ($CMTAG) AREA
2864 012322 012706 001100 MOV #CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
2865 012326 005026 CLR (R6)+ ;:CLEAR MEMORY LOCATION
2866 012330 022706 001140 CMP #SWR,R6 ;:DONE?
2867 012334 001374 BNE -6 ;:LOOP BACK IF NO
2868 012336 012706 001100 MOV #STACK,SP ;:SETUP THE STACK POINTER
2869 ;:INITIALIZE A FEW VECTORS
2870 012342 012737 022300 000020 MOV #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
2871 012350 012737 000340 000022 MOV #340,@IOTVEC+2 ;:LEVEL 7
2872 012356 012737 022554 000030 MOV #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
2873 012364 012737 000340 000032 MOV #340,@EMTVEC+2 ;:LEVEL 7
2874 012372 012737 024160 000034 MOV #STRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
2875 012400 012737 000340 000036 MOV #340,@TRAPVEC+2 ;:LEVEL 7
2876 012406 012737 023770 000024 MOV #SPWRDN,@PWRVEC ;:POWER FAILURE VECTOR
2877 012414 012737 000340 000026 MOV #340,@PWRVEC+2 ;:LEVEL 7
2878 012422 013737 011176 011170 MOV $ENDCT,$EOPCT ;:SETUP END-OF-PROGRAM COUNTER
2879 012430 005037 001160 CLR $TIMES ;:INITIALIZE NUMBER OF ITERATIONS
2880 012434 005037 001162 CLR $ESCAPE ;:CLEAR THE ESCAPE ON ERROR ADDRESS
2881 012440 112737 000001 001115 MOVB #1,$ERMAX ;:ALLOW ONE ERROR PER TEST
2882 ;:INITIALIZE THE 'T-BIT' TRAP VECTOR. THEN LOAD LOCATION '$RTRN', IN
2883 ;:THE 'END-OF-PASS' ($EOP) ROUTINE, WITH A 'RTI' OR 'RTT'.
2884 012446 012737 011312 000014 MOV #RTRN,@TBITVEC ;:SET 'T' BIT VECTOR TO $RTRN
2885 012454 012737 000340 000016 MOV #340,@TBITVEC+2 ;:LEVEL 7
2886 012462 012737 000002 011312 MOV #RTI,$RTRN ;:SET $RTRN TO A RTI
2887 012470 012737 012516 000010 MOV #65$,@RESVEC ;:TRY TO DO A RTT
2888 012476 005046 CLR -(SP) ;:DUMMY PS
2889 012500 012746 012506 MOV #64$,-(SP) ;:AND PC
2890 012504 000006 RTT ;:TRY THE RTT
2891 012506 012737 000006 011312 64$: MOV #RTT,$RTRN ;:RTT IS LEGAL--SET $RTRN TO A RTT
2892 012514 000402 BR 66$
2893 012516 062706 000010 65$: ADD #10,SP ;:RTT ILLEGAL--CLEAN OFF THE STACK
2894 012522 012737 000012 000010 66$: MOV #RESVEC+2,@RESVEC ;:RESTORE TRAP CATCHER
2895 012530 005037 011320 CLR $TBIT ;:CLEAR 'T' BIT SWITCH
2896 012534 012737 012534 001106 MOV #,$LPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
2897 012542 012737 012542 001110 MOV #,$LPERR ;:SETUP THE ERROR LOOP ADDRESS
2898 ;:SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2899 ;:EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
2900 012550 013746 000004 MOV @ERRVEC,-(SP) ;:SAVE ERROR VECTOR
  
```

```

2901 012554 012737 012610 000004      MOV    #67$,@#ERRVEC    ;;SET UP ERROR VECTOR
2902 012562 012737 177570 001140      MOV    #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
2903 012570 012737 177570 001142      MOV    #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
2904 012576 022777 177777 166334      CMP    #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
2905 012604 001012                BNE    69$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2906                                ;;AND THE HARDWARE SWR IS NOT = -1
2907 012606 000403                BR     68$            ;;BRANCH IF NO TIMEOUT
2908 012610 012716 012616      67$:   MCV    #68$, (SP)    ;;SET UP FOR TRAP RETURN
2909 012614 000002                RTI
2910 012616 012737 000176 001140      68$:   MOV    #SWREG,SWR    ;;POINT TO SOFTWARE SWR
2911 012624 012737 000174 001142      MOV    #DISPREG,DISPLAY
2912 012632 012637 000004      69$:   MOV    (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
2913
2914      .SBTTL  TYPE PROGRAM NAME
2915      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2916 012636 005227 177777      INC    #-1            ;;FIRST TIME?
2917 012642 001024                BNE    70$            ;;BRANCH IF NO
2918 012644 022737 011246 000042      CMP    #SENDAD,@#42  ;;ACT-11?
2919 012652 001420                BEQ    70$            ;;BRANCH IF YES
2920 012654 104401 012662      TYPE    ,71$         ;;TYPE ASCIZ STRING
2921 012660 000415                BR     70$            ;;GET OVER THE ASCIZ
2922      ;;71$: .ASCIZ <CRLF>#CZVTJBO DJ11 VT61 EXER#<CRLF>
2923      70$:
2924 012714 104401 024300      TYPE    ,STUPM      ;;ISSUE SET-UP MESSAGE.
2925 012720 012737 012746 000010      MOV    #TRPA,@#10   ;;AND VECTOR
2926 012726 000230                SPL    0              ;;PROCESSOR IS 11/45?
2927 012730 012737 000010 020512      MOV    #8.,PMULT    ;;YBES-DELAY MULTIPLIER = 8      (REV. B0)
2928 012736 012737 177777 002270      MOV    #-1,PTYPE    ;;11/35 OR 11/40      (REV. B0)
2929 012744 000424                BR     RTRP
2930
2931 012746 022626      TRPA:  POP2SP        ;;NO
2932 012750 012737 013000 000010      MOV    #TRPB,@#10   ;;RELOAD TRAP ADDRESS
2933 012756 006737 002220                SXT    CHR           ;;PROCESSOR IS 11/40 OR 35?
2934 012762 012737 000003 020512      MOV    #3,PMULT    ;;YES-DELAY MULTIPLIER = 3      (REV. B0)
2935 012770 012737 000000 002270      MOV    #0,PTYPE    ;;11/45 OR 11/70      (REV. B0)
2936 012776 000407                BR     RTRP
2937
2938 013000 022626      TRPB:  POP2SP
2939 013002 012737 000001 020512      MOV    #1,PMULT    ;;MULT. = 1
2940 013010 012737 000001 002270      MOV    #1,PTYPE    ;;11/04 OR 11/34      (REV. B0)
2941 013016 012737 000012 000010      RTRP:  MOV    #12,@#10 ;;RESTORE TRAP CATCHER
2942 013024 105737 002234                TSTB   MODE         ;;CHECK MODE FOR CORRECT EXIT.
2943 013030 001402                BEQ    70$
2944 013032 000137 002340                JMP    MANSA        ;;EXIT TO MANUAL SELECT
2945 013036 000137 002304      70$:   JMP    AUTOA        ;;EXIT TO AUTO MODE.
2946      ;;*****
2947      ;;THIS SUBROUTINE WILL VERIFY EACH ADDRESS AS IT IS
2948      ;;ENTERED VIA THE KEYBOARD WHEN IN MANUAL MODE.
2949      ;;*****
2950
2951 013042 020227 160000      TMNAD:  CMP    R2,#160000  ;;INSURE ADDRESS IS IN RANGE.
2952 013046 103407                BLO    BDEXT        ;;ITS NOT-TYPE A ? AND EXIT.
2953 013050 012737 013064 000004      MOV    #BDEXTA,@#4  ;;SET UP TRAP EXIT.
2954 013056 005712                TST    @R2          ;;CHECK THE ADDRESS.
2955 013060 000240                NOP
2956 013062 000405                BR     ALEXT        ;;IF WE GOT THIS FAR ITS OK.

```

```

2957
2958 013064 022626          BDEXTA: POP2SP          ;ADDRESS TRAPPED-PURGE IT, TYPE A
2959 013066 104401 030342  BDEXT:  TYPE          ,QMRK ;QUESTION MARK, RESTORE TRAP
2960 013072 012700 002354          MOV          #BLDADD,R0 ;LOCATION, AND EXIT TO BUILD THE
2961 013076 012737 000006 000004 ALEXT:  MOV          #6,@#4 ;NEXT ENTRY.
2962 013104 000200          RTS          R0
2963
2964 ;:*****
2965 ;:THIS ROUTINE MAPS ALL POSSIBLE DJ11 ADDRESSES AND STORES
2966 ;:THEM IN A TABLE (INTAB). ALL ADDRESSES WHICH DO NOT
2967 ;:RESULT IN TIMEOUTS ARE STORED.
2968 ;:*****
2969
2970 013106 012701 000300  TRPVEC: MOV          #300,R1 ;START AT BEG. OF FLOATING VECTORS
2971 013112 012702 000302          MOV          #302,R2
2972 013116 012703 000004          MCV          #4,R3 ;R3 CONTAINS IOT TRAP INST.
2973 013122 010221          1$:  MOV          R2,(R1)+ ;START LOADING ADDRESSES
2974 013124 010321          MOV          R3,(R1)+ ;LOAD THE TRAP
2975 013126 062702 000004          ADD          #4,R2 ;ASSUME 4 REGISTERS PER INTERFACE
2976 013132 020127 001000          CMP          R1,#1000 ;DONE?
2977 013136 002771          BLT          1$ ;NO CONTINUE LOADING TRAPS
2978 013140 005037 000006          CLR          @#6
2979 013144 012737 013204 000004          MOV          #TPENT,@#4 ;SET UP TIME-OUT TRAP ADDRESS
2980 013152 005001          CLR          R1 ;CLEAR THE TABLE POINTER
2981 013154 012705 001572          MOV          #INTAB,R5 ;R5=DESTINATION TABLE
2982 013160 013702 001760  FADD:  MOV          STRIAB,R2 ;PUT THE ADDRESS TO BE TESTED IN R2
2983 013164 023702 001762  TRPE:  CMP          ENDTAB,R2 ;HAVE WE EXCEEDED END OF TABLE ADDRESS?
2984 013170 103407          BLO          TBLCK ;YES GET NEXT BASE ADDRESS.
2985 013172 005712          TST          @R2 ;ADDRESS THE DEVICE IF POSSIBLE
2986 013174 010225          MOV          R2,(R5)+ ;IF WE GOT THIS FAR THERE IS A DEVICE THERE-SAVE IT
2987 013176 062702 000010  FADD1: ADD          #10,R2 ;INCREMENT TO THE NEXT POSSIBLE ADDRESS
2988 013202 000770          BR          TRPE ;GO TEST THE NEXT ADDRESS
2989 013204 022626          TPENT: POP2SP          ;RESTORE THE STACK AND TEST
2990 013206 000773          BR          FADD1 ;NEXT ADDRESS
2991 013210 005015          TBLCK: CLR          (R5) ;SET UP TABLE TERMINATOR OF ZEROS.
2992 013212 000200          RTS          R0
2993 ;:*****
2994 ;:THIS ROUTINE WILL INSURE THAT THE DEVICE(DJ11)
2995 ;:WILL INTERRUPT WHEN XMIT INT. ENABLE BIT IS SET.
2996 ;:*****
2997
2998 013214 005046          CDEV:  CLR          -(SP) ;CLEAR THE PSW,LSI11 STYLE.
2999 013216 012746 013224          MOV          #100$,-(SP)
3000 013222 000002          RTI
3001 013224 012737 000004 000004 100$:  MOV          #4,@#4 ;INSTALL IOT TRAP INST. AT LOCATION 4.
3002 013232 012737 013350 000020          MOV          #TDEV,@#IOTVEC ;SET UP IOT TRAP EXIT ADDRESS
3003 013240 012737 000340 000022          MOV          #340,@#IOTVEC+2 ;SET PSW TO 7-ALLOW NO OTHER INTERRUPTS
3004 013246 000005          RESET ;INSURE ALL XMIT FLAGS HIGH.
3005 013250 012703 001552          MOV          #VVECT,R3 ;VECTOR STORAGE ADDRESS SET
3006 013254 012702 001572          MOV          #INTAB,R2 ;PRIMARY DEVICE TABLE ADDRESS SET
3007 013260 012704 001632          MOV          #DJLNE,R4 ;DJ11 LINE TABLE SET.
3008 013264 012705 001562          MOV          #DJTBL,R5 ;DEVICE TABLE ADDRESS SET.
3009 013270 012701 001764  CDEVA: MOV          #VJCSR,R1 ;VT61 DEVICE ADDRESS SET.
3010 013274 005712          TST          (R2) ;CHECKED ALL DEVICES?
3011 013276 001574          BEQ          AOUT ;YES-EXIT
3012 013300 100403          BMI          1$ ;INSURE ADDRESS IS IN PROPER RANGE(1/XXXX)

```

```

3013
3014 013302 062702 000002      ADD      #2,R2      ;ADDRESS IS DEFINITELY NOT GOOD -PURGE
3015 013306 000770              BR      CDEVA      ;AND LOOK FOR ANOTHER.
3016 013310 004037 014314      1$:     JSR      RO,LDADD ;LOAD NEXT ADDRESSES TO BE CHECKED
3017 013314 012701 001200      MOV      #1200,R1  ;NOW USE R1 AS FAILSAFE COUNTER
3018 013320 052777 177777 166442      BIS      #177777,@VXTCR ;ENABLE ALL LINES TO BE SCANNED.
3019 013326 052777 000400 166430      BIS      #XSCN,@VJCSR  ;ENABLE THE XMITTER SCANNER.
3020 013334 052777 040000 166422      BIS      #XENA,@VJCSR  ;SET XMIT ENABLE
3021 013342 005301              DEC      R1        ;IF DEVICE DOES NOT INTERRUPT WITHIN
3022 013344 001376              BNE     .-2        ;APPROX. 200US IT IS NOT A DJ11.
3023 013346 000750              BR      CDEVA      ;THEREFORE, GO TRY ANOTHER DEVICE.
3024 013350 042777 040000 166406 TDEV:   BIC      #XENA,@VJCSR ;CLEAR XMIT ENABLE.
3025 013356 042777 000400 166400      BIC      #XSCN,@VJCSR ;DISABLE THE XMIT SCANS.
3026 013364 005077 166400      CLR      @VXTCR    ;CLEAR XMIT LINE SCAN REG.
3027 013370 162716 000010      SUB      #10,(R6)  ;RESET TO RECEIVER VECTOR ADDRESS
3028 013374 012613      MOV      (R6)+,(R3) ;STORE IT IN VECTOR TABLE(VVECT).
3029 013376 005726      TST     (R6)+     ;POP THE OLD PSW AND DISCARD
3030 013400 022626      POP2SP ;POP THE ADD. AND PS. PRIOR TO INTERRUPT.
3031
3032 ;:*****
3033 ;:THIS ROUTINE IS A QUICK TEST OF ANY DJ11 ENCOUNTERED
3034 ;:A DATA PATTERN WILL BE RUN ON ALL ENTRIES IN INTAB
3035 ;:*****
3035 013402 005046              CLR      -(SP)     ;CLEAR THE PSW,LSI11 STYLE.
3036 013404 012746 013412      MOV      #100$,-(SP)
3037 013410 000002              RTI
3038 013412 012301      100$:   MOV      (R3)+,R1  ;GET THE RECEIVE VECTOR ADDRESS
3039 013414 012721 015146      MOV      #RECAD,(R1)+ ;AND STORE SAME.
3040 013420 012721 000340      MOV      #340,(R1)+  ;SET RECEIVE PSW TO 7.
3041 013424 012721 015240      MOV      #TSMAD,(R1)+ ;STORE THE XMIT VECTOR ADDRESS
3042 013430 012711 000340      MOV      #340,(R1)  ;SET XMIT PSW TO 7.
3043 013434 012737 000001 001776      MOV      #1,TSTLINE ;SET UP TO TEST LINE 0 FIRST.
3044 013442 005037 020366      CLR      HDFLG
3045 013446 012737 000001 002250 LNECK:  MOV      #BIT00,TPREG ;TPREG IS NOW DATA PATTERN OF 1.
3046 013454 005001              CLR      R1        ;SET UP FAILSAFE DELAY.
3047 013456 052777 000010 166300      BIS      #MCLR,@VJCSR ;CLEAR SILO AND UARTS.
3048 013464 000240              NOP              ;CLEAR PROPOGATION TIME
3049 013466 013777 001776 166274      MOV      TSTLINE,@VXTCR ;LOAD THE LINE TO BE CHECKED.
3050 013474 052777 000401 166262      BIS      #SCAN,@VJCSR  ;ENABLE XMIT AND RECIEVE SCANNERS.
3051 013502 052777 000100 166254      BIS      #RENA,@VJCSR  ;SET RECEIVE ENABLE.
3052 013510 052777 040004 166246      BIS      #TCOMB,@VJCSR ;ENABLE XMIT INT. AND MAINT MODE.
3053 013516 105737 002250      1$:     TSTB   TPREG      ;SEE IF COMPLETE PATTERN XMITTED.
3054 013522 001436      BEQ     GDAD      ;YES GO STORE THIS ADDRESS
3055 013524 005301              DEC      R1        ;CYCLE TIMEOUT DELAY
3056 013526 001373      BNE     1$       ;NOT YET 'TIMEOUT',KEEP CYCLING.
3057 013530 042777 040000 166226      BIC      #XENA,@VJCSR  ;CLEAR INTERRUPT ENABLES ON BAD UNITS.
3058 013536 042777 000100 166220      BIC      #RENA,@VJCSR
3059 013544 005737 020366      TST     HDFLG
3060 013550 001010      BNE     2$       ;ADDRESSES ISSUED?
3061 013552 104401 024554      TYPE    ,DLERR    ;YES-ISSUE LINES.
3062 013556 013746 001764      MOV      VJCSR,-(SP) ;ISSUE DJ11 FAILURE MESSAGE.
3063 ;:TYPE BD. ADDRESS
3064 013562 104403      TYPOS   ;GO TYPE--OCTAL ASCII
3065 013564 006              .BYTE   6        ;:TYPE 6 DIGIT(S)
3066 013565 001              .BYTE   1        ;:TYPE LEADING ZEROS
3067 013566 104401 001171      TYPE    ,%CRLF
3068 013572 004037 021004      2$:     JSR      RO,CONVLN ;CONVERT BINARY LINE TO OCTAL LINE.

```

```

3069 013576 000337 002000          SWAB      OCTALNE      ;MOVE OCTAL LINE # INTO POSITION.
3070 013602 013746 002000          MOV       OCTALNE,-(SP) ;:SAVE OCTALNE FOR TYPEOUT
3071                                     ;:TYPE A BAD LINE
3072 013606 104403          TYPOS     ;:GO TYPE--OCTAL ASCII
3073 013610          003          .BYTE    3          ;:TYPE 3 DIGIT(S)
3074 013611          000          .BYTE    0          ;:SUPPRESS LEADING ZEROS
3075 013612 005237 020366          INC      HDFLG
3076 013616 000402          BR       BDAD        ;GO TRY ANOTHER LINE OR ADDRESS.
3077 013620 013724 001776          GDAD: MOV   TSTLNE,(R4)+ ;STORE THE GOOD LINE
3078 013624 006337 001776          BDAD: ASL   TSTLNE     ;SET UP TO CHECK NEXT LINE.
3079 013630 001306          BNE     LNECK       ;IF NOT ZERO-GO CHECK IT.
3080 013632 012724 177777          MOV     #-1,(R4)+   ;STORE DJ11 LINE SEPARATOR.
3081 013636 013725 001764          MOV     VJCSR,(R5)+ ;STORE THE DJ11 ADDRESS.
3082 013642 005037 020366          CLR     HDFLG       ;SET UP TO TYPE AN ADDRESS.
3083 013646 042777 000401 166110          BIC     #SCAN,@VJCSR ;DISABLE REC. AND XMIT SCANNERS.
3084 013654 052777 000010 166102          BIS     #MCLR,@VJCSR ;CLEAR SILO AND UARTS
3085 013662 104401 001171          TYPE    $CRLF
3086 013666 000600          BR       CDEVA      ;:CHECK ANOTHER DJ11
3087 013670 005015          ADUT: CLR   (R5)     ;SET A ZERO TABLE TERMINATOR.
3088 013672 005014          CLR     (R4)       ;SET THE LINE TABLE TERMINATOR
3089 013674 012737 000006 000004          MOV     #6,@#4     ;RESTORE LOCATION 4 TO HALT CONDITION
3090 013702 005037 000006          CLR     @#6        ;TO CATCH ERRORS AND ILLEGAL INTERRUPTS.
3091 013706 012737 022300 000020          MOV     $$SCOPE,@#IOTVEC ;RELOAD IOT VECTOR FOR SCOPE
3092 013714 012737 000340 000022          MOV     #340,@#IOTVEC+2 ;LOOP.
3093 013722 012701 000300          MOV     #300,R1
3094 013726 012702 000302          MOV     #302,R2
3095 013732 010221          1$: MOV    R2,(R1)+
3096 013734 005021          CLR     (R1)+      ;RESTORE HALTS TO ALL LOCATIONS CONTAINING IOTS
3097 013736 062702 000004          ADD     #4,R2
3098 013742 020127 001000          CMP     R1,#1000   ;TO LOCATION 1000
3099 013746 103771          BLO    1$
3100 013750 000005          RESET
3101 013752 000200          RTS     RO         ;CLEAR ALL FLAGS
3102
3103          ;:*****
3104          ;:INITIALIZATION ROUTINE FOR AUTO SELECTION. THIS ROUTINE
3105          ;:WILL INSURE THAT ALL DJ11S IN DJTBL HAVE A VT61 CONNECTED
3106          ;:ALL UNITS WHICH EITHER DO NOT OR DO NOT RESPOND WILL BE PURGED.
3107          ;:*****
3108 013754 012702 001562          INITA: MOV   #DJTBL,R2 ;R2 POINTS TO DJ11 ADDRESS TABLE
3109 013760 012703 001632          MOV   #DJLNE,R3      ;POINTER TO DJ11 LINES.
3110 013764 012701 001764          11$: MOV  #VJCSR,R1   ;POINTER TO VT61 DJ11
3111 013770 052777 000010 165766          BIS  #MCLR,@VJCSR   ;CLEAR SILO AND UARTS.
3112 013776 000240          NOP                ;CLEAR PROPOGATION TIME
3113
3114          TST   (R2)     ;SEE IF ALL CHECKED
3115 014002 001456          BEQ   INTXT        ;YES-EXIT
3116 014004 004037 014314          JSR  RO,LDADD      ;NO-GO LOAD THE ADDRESSES
3117 014010 022713 177777          12$: CMP  #-1,(R3)  ;AT A LINE TABLE TERMINATOR?
3118 014014 001004          BNE   13$         ;NO-CONTINUE TESTING THIS ADDRESS.
3119 014016 005723          TST  (R3)+        ;IT IS-BUMP THE POINTER AND GET NEXT
3120 014020 005077 165740          CLR  @VJCSR       ;SHUT DOWN DJ11 AFTER SAMPLING COMPLETE.
3121 014024 000757          BR   11$         ;DJ11 ADDRESS(IF ANY).
3122 014026 011337 001776          13$: MOV  (R3),TSTLNE ;LOAD LINE TO BE TESTED.
3123 014032 004037 021004          JSR  RO,CONVLN    ;CONVERT BINARY LINE TO OCTAL LINE.
3124 014036 012377 165726          MOV  (R3)+,@VXTCR ;ENABLE LINE TO BE SCANNED.

```


| | | | | | | | | |
|------|--------|--------|--------|--------|---------|-------|---------------|--------------------------------------|
| 3125 | 014042 | 052777 | 000401 | 165714 | | BIS | #SCAN,@VJCSR | ::ENABLE XMIT AND REC SCANNERS. |
| 3126 | 014050 | 004037 | 016706 | | | JSR | RO,ZFLAG | ::ISSUE ESCZ AND LOOK FOR RESPONSE. |
| 3127 | 014054 | | | | 2\$: | | | |
| 3128 | 014054 | 012637 | 002220 | | | MOV | (SP)+,CHRD | ::POP STACK INTO CHRD |
| 3129 | 014060 | 100414 | | | | BMI | 5\$ | ::TIMEOUT OCCURRED NO CHARACTER |
| 3130 | 014062 | 123727 | 002220 | 000140 | | CMPB | CHRD,#140 | ::CHECK IDENT FOR VT61 IDENTIFIERS |
| 3131 | 014070 | 103410 | | | | BLO | 5\$ | ::NOT A VT61-SET UP TO PURGE ADDRESS |
| 3132 | 014072 | 123727 | 002220 | 000172 | | CMPB | CHRD,#172 | ::IDENTS ARE SMALL A THRU Z |
| 3133 | 014100 | 101004 | | | | BHI | 5\$ | ::NOT A VT61-PURGE |
| 3134 | 014102 | | | | 4\$: | | | |
| 3135 | 014102 | 012637 | 002220 | | | MOV | (SP)+,CHRD | ::POP STACK INTO CHRD |
| 3136 | 014106 | 001375 | | | | BNE | 4\$ | |
| 3137 | 014110 | 000737 | | | | BR | 12\$ | ::TEST ANOTHER ADDRESS |
| 3138 | 014112 | | | | 5\$: | | | |
| 3139 | 014112 | 012637 | 002220 | | | MOV | (SP)+,CHRD | ::POP STACK INTO CHRD |
| 3140 | 014116 | 001375 | | | | BNE | 5\$ | |
| 3141 | 014120 | 162703 | 000002 | | | SUB | #2,R3 | ::RESET LINE POINTER. |
| 3142 | 014124 | 010346 | | | | MOV | R3,-(SP) | ::PUSH R3 ON STACK |
| 3143 | 014126 | 012746 | 000001 | | | MOV | #1,-(SP) | ::PUSH #1 ON STACK |
| 3144 | 014132 | 004037 | 014534 | | | JSR | RO,BBLUP | |
| 3145 | 014136 | 000724 | | | | BR | 12\$ | ::TRY ANOTHER DJ11 LINE. |
| 3146 | 014140 | 022737 | 177777 | 001632 | INTXT: | CMP | #-1,DJLNE | ::GOOD LINE FROM 1ST ADDRESS? |
| 3147 | 014146 | 001021 | | | | BNE | EXINT | ::YES-BEGIN TESTING. |
| 3148 | 014150 | 022737 | 177777 | 001634 | | CMP | #-1,DJLNE+2 | ::SEE IF SECOND DJ HAS GOOD LINES. |
| 3149 | 014156 | 001403 | | | | BEQ | NOUNIT | ::NO GOOD LINES ON 2ND DJ11. |
| 3150 | 014160 | 005737 | 001634 | | | TST | DJLNE+2 | ::ZERO TERMINATOR FOUND? |
| 3151 | 014164 | 001012 | | | | BNE | EXINT | ::NO-FIRST DJ11 HAS GOOD LINES. |
| 3152 | 014166 | 104401 | 024465 | | NOUNIT: | TYPE | ,NOVT | ::NO-ISSUE NO VT61 MESSAGE. |
| 3153 | 014172 | 012737 | 005670 | 020514 | | MOV | #3000.,DCJUNT | ::SET DELAY TO 30 SEC. |
| 3154 | 014200 | 004037 | 020452 | | | JSR | RO,DELAY | ::AND DO IT. |
| 3155 | 014204 | 062700 | 000004 | | | ADD | #4,RO | ::SET UP 'NO VT61 FOUND' EXIT |
| 3156 | 014210 | 000200 | | | | RTS | RO | |
| 3157 | 014212 | 012702 | 001562 | | EXINT: | MOV | #DJTBL,R2 | ::LOAD AND ISSUE GOOD ADDRESSES |
| 3158 | 014216 | 005712 | | | | TST | (R2) | ::INSURE A GOOD ADDRESS. |
| 3159 | 014220 | 001762 | | | | BEQ | NOUNIT | ::NONE FOUND-EXIT |
| 3160 | 014222 | 012703 | 001632 | | | MOV | #DJLNE,R3 | ::LOAD TABLE ADDRESS OF GOOD LINES. |
| 3161 | 014226 | 104401 | 024414 | | | TYPE | ,DUNTST | ::OF RESPONSIVE VT61S. |
| 3162 | 014232 | | | | 1\$: | | | |
| 3163 | 014232 | 012246 | | | | MOV | (R2)+,-(SP) | ::SAVE (R2)+ FOR TYPEOUT |
| 3164 | | | | | | | | ::TYPE AN ADDRESS |
| 3165 | 014234 | 104403 | | | | TYPOS | | ::GO TYPE--OCTAL ASCII |
| 3166 | 014236 | 006 | | | | .BYTE | 6 | ::TYPE 6 DIGIT(S) |
| 3167 | 014237 | 001 | | | | .BYTE | 1 | ::TYPE LEADING ZEROS |
| 3168 | 014240 | 104401 | 001171 | | | TYPE | ,\$CRLF | |
| 3169 | 014244 | 012337 | 001776 | | 2\$: | MOV | (R3)+,TSTLNE | ::LOAD A LINE TO PRINT |
| 3170 | 014250 | 022737 | 177777 | 001776 | | CMP | #-1,TSTLNE | ::IF LINE IS A TERMINATOR, DO NOT |
| 3171 | 014256 | 001411 | | | | BEQ | 3\$ | ::DISPLAY - GO SET UP NEXT DJ11. |
| 3172 | 014260 | 004037 | 021004 | | | JSR | RO,CONVLN | ::CONVERT IT TO A OCTAL #. |
| 3173 | 014264 | 000337 | 002000 | | | SWAB | OCTLNE | ::MOVE 1ST INTO PRINT POSITION. |


```

3174 014270 013746 002000      MOV      OCTLNE,-(SP)      ;;SAVE OCTLNE FOR TYPEOUT
3175                                ;;TYPE A GOOD LINE
3176 014274 104403              TYPOS                      ;;GO TYPE--OCTAL ASCII
3177 014276      003            .BYTE      3              ;;TYPE 3 DIGIT(S)
3178 014277      000            .BYTE      0              ;;SUPPRESS LEADING ZEROS
3179 014300 000761              BR        2$              ;NO-TYPE ANOTHER LINE.
3180 014302 104401 001171      3$:      TYPE      ,SCLF
3181 014306 005712              TST      (R2)              ;AT END OF GOOD UNITS?
3182 014310 001350              BNE      1$              ;NO PRINT ANOTHER ADDRESS.
3183 014312 000200              RTS      R0
3184
3185                                ;:*****
3186                                ;SUBROUTINE TO LOAD 4 ADDRESSES FROM THE LOCATION AT (R2).
3187                                ;TO A LOCATION POINTED TO BY R1.EXITES WITH R2 INC. BY 2.
3188                                ;:*****
3189
3190
3191
3192 014314 012211      LDADD:  MOV      (R2)+,(R1)      ;LOAD THE ADDRESS
3193 014316 012111      1$:      MOV      (R1)+,(R1)      ;STORE AN ADDRESS
3194 014320 062711 000002      ADD      #2,(R1)          ;INCREMENT THE ADDRESS
3195 014324 020127 001772      CMP      R1,#VXBUF        ;LOADED 4?
3196 014330 002772              BLT      1$              ;NO LOAD ANOTHER
3197 014332 000200              RTS      R0              ;YES-EXIT
3198
3199                                ;:*****
3200                                ;ROUTINE TO RECEIVE CHARACTER(S). ENTERED WITH
3201                                ;NUMBER OF CHARACTERS TO RECEIVE ON THE STACK.
3202                                ;ROUTINE EXITS WITH CHARACTER(S) ON STACK. IF A
3203                                ;PROGRAM TIME-OUT (100 M.S.) OCCURS BEFORE A CHARACTER
3204                                ;IS RECEIVED ROUTINE EXITS WITH -1 ON STACK. FORMAT
3205                                ;FOR DATA IS (BYTE2, BYTE1) ETC. A WORD OF ZEROS TERMINATES
3206                                ;DATA STRING ON THE STACK. SOM/EOM, IF SENT, ARE RECEIVED
3207                                ;BUT NOT STORED.
3208
3209                                ;:*****
3210
3211
3212 014334 012637 002260      RECTM:  MOV      (SP)+,ROSVE      ;;POP STACK INTO ROSVE
3213 014340 012637 002246      MOV      (SP)+,@#BUBCT    ;;POP STACK INTO @#BUBCT
3214 014344 013746 002226      MOV      ZERO,-(SP)      ;;PUSH ZERO ON STACK
3215 014350 005037 002220      1$:      CLR      CHR                      ;CLEAR CHARACTER STORAGE LOCATION.
3216 014354 005037 002256      CLR      DLAY              ;SET UP FAILSAFE DELAY
3217 014360 032777 000200 165376 3$:      BIT      #RECDN,@VJCSR    ;SEE IF DONE FLAG SET
3218 014366 001007              BNE      4$
3219 014370 005337 002256      DEC      DLAY              ;DECREMENT FAILSAFE CNTR.
3220 014374 001371              BNE      3$              ;NOT AT ZERO-CONTINUE WAITING.
3221 014376 012737 177777 002220 31$:      MOV      #-1,CHR          ;SET UP FOR FAILSAFE EXIT.
3222 014404 000446              BR        RECX              ;EXIT ROUTINE.
3223 014406 017737 165354 002220 4$:      MOV      @VRBUF,CHR        ;STORE THIS CHARACTER.
3224 014414 042737 170200 002220      BIC      #170200,CHR      ;STRIP ALL BUT CHAR. AND LINE #.
3225 014422 123737 002001 002221      CMPB    OCTLNE+1,CHR+1    ;RECEIVED FROM CORRECT LINE?
3226 014430 001347              BNE      1$              ;NO-IGNORE THIS CHAR.
3227 014432 122737 000057 002220      CMPB    #SLSH,CHR         ;RECEIVED A IDENT SLASH(5)?
3228 014440 001007              BNE      41$              ;NO-STORE A CHARACTER.
3229 014442 105337 002247      DECB    BUBCT+1          ;DECREMENT ALLOWABLE SLASH COUNT.

```

```

3230 014446 001753          BEQ      31$          ;COUNT EQUAL ZERO-SET UP ERROR EXIT.
3231 014450 123727 002247 000213      CMPB    BUBCT+1,#139. ;RECEIVED FIRST SLASH?
3232 014456 103734          BLO     1$           ;YES-IGNORE THIS ONE.
3233 014460 122737 000002 002220 41$:    CMPB    #SOM,CHRD    ;IS CHAR. ACTUALLY SOM?
3234 014466 001003          BNE     5$           ;NO
3235 014470 105237 002246          INCB    BUBCT        ;YES -SET UP TO RECEIVE EOM ALSO
3236 014474 000725          BR      1$           ;AND RECEIVE NEXT CHAR.
3237 014476 122737 000004 002220 5$:    CMPB    #EOM,CHRD    ;CHAR. = EOM?
3238 014504 001410          BEQ     RECEXA       ;YES- DO NOT PUSH IT ON STACK
3239 014506 105337 002246          DECB    BUBCT        ;DECREMENT CHARACTER COUNT.
3240 014512 001403          BEQ     RECEX        ;COUNT=0. EXIT WERE DONE.
3241 014514 013746 002220          MOV     CHRD,-(SP)   ;:PUSH CHR D ON STACK
3242 014520 000713          BF      1$           ;GO READ AGAIN.
3243
3244 014522          RECEX:
3245 014522 013746 002220          MOV     CHRD,-(SP)   ;:PUSH CHR D ON STACK
3246 014526          RECEXA:
3247 014526 013746 002260          MOV     ROSVE,-(SP) ;:PUSH ROSVE ON STACK
3248 014532 000200          RTS     RO
3249
3250
3251 ;:*****
3252 ;:THIS ROUTINE WILL 'BUBBLE UP' XX WORDS TO
3253 ;:ELIMINATE NON-RESPONSIVE ADDRESSES. ENTERED
3254 ;:WITH ADDRESS TO BE 'BUBBLED' TO ON THE STACK. LOCATIONS
3255 ;:ELIMINATED WILL BE FILLED WITH ZEROS. THE STACK MUST ALSO
3256 ;:BE LOADED WITH THE NUMBER OF POSITIONS TO BUBBLE.
3257 ;:*****
3258
3259 014534          BBLUP:
3260 014534 012637 002260          MOV     (SP)+,ROSVE ;:POP STACK INTO ROSVE
3261 014540 012637 002246          MOV     (SP)+,BUBCI ;:POP STACK INTO BUBCT
3262 014544 012637 002244          MOV     (SP)+,TOADD ;:POP STACK INTO TOADD
3263 014550 010446          MOV     R4,-(SP)    ;:PUSH R4 ON STACK
3264 014552 013704 002244 2$:    MOV     @#TOADD,R4   ;PUT LAST GOOD DJ11 ADDRESS IN R4
3265 014556 012437 002220          MOV     (R4)+,CHRD  ;MOVE NEXT WORD TO CHRD FOR STORAGE
3266 014562 012464 177774 1$:    MOV     (R4)+,-4(R4) ;BUBBLE UP DATA.
3267 014566 001375          BNE     1$           ;BUBBLE UNTIL ZERO BYTE MOVED.
3268 014570 005337 002246          DEC     BUBCT        ;SUBTRACT ONE FROM BUBBLE COUNT.
3269 014574 001366          BNE     2$           ;IF BUBBLE COUNT NOT ZERO - DO AGAIN.
3270 014576          3$:
3271 014576 012604          MOV     (SP)+,R4     ;:POP STACK INTO R4
3272 014600 013746 002260          MOV     ROSVE,-(SP) ;:PUSH ROSVE ON STACK
3273 014604 000200          RTS     RO           ;YES-EXIT
3274
3275 ;:*****
3276 ;:THIS ROUTINE OUTPUTS THE ESC SEQUENCE FOUND ON
3277 ;:THE STACK. A WORD OF ZEROS MUST TERMINATE THE SEQUENCE.
3278 ;:FORMAT FOR STACK WORD IS SEQ-ESC, IE-XXX033.
3279 ;:*****
3280
3281 014606          TESC:
3282 014606 012637 002260          MOV     (SP)+,ROSVE ;:POP STACK INTO ROSVE
3283 014612 010437 002246          MOV     R4,BUBCT    ;:SAVE R4.
3284 014616 112777 000002 165146 165146 1$:    MOV     #SOM,@VXBUF ;SEND A START OF MESSAGE.
3285 014624 012705 177777          MOV     #-1,R5      ;ALL ONES TO THE CHECK LOCATION.

```

| | | | | | | | | |
|------|--------|--------|--------|--------|------|--------------|--------------|---------------------------------------|
| 3286 | 014630 | 012604 | | | MOV | (R6)+,R4 | | :GET COMMAND FROM STACK. |
| 3287 | 014632 | 001415 | | | BEQ | 3\$ | | :IF ZERO TERMINATOR FOUND-EXIT. |
| 3288 | 014634 | 110405 | | | MOVB | R4,R5 | | :LOAD CHECK BYTE. |
| 3289 | 014636 | 105704 | | 2\$: | TSTB | R4 | | :CHECK BYTE FOR A ZERO. |
| 3290 | 014640 | 001406 | | | BEQ | 20\$ | | :IF ZERO_DO NOT XMIT IT. |
| 3291 | 014642 | 032777 | 100000 | 165114 | BIT | #TRDY,@VJCSR | | |
| 3292 | 014650 | 001774 | | | BEQ | .-6 | | :WAIT FOR XMIT READY BIT |
| 3293 | 014652 | 110477 | 165114 | | MOVB | R4,@VXBUF | | :XMMIT A BYTE. |
| 3294 | 014656 | 000304 | | 20\$: | SWAB | R4 | | :GET THE OTHER BYTE . |
| 3295 | 014660 | 120405 | | | CMPB | R4,R5 | | :IF GOOD COMPARE WE HAVE CHECKED BOTH |
| 3296 | 014662 | 001760 | | | BEQ | 1\$ | | :BYTES SO POP ANOTHER WORD. |
| 3297 | 014664 | 000764 | | | BR | 2\$ | | :GO XMIT ANOTHER BYTE |
| 3298 | 014666 | 032777 | 100000 | 165070 | 3\$: | BIT | #TRDY,@VJCSR | :SEE IF READY SET |
| 3299 | 014674 | 001774 | | | BEQ | .-6 | | |
| 3300 | 014676 | 012777 | 000004 | 165066 | MOV | #EOM,@VXBUF | | :SEND A EOM. |

```

3301 014704 032777 100000 165052 BIT #TRDY,@VJCSR ;SEE IF READY SET
3302 014712 001774 BEQ #-6
3303 014714 013704 002246 MOV BUBCT,R4 ;RESTORE R4.
3304 014720 013746 002260 MOV ROSVE,-(SP) ;;PUSH ROSVE ON STACK
3305 014724 000200 RTS RO
3306
3307 ;:*****
3308 ;ROUTINE TO READ A CHARACTER FROM THE CONSOLE.
3309 ;EXITS WITH CHARACTER ON THE STACK.
3310 ;:*****
3311
3312 014726 CONRD:
3313 014726 012637 002260 MOV (SP)+,ROSVE ;;POP STACK INTO ROSVE
3314 014732 032777 000200 164204 BIT #RECDN,@STKS ;LOOK FOR DONE BIT
3315 014740 001774 BEQ #-6 ;WAIT FOR IT
3316 014742 117746 164200 MOVB @STKB,-(R6) ;PUSH CHARACTER TO STACK
3317 014746 042716 000200 BIC #200,(R6) ;STRIP ANY PARITY BIT.
3318 014752 013746 002260 MOV ROSVE,-(SP) ;;PUSH ROSVE ON STACK
3319 014756 000200 RTS RO
3320
3321 ;:*****
3322 ;MANUAL TEST SELECT MONITOR
3323 ;SELECTS TESTS TO BE EXECUTED FROM THOSE ENTERED IN
3324 ;INITIAL DIALOGUE. IF TEST 377 WAS REQUESTED THE TESTS WILL
3325 ;REPEAT INFINITELY.
3326 ;:*****
3327
3328 014760 105737 002234 MONIT: TSTB MODE ;TEST MODE SWITCH
3329 014764 001012 BNE 1$ ;MANUAL MODE
3330 014766 023737 002236 002240 CMP FTLCNT,ALWCNT ;COMPARE FATAL XMITTS WITH ALLOWED.
3331 014774 103405 BLO 100$ ;FATALS LESS THAN ALLOWED-CONTINUE.
3332 014776 104401 026313 TYPE ,DABRT ;ISSUE ABORT MESSAGE.
3333 015002 000005 200$: RESET ;CLEAR ALL INTERFACE FLAGS.
3334 015004 000137 012322 JMP SETA ;SET UP TO RESTART TEST.
3335 015010 000200 100$: RTS RO ;AUTO MODE
3336 015012 005726 1$: TST (R6)+ ;POP THE STACK
3337 015014 022626 POP2SP ;POP SCOPE RETURN AND VECTOR
3338 015016 005037 002236 CLR FTLCNT ;DO NOT INC. FATAL COUNT IN MANUAL MODE.
3339 015022 032777 000200 164114 10$: BIT #RECDN,@STKS ;CONSOLE ACTIVE?
3340 015030 001407 BEQ 11$
3341 015032 117701 164110 MOVB @STKB,R1 ;STORE INPUT BUFFER
3342 015036 042701 000200 BIC #200,R1 ;CLEAR THE PARITY BIT
3343 015042 122701 000003 CMPB #3,R1 ;CHAR. EQUAL ESC. C?
3344 015046 001755 BEQ 200$ ;YES-EXIT
3345 015050 117701 165156 11$: MOVB @TSTPTR,R1 ;GET THE NEXT TEST #
3346 015054 001005 BNE 2$ ;NOT AT END OF LIST
3347 015056 042777 000100 164700 12$: BIC #RENA,@VJCSR ;CLEAR REC. INTERRUPTS BEFORE NEXT UNIT SELECT.
3348 015064 000137 002760 JMP MODCA ;END OF LIST-GO SET UP NEXT 61
3349 015070 100004 2$: BPL 3$ ;WAS TEST REPEAT REQUESTED?
3350 015072 012737 001572 002232 MOV #INTAB,TSTPTR ;YES-RESET TEST POINTER
3351 015100 000750 BR 10$ ;AND GET FIRST TEST SELECTED
3352 015102 005301 3$: DEC R1 ;ADJUST OFFSET
3353 015104 006301 ASL R1 ;USE TEST # TO FORM ADDRESS OFFSET
3354 015106 016137 024230 015144 MOV TSTADD(R1),JMPADD+2 ;LOAD NEW ADDRESS
3355 015114 062737 000002 015144 ADD #2,JMPADD+2 ;BYPASS INITIAL SCOPE LOOP
3356 015122 005237 002232 INC TSTPTR ;INCREMENT TEST OPINTER

```

```

3357 015126 005037 177776 CLR PSW ;SET NON-INT. PRIORITY TO ZERO
3358 015132 005046 CLR -(SP) ;CLEAR THE PSW,LSI11 STYLE.
3359 015134 012746 015142 MOV #JMPADD,-(SP)
3360 015140 000002 RTI
3361 015142 000137 015142 JMPADD: JMP JMPADD ;EXIT TO NEXT SELECTED TEST
3362 :*****
3363 :*****
3364 ;FOLLOWING ROUTINES ARE INTERRUPT HANDLERS FOR THE
3365 ;DJ11 QUICK-TEST.
3366 :*****
3367 :*****
3368 015146 117737 164614 002220 RECAD: MOVB @VRBUF,CHRD ;GET THE RECEIVED CHAR.
3369 015154 042737 000200 002220 BIC #200,CHRD ;CLEAR ANY PARITY.
3370 015162 123737 002250 002220 CMPB TPREG,CHRD ;COMPARE RECEIVED TO XMITTED
3371 015170 001407 BEQ UPD4 ;AND UPDATE PATTERN IF OK.
3372 015172 042777 040004 164564 TOFF: BIC #TCOMB,@VJCSR ;DATA ERROR OCCURED OR WE ARE DONE
3373 015200 042777 000100 164556 BIC #RENA,@VJCSR ;EITHER WAY-EXIT.
3374 015206 000002 REEX: RTI
3375 015210 052777 040000 164546 UPD4: BIS #XENA,@VJCSR ;ENABLE XMIT INT.
3376 015216 106337 002250 ASLB TPREG ;UPDATE DATA PATTERN.
3377 015222 032737 000200 002250 BIT #BIT07,TPREG ;ROTATED TO PARITY BIT?
3378 015230 001766 BEQ REEX ;NO-CONTINUE TESTING
3379 015232 005037 002250 CLR TPREG ;YES-SET UP COMPLETE FLAG
3380 015236 000755 BR TOFF ;AND EXIT.
3381 015240 113777 002250 164524 TSMAD: MOVB TPREG,@VXBUF ;XMIT DATA
3382 015246 042777 040000 164510 BIC #XENA,@VJCSR ;CLEAR XMIT INT. UNTIL LAST BIT REC.
3383 015254 000002 RTI
3384
3385 :*****
3386 ;RECEIVE INTERRUPT ROUTINE. ROUTINE WILL TRAP ALL ESC
3387 ;FUNCTIONS AND WILL SET FLAGS IN VSTAT FOR SOM, EOM, XON,
3388 ;XOFF, AND OTHER SPECIAL FUNCTIONS(SEE VSTAT TABLE DEFINITION).
3389 ;ALL INTERFACE STATUS ERRORS WILL BE REPORTED. MAXIMUM EXECUTION
3390 ;TIME FOR THIS ROUTINE IS 200 MICRO SECONDS. AV. = 100.
3391 ;UPON RECEIPT ON XON, XMTKIL BIT IS CHECKED IN VSTAT
3392 ;AND IF SET, WILL BE CLEARED AND XMIT INT. ENABLE SET.
3393 ;LOCATION ESAMB IS USED FOR ESC ASSEMBLY FLAGS. IE. BIT
3394 ;00 SET MEANS A033 WAS RECEIVED, BIT 01 SET MEANS AN ESCP
3395 ;SEQUENCE IS BEING ASSEMBLED. BIT 03
3396 ;SET INDICATES AND ESCAPE 0 SEQUENCE IS BEING ASSEMBLED.
3397 :*****
3398
3399 015256 INTRC: MOV R1,-(SP) ;;PUSH R1 ON STACK
3400 015256 010146 MOV R2,-(SP) ;;PUSH R2 ON STACK
3401 015260 010246 MOV @VRBUF,R1 ;USE R1 FOR STORAGE OF STATUS AND CH.
3402 015262 017701 164500 MOV R1,R2 ;SET UP LINE CHECK LOCATION.
3403 015266 010102 BIC #170377,R2 ;CLEAR ALL BUT LINE BITS.
3404 015270 042702 170377 CMP OCTLINE,R2 ;COMPARE CHAR LINE TO LINE UNDER TEST.
3405 015274 023702 002000 BEQ 13$ ;YES-EXAMINE IT.
3406 015300 001404 BR #ILLNE,VSTAT ;NO-SET ILLEGAL LINE FLAG AND EXIT.
3407 015302 052737 000310 002262 13$: BIC #200,R1 ;STRIP PARITY BIT.
3408 015310 00567 BR ERLNE ;CHECKSUM CALCULATION REQUESTED?
3409 015312 042701 000200 BIT #TXSUM,VSTAT ;NO
3410 015316 032737 000100 002262 BEQ 11$ ;YES-STORE CHAR. AND
3411 015324 001403 MOV R1,R5
3412 015326 010105

```

| | | | | | | | | |
|------|--------|--------|--------|--------|-------|------|--------------------------|---------------------------------------|
| 3413 | 015330 | 004037 | 021174 | | | JSR | RO,CALCK | :CALCULATE THE CHECKSUM. |
| 3414 | 015334 | 005237 | 032672 | | 11\$: | INC | ABUFP | :INCREMENT THE RAW DATA POINTER |
| 3415 | 015340 | 023727 | 032672 | 032756 | | CMP | ABUFP,#ABBUF+50. | :AT THE END OF BUFFER? |
| 3416 | 015346 | 001003 | | | | BNE | 12\$ | :NO |
| 3417 | 015350 | 012737 | 032674 | 032672 | | MOV | #ABBUF,ABUFP | :YES-RESET IT |
| 3418 | 015356 | 110177 | 015310 | | 12\$: | MOVB | R1,@ABUFP | :STORE THE RAW DATA |
| 3419 | 015362 | 001505 | | | | BEQ | 6\$ | :IF CHAR. IS NULL-GO STORE IT |
| 3420 | 015364 | 032737 | 000013 | 016200 | | BIT | #BIT00+BIT01+BIT03,ESAMB | :ESC OR ESC O? |
| 3421 | 015372 | 001152 | | | | BNE | ASESC | :YES-KEEP ASSEMBLING |
| 3422 | 015374 | 120137 | 002170 | | | CMPB | R1,ESCN | :BYTE = ESCN? |
| 3423 | 015400 | 101076 | | | | BHI | 6\$ | :NO-PROBABLY A DISPLAY CH.-STORE IT. |
| 3424 | 015402 | 001007 | | | | BNE | 1\$ | :NO-DECODE FOR XON,XOFF,SOM,EOM |
| 3425 | 015404 | 012737 | 000001 | 016200 | | MOV | #1,ESAMB | :YES SET ESC ASSEMBLY FLAG. |
| 3426 | 015412 | 052737 | 000400 | 002262 | | BIS | #ESC,VSTAT | :SET ESC RECEIVED FLAG |
| 3427 | 015420 | 000515 | | | | BR | RSTER | :AND EXIT |
| 3428 | 015422 | 120127 | 000023 | | 1\$: | CMPB | R1,#XOFF | :SEE IF RECEIVED BYTE WAS XOFF |
| 3429 | 015426 | 001004 | | | | BNE | 2\$ | :NO |
| 3430 | 015430 | 052737 | 100000 | 002262 | | BIS | #RXOFF,VSTAT | :YES, SET XOFF IN STATUS REG. |
| 3431 | 015436 | 000506 | | | | BR | RSTER | :EXIT |
| 3432 | 015440 | 120127 | 000021 | | 2\$: | CMPB | R1,#XON | :SEE IF BYTE WAS XON |
| 3433 | 015444 | 001016 | | | | BNE | 3\$ | :NO |
| 3434 | 015446 | 042737 | 100000 | 002262 | | BIC | #RXOFF,VSTAT | :YES, CLEAR XOFF IN VSTAT. |
| 3435 | 015454 | 032737 | 000200 | 002262 | | BIT | #XMKIL,VSTAT | :CHECK XMIT KILL BIT. |
| 3436 | 015462 | 001474 | | | | BEQ | RSTER | :NOT SET, EXIT |
| 3437 | 015464 | 052777 | 040000 | 164272 | | BIS | #XENA,@VJCSR | :SET XMIT INT. ENABLE. |
| 3438 | 015472 | 042737 | 000200 | 002262 | | BIC | #XMKIL,VSTAT | :CLEAR THE XMIT KILLED FLAG |
| 3439 | 015500 | 000465 | | | | BR | RSTER | :EXIT |
| 3440 | 015502 | 120127 | 000002 | | 3\$: | CMPB | R1,#SOM | :SEE IF BYTE WAS SOM |
| 3441 | 015506 | 001004 | | | | BNE | 4\$ | :NO |
| 3442 | 015510 | 052737 | 040000 | 002262 | 31\$: | BIS | #RSOM,VSTAT | :YES, SET SOM IN VSTAT. |
| 3443 | 015516 | 000456 | | | | BR | RSTER | :EXIT |
| 3444 | | | | | | | | |
| 3445 | 015520 | 120127 | 000004 | | 4\$: | CMPB | R1,#EOM | :WAS BYTE EOM? |
| 3446 | 015524 | 001012 | | | | BNE | 5\$ | :NO |
| 3447 | 015526 | 052737 | 020000 | 002262 | | BIS | #REOM,VSTAT | :NOW SET EOM IN VSTAT. |
| 3448 | 015534 | 013737 | 016172 | 016176 | | MOV | RBBUF,RBUFP | :RESET THE BUFFER POINTER. |
| 3449 | 015542 | 042737 | 000100 | 002262 | | BIC | #TXSUM,VSTAT | :CLEAR CHECKSUM REQUEST BIT. |
| 3450 | 015550 | 000441 | | | | BR | RSTER | :AND EXIT |
| 3451 | 015552 | 123701 | 002010 | | 5\$: | CMPB | CARRT,R1 | :CHAR. =CARRIAGE RETURN? |
| 3452 | 015556 | 001403 | | | | BEQ | 51\$ | :YES-GO SET END OF LINE FLAG |
| 3453 | 015560 | 123701 | 002012 | | | CMPB | LNFEED,R1 | :CHAR.= LINEFEED? |
| 3454 | 015564 | 001004 | | | | BNE | 6\$ | :NO- GO STORE IT |
| 3455 | 015566 | 052737 | 001000 | 002262 | 51\$: | BIS | #EPL,VSTAT | :SET END OF LINE INDICATOR |
| 3456 | 015574 | 000427 | | | | BR | RSTER | |
| 3457 | | | | | | | | |
| 3458 | 015576 | 023737 | 016176 | 016174 | 6\$: | CMP | RBUFP,REBUF | :IS CIRCULAR BUFFER FILLED? |
| 3459 | 015604 | 001003 | | | | BNE | 61\$ | :NO |
| 3460 | 015606 | 013737 | 016172 | 016176 | | MOV | RBBUF,RBUFP | :YES, RESET POINTER TO BEGINNING |
| 3461 | 015614 | 032737 | 000020 | 002262 | 61\$: | BIT | #COMGP,VSTAT | :RECEIVING GRAPHICS CHAR.? |
| 3462 | 015622 | 001402 | | | | BEQ | 7\$ | :NO |
| 3463 | 015624 | 162701 | 000137 | | | SUB | #137,R1 | :YES-SUBTRACT 137 FROM RECEIVED CHAR. |
| 3464 | | | | | | | | |
| 3465 | 015630 | 032737 | 000040 | 002262 | 7\$: | BIT | #REVID,VSTAT | :REVERSE VIDEO MODE? |
| 3466 | 015636 | 001402 | | | | BEQ | 70\$ | :NO STORE RECEIVED BYTE. |
| 3467 | 015640 | 052701 | 000200 | | | BIS | #200,R1 | :YES-FORCE BIT7 AS REV. VIDEO IND. |
| 3468 | 015644 | 110177 | 000326 | | 70\$: | MOVB | R1,@RBUFP | :STORE BYTE AND |

```

3469 015650 005237 016176          INC      RBUF      ;INCREMENT POINTER.
3470
3471 015654 032701 070000          RSTER:  BIT      #70000,R1 ;CHECK FOR STATUS ERROR
3472 015660 001414          BEQ      RECXT     ;NO, EXIT ROUTINE
3473
3474 015662 052737 004000 002262          BIS      #RSTT,VSTAT ;SET STATUS ERROR FLAG IN VSTAT
3475 015670 027727 000330 177777          ERLNE:  CMP      @STEP,#-1 ;IS ERROR TABLE FULL?
3476 015676 001405          BEQ      RECXT     ;YES, EXIT ROUTINE
3477 015700 010177 000320          MOV      R1,@STEP  ;NO, STORE STATUS ERR. AND CHECK
3478 015704 062737 000002 016224          ADD      #2,STTEP  ;INCREMENT STATUS ERR. POINTER
3479
3480 015712          RECXT:
3481 015712 012602          MOV      (SP)+,R2  ;;POP STACK INTO R2
3482 015714 012601          MOV      (SP)+,R1  ;;POP STACK INTO R1
3483 015716 000002          RTI          ;EXIT
3484 015720 032737 000002 016200          ADESC:  BIT      #2,ESAMB ;ASSEMBLING ESC P?
3485 015726 001063          BNE      AESCP     ;YES-GO GET LAST CH,
3486 015730 032737 000010 016200          BIT      #BIT03,ESAMB ;ASSEMBLING ESC O?
3487 015736 001062          BNE      AESCO     ;YES
3488 015740 122701 000120          CMPB    #120,R1   ;CH.= A P?
3489 015744 001004          BNE      10$      ;NO KEEP CHECKING
3490 015746 052737 000002 016200          BIS      #BIT01,ESAMB ;YES-SET ESCP ASSEMBLY FLAG
3491 015754 000737          BR       RSTER    ;AND EXIT
3492 015756 122701 000077          10$:  CMPB    #77,R1   ;CHAR.IS AN ESC ? ?
3493 015762 001403          BEQ      110$     ;YES-FAKE AN ESC O.
3494 015764 122701 000117          CMPB    #117,R1  ;CHAR = O?
3495 015770 001004          BNE      11$      ;NO
3496 015772 052737 000010 016200          110$: BIS      #BIT03,ESAMB ;YES SET ESC O ASSEMBLY FLAG
3497 016000 000725          BR       RSTER    ;AND EXIT
3498 016002 123701 002106          11$:  CMPB    RDCUR,R1 ;BYTE= CURSOR POSITION?
3499 016006 001004          BNE      1$       ;NO-
3500 016010 052737 000004 002262          BIS      #CURPOS,VSIAT ;YES-SET RECEIVED CURSOR POSITION.
3501 016016 000424          BR       CESAM
3502 016020 122701 000057          1$:  CMPB    #SLSH,R1  ;BYTE=TERMINAL ID ESC?
3503 016024 001004          BNE      2$       ;NO-CHECK FOR GRAPHICS SEQUENCE.
3504 016026 052737 000002 002262          BIS      #TRMID,VSTAT ;YES-SET TERM. IDENT FLAG IN VSTAT
3505 016034 000415          BR       CESAM
3506 016036 122701 000106          2$:  CMPB    #CKGP,R1  ;RECEIVED GRAPHICS CHAR. SEQUENCE?
3507 016042 001004          BNE      3$       ;NO
3508 016044 052737 000020 002262          BIS      #COMGP,VSTAT ;YES-SET GRAPHICS DATA FLAG.
3509 016052 000406          BR       CESAM
3510 016054 122701 000107          3$:  CMPB    #NCKGP,R1 ;RECEIVED RESET GRAPHICS SEQ.?
3511 016060 001003          BNE      CESAM    ;NO
3512 016062 042737 000020 002262          BIC      #COMGP,VSTAT ;YES-SET NORMAL CHAR. RECEIVE.
3513 016070 005037 016200          CESAM:  CLR      ESAMB  ;CLEAR ASSEMBLY FLAG.
3514 016074 000667          BR       RSTER    ;AND EXIT.
3515
3516 016076 110137 016230          AESCP:  MOVB    R1,STRP ;STORE ANY UNCHECKED FOR ESC. P
3517 016102 000772          BR       CESAM
3518
3519 016104 123701 002054          AESCO:  CMPB    EEMP,R1 ;BYTE=ESC O -REV. VIDEO- ?
3520 016110 001004          BNE      1$       ;NO
3521 016112 052737 000040 002262          BIS      #REVID,VSTAT ;YES-SET REVERSE VIDEO MODE IN VSTAT.
3522 016120 000763          BR       CESAM
3523
3524 016122 123701 002056          1$:  CMPB    DEMP,R1  ;BYTE=ESC O DISABLE REV. VIDEO MODE?
  
```

```

3525 016126 001004          BNE      2$          ;NO
3526 016130 042737 000040 002262  BIC      #REVID,VSTAT ;YES-CLEAR REVERSE VIDEO MODE IN VSTAT.
3527 016136 000754          BR       CESAM
3528 016140 122701 000171          2$:  CMPB    #CPABRT,R1    ;COPIER ABORT?
3529 016144 001403          BEQ     3$          ;YES-SET ABORT FLAG IN VSTAT
3530 016146 122701 000172          CMPB    #PRABRT,R1    ;PRINTER ABORT?
3531 016152 001004          BNE     4$          ;NO
3532 016154 052737 010000 002262  3$:  BIS     #PABRT,VSTAT ;YES-SET THE ABORT FLAG.
3533 016162 000742          BR       CESAM      ;AND EXIT.
3534 016164 110137 016226          4$:  MOVB    R1,STRO    ;STORE ESCAPE O COMMAND
3535 016170 000737          BR       CESAM
3536
3537 016172 000000          RBBUF: .WORD          ;ADDRESS OF STAT OF BUFFER
3538 016174 000000          REBUF: .WORD          ;ADDRESS OF END OF BUFFER.
3539 016176 000000          RBUFP: .WORD          ;READ BUFFER POINTER.
3540 016200 000000          ESAMB: .WORD 0      ;ESCAPE SEQ.ASSEMBLY AREA
3541
3542 016202 000010          STTER:
3543 016202 000000          0
3544 016204 000000          0
3545 016206 000000          0
3546 016210 000000          0
3547 016212 000000          0
3548 016214 000000          0
3549 016216 000000          0
3550 016220 000000          0
3551 016222 177777          .WORD  -1          ;STATUS REGISTER DELIMITER.
3552 016224 000000          STTEP: .WORD          ;STATUS ERROR POINTER.
3553 016226 000000          STRO:  .WORD  0      ;ESCAPE O STORAGE
3554 016230 000000          STRP:  .WORD          ;ESCAPE P STORAGE
3555
3556          ;:*****
3557          ;TRANSMIT INTERRUPT ROUTINE. IF XOFF BIT IS SET
3558          ;IN VSTAT TRANSMISSION WILL NOT OCCUR AND THIS ROUTINE
3559          ;WILL RESET XMIT INT. ENABLE. IF AFTER TRANSMISSION
3560          ;OF THE CHARACTER DURING THIS INTERRUPT CYCLE, THE
3561          ;XMIT COUNT (XMCNT) IS EQUAL TO ZERO,
3562          ;THE XMIT DONE BIT WILL BE SET IN VSTAT AND XMIT
3563          ;INT ENABLE BIT WILL CLEARED. TRANSMIT COUNT(XMCNT) MUST BE
3564          ;SET TO THE NUMBER OF BYTE/CHARACTER TO TRANSMIT.
3565          ;IF LOCATION BLKM IS SET TO 1001,A SOM WILL PRECEED THE
3566          ;DATA AND A EOM WILL FOLLOW IT. IF XMZER IS SET TO NON-
3567          ;ZERO, ALL DATA(INCLUDING ZEROS) WILL BE XMITTED.
3568          ;:*****
3569 016232 005737 002262          INTXM: TST     VSTAT    ;HAS 61 TRANSMITTED XOFF?
3570 016236 100004          BPL     NOKIL        ;NO XMIT ANOTHER
3571 016240 052737 000200 002262  BIS     #XMKIL,VSTAT ;SET XMIT KILLED BIT IN VSTAT
3572 016246 000310          BR      KIENA        ;GO KILL XMIT ENABLE
3573
3574 016250 105737 002265          NOKIL: TSTB    BLKM+1  ;SOM/EOM TRANSMIT?
3575 016254 001406          BEQ     NOSOM        ;NO
3576 016256 112777 000002 163506  MOVB    #SOM,@VXBUF  ;YES-ISSUE START OF MESSAGE.
3577 016264 105037 002265          CLRB    BLKM+1      ;AND CLEAR SOM FLAG.
3578 016270 000002          RTI
3579 016272 005737 016506          NOSOM: TST     XMCNT   ;XMITTED THE BUFFER?
3580 016276 001006          BNE     100$        ;NO-XMIT A NORMAL CHAR.

```



```

3581 016300 112777 000004 163464      MOVB  #EOM,@VXBUF      ;YES SEND EOM AND EXIT
3582 016306 105037 002264      CLRB  BLKM
3583 016312 000452      BR    2$
3584 016314 105777 000164      100$: TSTB  @TBUF      ;CHECK FOR CH.= ZERO. IF SO DO NOT XMIT
3585 016320 001016      BNE   1$           ;OR COUNT BYTE. OR ARE WE
3586 016322 005737 022146      TST   XMZER       ;XMITTING ZEROS?
3587 016326 001023      BNE   22$        ;YES-XMIT NEXT BYTE
3588 016330 023737 016504 016502      CMP   TBUF,TEBUF  ;AT END OF BUFFER?
3589 016336 001004      BNE   10$        ;NO
3590 016340 013737 016500 016504      MOV   TBUF,TBUF   ;YES-RESET BUFFER POINTER
3591 016346 000740      BR    NOKIL
3592 016350 005237 016504      10$: INC  TBUF
3593 016354 000735      BR    NOKIL      ;LOOK FOR NON-ZERO BYTE TO TRANSMIT.
3594
3595 016356 032737 002000 002262 1$: BIT  #CKSUM,VSTAT ;CHECKSUM REQUESTED?
3596 016364 001404      BEQ  22$
3597 016366 117705 000112      MOVB  @TBUF,R5     ;YES,LOAD THE BYTE
3598 016372 004037 021174      JSR   RC,CALCK    ;AND CALCULATE THE NEW CHECKSUM.
3599 016376 117777 000102 163366 22$: MOVB  @TBUF,@VXBUF ;TRANSMIT A CHARACTER
3600 016404 023737 016504 016502      CMP   TBUF,TEBUF  ;AT END OF CIRCULAR BUFFER?
3601 016412 001004      BNE   11$        ;NO
3602 016414 013737 016500 016504      MOV   TBUF,TBUF   ;YES, RESET IT TO START.
3603 016422 000402      BR    12$        ;BY-PASS INCREMENT BUFF. POINTER
3604 016424 005237 016504      11$: INC  TBUF     ;INCREMENT BUFFER POINTER.
3605
3606 016430 005337 016506      12$: DEC  XMCNT    ;DECREMENT THE TRANSMIT COUNT
3607 016434 001401      BEQ  2$           ;YES,CLEANUP,REQUEST ERRORS AND EXIT.
3608 016436 000002      RTI
3609 016440 105737 002264      2$: TSTB  BLKM     ;NO, CONTINUE
3610 016444 001014      BNE   TXEX       ;SOM/EOM XMIT?
3611 016446 052737 000001 002262      BIS   #XMDNE,VSTAT ;YES-DO NOT SET XMDNE UNTIL EOM SENT.
3612 016454 042737 002000 002262      BIC   #CKSUM,VSTAT ;SET THE DONE BIT IN VSTAT.
3613 016462 013737 016500 016504      MOV   TBUF,TBUF   ;CLEAR THE CHECKSUM FLAG WHEN DONE.
3614 016470 042777 040000 163266 KIENA: BIC  #XENA,@VJCSR ;RESET BUFFER POINTER.
3615 016476 000002      TXEX: RTI        ;CLEAR XMIT. INT. ENABLE
3616
3617
3618 016500 000000      TBUF: .WORD      ;CONTAINS INITIAL ADDRESS
3619 016502 000000      TEBUF: .WORD     ;CONTAIN LAST ADDRESS
3620 016504 000000      TBUF: .WORD     ;CONTAINS CURRENT LOCATION
3621
3622 016506 000000      XMCNT: .WORD 0   ;LOADED WITH NUMBER OF XMITS.
3623      ;:*****
3624
3625
3626      ;SUBROUTINE TO ISSUE RESET TO THE VT61, ENTERS MAINTENANCE MODE
3627      ;AND FORCES LINEAR ADDRESSING.
3628      ;:*****
3629
3630 016510 113737 001102 002266 RESETV: MOVB  $TSTNM,TSTNM ;LOAD THE TEST NUMBER IN ERROR PRINT AREA.
3631 016516 013746 002226      MOV   ZERO,-(SP)  ;:PUSH ZERO ON STACK
3632 016522 013746 002166      MOV   RESET,-(SP) ;:PUSH RESET ON STACK
3633 016526 013746 002112      MOV   ESCO,-(SP) ;:PUSH ESCO ON STACK
3634 016532 004037 014606      JSR   RO,TESC     ;GO XMIT IT
3635 016536 052737 100000 002262      BIS   #RXOFF,VSTAT ;TURN RECVR OFF
3636 016544 004037 016626      JSR   RO,GETON    ;GO LOOK FOR XON.

```

```

3637 016550 000405          BR      1$          ;FOUND IT.
3638 016552 005237 002236  INC      FTLCNT      ;ADD 1 TO FATAL XMIT COUNT.
3639 016556 010037 001120  MOV      RO,$GDADR   ;NO XON ISSUE XON ERROR
3640 016562 104017          ERROR    17
3641 016564          1$:
3642 016564 013746 002226  MOV      ZERO,-(SP)  ;;PUSH ZERO ON STACK
3643 016570 013746 002036  MOV      EMAIN,-(SP) ;;PUSH EMAIN ON STACK
3644 016574 013746 002112  MOV      ESCO,-(SP)  ;;PUSH ESCO ON STACK
3645 016600 013746 002046  MOV      DRECT,-(SP) ;;PUSH DRECT ON STACK
3646 016604 013746 002112  MOV      ESCO,-(SP)  ;;PUSH ESCO ON STACK
3647 016610 004037 014606  2$:     JSR      RO,TE$C
3648 016614 005037 002262  CLR      VSTAT       ;CLEAR INT. FLAGS AFTER TERMINAL RESET
3649 016620 005037 020366  CLR      HDFLG       ;CLEAR PRINT HEADER FLAG.
3650 016624 000200          RTS      RO
3651
3652          ;;*****
3653          ;SUBROUTINE TO WAIT FOR AN XON. NO XON EXIT IS PC +2.
3654          ;;*****
3655
3656 016626 012737 000454 002246  GETON:  MOV      #300.,BUBCT ;SET UP TO LOOK FOR 3 SEC.
3657 016634 032737 100000 002262  BIT      #RXOFF,VSTAT ;RECDV XON? (REV. B0)
3658 016642 001420          BEQ      GOTON       ;YES, EXIT (REV. B0)
3659 016644 105077 014022          CLRB    @ABUFP
3660 016650 127727 014016 000021  1$:     CMPB    @ABUFP,#XON ;RECEIVED A XON?
3661 016656 001412          BEQ      GOTON       ;YES-EXIT.
3662 016660 012737 000001 020514  MOV      #1,DCOUNT   ;NO-DELAY 10 M.S.
3663 016666 004037 020452          JSR      RO,DELAY
3664 016672 005337 002246          DEC      BUBCT       ;AT END OF DELAY?
3665 016676 001364          BNE     1$          ;NO
3666 016700 062700 000002          ADD     #2,RO       ;YES-SET UP ERROR EXIT.
3667 016704 000200          GOTON: RTS      RO
3668
3669          ;;*****
3670
3671          ;SUBROUTINE TO ISSUE ESCZ AND LOOK FOR A RESPONSE-EITHER
3672          ;A -1 OR THE RETURNED IDENT. THE -1 INDICATES NO
3673          ;RESPONSE FROM THE UNIT UNDER TEST.
3674          ;;*****
3675
3676          ZFLAG:
3677 016706 012637 016744          MOV     (SP)+,ROSV1  ;;POP STACK INTO ROSV1
3678 016712 013746 002226          MOV     @#ZERO,-(SP) ;;PUSH @#ZERO ON STACK
3679 016716 013746 002164          MOV     @#ESCZ,-(SP) ;;PUSH @#ESCZ ON STACK
3680 016722 004037 014606          JSR     RO,TE$C     ;GO ISSUE ESZ SEQUENCE
3681 016726 012746 106003          MOV     #106003,-(SP) ;;PUSH #106003 ON STACK
3682 016732 004037 014334          JSR     RO,RECTM    ;GO READ THE CHARACTER
3683 016736 013746 016744          MOV     ROSV1,-(SP) ;;PUSH ROSV1 ON STACK
3684 016742 000200          RTS      RO
3685 016744 000000          ROSV1: .WORD    0
3686
3687          ;;*****
3688          ;ROUTINE TO CHECK SOFTWARE STATUS REGISTER (VSTAT)
3689          ;RECEIVE FLAGS ONLY. ENTERED WITH ANTICIPATED
3690          ;STATUS WORD ON THE STACK.
3691          ;;*****
3692

```

```

3693 016746
3694 016746 012637 002260
3695 016752 010137 002222
3696 016756 010237 002224
3697
3698 016762 012601
3699 016764 013702 002262
3700
3701 016770 042702 003566
3702 016774 020102
3703 016776 001432
3704
3705 017000 010137 001124
3706 017004 013737 002262 001126
3707 017012 104003
3708
3709
3710
3711
3712
3713
3714
3715 017014 012701 016202
3716 017020 013702 016224
3717 017024 020102
3718 017026 001416
3719 017030 004037 017110
3720 017034 013737 001764 001120
3721 017042 017737 162716 001124
3722 017050 112137 001126
3723 017054 112137 001123
3724 017060 104002
3725 017062 000760
3726 017064 013701 002222
3727 017070 013702 002224
3728 017074 012737 016202 016224
3729 017102 013746 002260
3730 017106 000200
3731
3732
3733
3734
3735
3736
3737 017110 005037 001120
3738 017114 005037 001122
3739 017120 005037 001124
3740 017124 005037 001126
3741 017130 000200
3742
3743
3744
3745
3746
3747
3748

CKSFT:
MOV (SP)+,ROSVE ;;POP STACK INTO ROSVE
MOV R1,SVER1 ;;SAVE R1
MOV R2,SVER2 ;;SAVE R2

MOV (SP)+,R1 ;;POP STACK INTO R1
MOV VSTAT,R2 ;;SET R2 EQUAL TO VSTAT

BIC #003566,R2 ;;CLEAR NON-ERROR BITS
CMP R1,R2 ;;COMPARE ANTICIPATED TO ACTUAL.
BEQ NOER ;;NO UNUSAL BITS EXIT

MOV R1,$GDDAT ;;MOVE GOOD STATUS TO MESSAGE
MOV VSTAT,$BDDAT ;;MOVE BAD STATUS TO MESSAGE
ERROR 3 ;;ISSUE ERROR MESSAGE.

;:*****
;:ROUTINE TO PRINT THE STATUS REGISTER IN THE FOLLOWING
;:FORMAT: STATUS BITS (XXX 000), CHARACTER TRANSFERRED (000 X X)
;:*****

MOV #STTER,R1 ;;SET R1 EQUAL TO FIRST ENTRY
MOV STTER,R2 ;;SET R2 EQUAL LAST ENTRY
1$: CMP R1,R2 ;;ARE THEY EQUAL
BEQ NOER ;;YES-RESET POINTERS AND EXIT.
JSR RO,CLREG ;;CLEAR ERROR PRINT LOC.
MOV VJCSR,$GDADR ;;LOAD ADDRESS
MOV @VJCSR,$GDDAT ;;LOAD CSR
2$: MOVB (R1)+,$BDDAT ;;MOVE CHARACTER AND
MOVB (R1)+,$BDADR+1 ;;STATUS BITS TO ERROR REGISTERS.
ERROR 2 ;;ISSUE ERROR MESSAGE
BR 1$ ;;DO AGAIN
NOER: MOV SVER1,R1 ;;RESTORE R1 AND
MOV SVER2,R2 ;;R2.
MOV #STTER,STTER ;;RESET STATUS ERROR POINTER.
MOV ROSVE,-(SP) ;;PUSH ROSVE ON STACK
RTS RO ;;EXIT

;:*****
;:SUBROUTINE TO CLEAR ERROR/DATA OUTPUT LOCATIONS. NEEDED
;:ONLY WHEN DISPLAYING BYTES IN WORD LOCATIONS.
;:*****

CLREG: CLR $GDADR
CLR $BDADR
CLR $GDDAT
CLR $BDDAT
RTS RO

;:*****
;:SUBROUTINE TO TRANSMIT THE BUFFER AND WAIT FOR XMIT DONE
;:AND END OF RECEIVE MESSAGE. SUBROUTINE WILL LOOP IF LOCATION
;:RECITT IS PRE-LOADED WITH A NUMBER HIGHER THAN1(IE. MULTIPLE
;:RECEIVES CAN BE ACCOMPLISHED WITH ONLY ONE ENTRY TO SUB-
;:ROUTINE).WDSTOR AND BYSTOR ARE THE WORD(CURSOR POS.) AND BYTE
;:STORAGE LOCATIONS,RESPECTIVELY.DEFAULT STORAGE IS THE REC. BUFFER.

```

```

3749
3750 ;:*****
3751
3752 017132 XMREC:
3753 017132 010546 MOV R5, -(SP) ;: PUSH R5 ON STACK
3754 017134 012737 001001 002264 MOV #1001, BLKM ;: SET UP FOR A SOM/EOM TRANSMIT.
3755 017142 042737 077577 002262 BIC #77577, VSTAT ;: CLEAR ALL FLAGS BUT XOFF AND XMKIL.
3756 017150 013701 017404 MOV BYSTOR, R1 ;: LOAD THE STORAGE POINTERS
3757 017154 013702 017402 MOV WDSTOR, R2
3758 017160 052777 040000 162576 BIS #XENA, @VJCSR ;: SET INTERRUPT ENABLES
3759 017166 042737 061466 002262 XMITT: BIC #61466, VSTAT ;: CLEAR SOM, EOM, EPL, ESC, REV.VID., PARA. DELIM., IDENT, CUR.
3760 017174 005037 002256 1$: CLR DLAY ;: SET UP TIME OUT DELAY.
3761 017200 032737 000001 002262 BIT #XMDNE, VSTAT ;: IS XMIT DONE?
3762 017206 001015 BNE 3$ ;: YES-LOOK FOR RECEIVE DONE.
3763 017210 032737 020000 002262 2$: BIT #REOM, VSTAT ;: RECEIVED AN EOM?
3764 017216 001401 BEQ 20$ ;: NO
3765 017220 000435 BR CKSTR ;: YES-GO HANDLE DATA
3766 017222 032737 100000 002262 20$: BIT #RXOFF, VSTAT ;: NO- IS XOFF SET?
3767 017230 001761 BEQ 1$ ;: NO-STILL TRANSMITTING.
3768 017232 005337 002256 DEC DLAY ;: YES- RUN DELAY
3769 017236 001364 BNE 2$ ;: WAITING FOR XON
3770 017240 000416 BR XMAD2 ;: NO XON-REPORT VT61 FAILURE.
3771
3772 017242 013705 032672 3$: MOV ABUFP, R5 ;: LOAD CH. RECEIVED FLAG.
3773 017246 005037 002256 CLR DLAY ;: SET UP RECEIVE DELAY.
3774 017252 032737 020000 002262 4$: BIT #REOM, VSTAT ;: RECEIVE END OF MESSAGE?
3775 017260 001015 BNE CKSTR ;: YES-CHECK DATA STORAGE POINTERS
3776 017262 020537 032672 CMP R5, ABUFP ;: RECEIVED ANOTHER CHARACTER?
3777 017266 001365 BNE 3$ ;: YES-RESET CH. FLAG AND DELAY
3778 017270 005337 002256 5$: DEC DLAY ;: RUN DELAY
3779 017274 001366 BNE 4$ ;: AND KEEP LOOKING FOR EOM.
3780 017276 062700 000002 XMAD2: ADD #2, RO ;: TIME OUT OCCURRED-SET UP ERROR EXIT.
3781 017302 005237 002236 INC FTLCNT ;: INCREMENT FATAL XMIT COUNT.
3782 017306 004037 017532 JSR RO, RESPTR ;: AND REST ALL INTERRUPT POINTERS.
3783 017312 000422 BR CKVST
3784 017314 020102 CKSTR: CMP R1, R2 ;: STORAGE POINTERS CLEARED?
3785 017316 001413 BEQ CHKITT ;: YES--LEAVE DATA IN REC. BUFFER.
3786 017320 032737 000004 002262 BIT #CURPOS, VSTAT ;: RECEIVED A CURSOR POSITION?
3787 017326 001403 BEQ STRBYT ;: NO-GO STORE A BYTE.
3788 017330 017722 176636 MOV @RBBUF, (R2)+ ;: YES, STORE IT.
3789 017334 000404 BR CHKITT ;: AND CHECK ITERATION COUNT.
3790 017336 005701 STRBYT: TST R1 ;: STORING A CHAR?
3791 017340 001402 BEQ CHKITT ;: NO
3792 017342 117721 176624 MOV @RBBUF, (R1)+ ;: STORE A RECEIVED BYTE
3793 017346 005337 017400 CHKITT: DEC RECITT ;: DONE RECEIVING?
3794 017352 001305 BNE XMITT ;: NO-LOOP SUBROUTINE
3795 017354 004037 022254 JSR RO, CKOFF ;: SEE IS XOFF IS UP.
3796 017360 CKVST:
3797 017360 012746 060001 MOV #60001, -(SP) ;: PUSH #60001 ON STACK
3798 017364 004037 016746 JSR RO, CKSFT
3799 017370 004037 017532 JSR RO, RESPTR ;: RESET INTERRUPT POINTERS.
3800 017374 012605 MOV (SP)+, R5 ;: POP STACK INTO R5
3801 017376 000200 XMXT: RTS RO ;: EXIT SUBROUTINE.
3802 017400 000000 RECITT: .WORD 0 ;: RECEIVE ITERATION COUNT.
3803 017402 000000 .STOR: .WORD 0 ;: WORD STORAGE POINTER
3804 017404 000000 .TOR: .WORD 0 ;: BYTE STORAGE POINTER

```

```

3805
3806
3807      ;:*****
3808      ;SUBROUTINE TO XMIT THE BYTE AT TBUF.
3809      ;:*****
3810 017406 042737 000001 002262 XMIT1: BIC      #1,VSTAT      ;CLEAR XMIT DONE FLAG
3811 017414 012737 000001 016506      MOV      #1,XMCNT      ;SET UP TO XMIT 1 BYTE
3812 017422 052777 040000 162334      BIS      #XENA,@VJCSR
3813 017430
3814 017430 012746 000001      1$:      MOV      #XMDNE,-(SP)    ;;PUSH #XMDNE ON STACK
3815      ;:***** (REV. B0)
3816 017434 005737 002270      TST      PTYPE        ;PROCESSOR TYPE
3817 017440 001403      BEQ      2$           ;IF 0 THEN 11/45 OR 11/70
3818 017442 012746 000001      MOV      #1,-(SP)    ;;PUSH #1 ON STACK
3819 017446 000402      BR       3$           ;
3820 017450
3821 017450 012746 000002      2$:      MOV      #2,-(SP)    ;;PUSH #2 ON STACK
3822 017454 004037 022150      3$:      JSR      RO,WTBGND    ;LOOK FOR XMIT DONE
3823      ;:*****
3824 017460 000401      BR       FTLEXT      ;HUNG TRANSMIT-CLEAR FLAGS AND EXIT
3825 017462 000402      BR       NORXT       ;NORMAL EXIT.
3826 017464 005037 002262 FTLEXT: CLR      VSTAT    ;CLEAR ANY FLAGS
3827 017470 000200      NORXT:  RTS      RO    ;AND EXIT
3828
3829      ;:*****
3830      ;SUBROUTINE TO ISSUE A BYTE AT A TIME UNTIL A ZERO
3831      ;BYTE IS ENCOUNTERED.
3832      ;:*****
3833
3834 017472 112777 000002 177004 LDXM1T: MOVB     #SOM,@TBUF    ;SEND THE START OF MESSAGE.
3835 017500 000403      BR       2$
3836 017502 112377 176776      1$:      MOVB     (R3)+,@TBUF    ;MOVE A BYTE TO XMIT BUFFER
3837 017506 001403      BEQ      LDOUT        ;IF A ZERO BYTE-EXIT
3838 017510 004037 017406      2$:      JSR      RO,XMIT1     ;GO XMIT A BYTE
3839 017514 000772      BR       1$          ;XMIT AGAIN.
3840 017516 112777 000004 176760 LDOUT:  MOVB     #EOM,@TBUF    ;SEND THE END OF MESSAGE.
3841 017524 004037 017406      JSR      RO,XMIT1
3842 017530 000200      RTS      RO
3843
3844      ;:*****
3845      ;ROUTINE TO RESET ALL INTERRUPT POINTERS.
3846      ;:*****
3847
3848 017532 042777 040000 162224 RESPTR: BIC      #XENA,@VJCSR  ;CLEAR INTERRUPT ENABLES
3849 017540 013737 016172 016176      MOV      RBUF,RBUF    ;RESET RECEIVE BUF POINTER
3850 017546 013737 016500 016504      MOV      TBUF,TBUF    ;RESET XMIT BUF POINTER
3851 017554 012737 016202 016224      MOV      #STTER,STTER  ;RESET RECEIVE STATUS ERR POINTER
3852 017562 005037 016506      CLR      XMCNT        ;CLEAR TRANSMIT COUNT
3853 017566 005037 016200      CLR      ESAMB        ;CLEAR ESC ASSEMBLY FLAGS
3854 017572 012737 000001 017400      MOV      #1,RECITT    ;RESET REC. ITERATION COUNT
3855 017600 005037 017402      CLR      WDSTOR       ;CLEAR STORAGE POINTERS
3856 017604 005037 017404      CLR      BYSTOR
3857 017610 000200      RTS      RO
3858
3859
3860      ;:*****

```

```

3861 ;SUBROUTINE TO ISSUE CURSOR POSITION ERROR. GOOD
3862 ;LINE/COLUMN MUST BE A WORD ON STACK. ERROR
3863 ;POSITION IS EXPECTED TO BE @ RBBUF.
3864 ;:*****
3865
3866 017612 CURER:
3867 017612 012637 002260 MOV (SP)+,ROSVE ;:POP STACK INTO ROSVE
3868 017616 012637 002220 MOV (SP)+,CHRD ;:POP STACK INTO CHRD
3869 017622 162737 020040 002220 SUB #20040,CHRD ;:EXTRACT MOD 40 FROM GOOD POSITION
3870 017630 004037 017110 JSR RO,CLREG
3871 017634 113737 002221 001124 MOVB CHRD+1,$GDDAT ;:LOAD MESSAGE WITH GOOD
3872 017642 113737 002220 001120 MOVB CHRD,$GDADR ;:LINE AND COLUMN
3873 017650 017737 176316 002220 MOV @RBBUF,CHRD ;:LINE AND COLUMN.
3874 017656 162737 020040 002220 SUB #20040,CHRD ;:EXTRACT MOD 40 FROM BAD POSITION.
3875 017664 113737 002221 001126 MOVB CHRD+1,$BDDAT ;:LOAD MESSAGE WITH BAD
3876 017672 113737 002220 001122 MOVB CHRD,$BDADR ;:LINE AND COLUMN.
3877 017700 104006 ERROR 6 ;:ISSUE ERROR
3878 017702 013746 002260 MOV ROSVE,-(SP) ;:PUSH ROSVE ON STACK
3879 017706 000200 RTS RO
3880
3881 ;:*****
3882
3883 ;:*****
3884 ;SUBROUTINE TO DECREMENT CURSOR POSITION IN A
3885 ;LINEAR SEQUENCE. (IE. ROW 20, COL 1 ;ROW 20 COL 0 ;ROW 17, COL 157).
3886 ;:*****
3887
3888 017710 123727 020015 000040 CMPOS: CMPB LNRW+1,#40 ;:AT LEFT EDGE OF ROW?
3889 017716 001403 BEQ 1$ ;:YES, GO ADJUST COL., ROW.
3890 017720 105337 020015 DECB LNRW+1 ;:NO, DECREMENT COL. AND EXIT
3891 017724 000200 RTS RO
3892 017726 123727 020014 000040 1$: CMPB LNRW,#40 ;:AT ROW 0?
3893 017734 001405 BEQ 2$ ;:YES, NO DECREMENT POSSIBLE-EXIT.
3894 017736 105337 020014 DECB LNRW ;:NO, DECREMENT ROW AND
3895 017742 112737 000157 020015 MOVB #157,LNRW+1 ;:SET COL. TO RIGHT EDGE.
3896 017750 000200 2$: RTS RO
3897
3898 ;:*****
3899 ;SUBROUTINE TO INCREMENT CURSOR POSITION IN A LINEAR
3900 ;SEQUENCE (IE. ROW 10, COL 78, ROW 10. COL 79, ROW 11 , COL 0).
3901 ;:*****
3902
3903 017752 123727 020015 000157 CPPOS: CMPB LNRW+1,#157 ;:AT RIGHT EDGE OF ROW
3904 017760 001403 BEQ 1$ ;:YES, ADJUST ROW AND COLUMN.
3905 017762 105237 020015 INCB LNRW+1 ;:NO, INCREMENT COL. COUNT
3906 017766 000200 RTS RO ;:AND EXIT
3907 017770 123727 020014 000067 1$: CMPB LNRW,#67 ;:AT BOTTOM ROW?
3908 017776 001405 BEQ 2$ ;:YES, NO INCREMENT POSSIBLE-EXIT.
3909 020000 105237 020014 INCB LNRW ;:NO, INCREMENT ROW COUNT AND
3910 020004 112737 000040 020015 MOVB #40,LNRW+1 ;:SET COL. TO LEFT EDGE.
3911 020012 000200 2$: RTS RO
3912
3913 020014 000000 LNRW: .WORD 0 ;CONTAINS UPDATED CURSOR POSITION.
3914 ;:*****
3915
3916 ;SUBROUTINE TO XMIT, RECEIVE AND COMPARE. DATA ERRORS
    
```

3917 ;ARE REPORTED FROM SUBROUTINE. IF THE TRANSMIT OR
 3918 ;RECEIVE LOOPS 'TIME OUT', EXIT FROM SUBROUTINE WILL
 3919 ;BE NORMAL EXIT +2. SUBROUTINE ENTERED WITH (R1)=
 3920 ;GOOD DATA BUFFER, (R2)=RECEIVE DATA BUFFER AND
 3921 ;R3=COMPARE COUNT. IF THE VT61 DOES NOT HANG,THE ROUTINE
 3922 ;WILL WAIT FOR END OF REC. MESSAGE(EOM).
 3923
 3924
 3925
 3926

```

3927 020016 XRCMP: MOV R4,-(SP) ;;PUSH R4 ON STACK
3928 020016 010446 CLR R4 ;;USE R4 A RECEIVE COUNTER.
3929 020020 005004 MOV #1001,BLKM ;;SET UP FOR A SOM/EOM TRANSMIT.
3930 020022 012737 001001 002264 BIC #77577,VSTAT ;;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
3931 020030 042737 077577 002262 BIS #XENA,@VJCSR ;;SET INTERRUPT ENABLES.
3932 020036 052777 040000 161720 CLR HDFLG ;;CLEAR ERROR 13 PRINT FLAG
3933 020044 005037 020366 MOV #TCRLB+450,R5 ;;R5 IS ERROR STORAGE POINTER
3934 020050 012705 032642 1$: CLR DLAY ;;SET UP TIME OUT DELAY
3935 020054 005037 002256 BIT #XMDNE,VSTAT ;;XMIT DONE?
3936 020060 032737 000001 002262 BNE XREC ;;YES-GO RECEIVE
3937 020066 001014 2$: CMP RBBUF,RBBUF ;;HAS RECEIVE OPERATION BEGUN?
3938 020070 023737 016172 016176 BLO XREC ;;YES-GO RECEIVE
3939 020076 103410 BIT #RXOFF,VSTAT ;;XMIT XOFF SET?
3940 020100 032737 100000 002262 BEQ 1$ ;;NO-KEEP LOOKING FOR XMIT DONE?
3941 020106 001762 DEC DLAY ;;YES RUN DELAY AND LOOK
3942 020110 005337 002256 BNE 2$ ;;FOR XON OR RECEIVED CH.
3943 020114 001365 BR XRERR ;;TRANSMIT TIMEOUT-SET UP ERROR EXIT
3944 020116 000432
3945
3946 020120 005037 002256 XREC: CLR DLAY ;;SET UP TIME OUT DELAY
3947 020124 020237 016176 1$: CMP R2,RBBUF ;;INSURE COMPARE POINTER
3948 020130 103410 BLO 2$ ;;LESS THAN RECEIVE POINTER
3949 020132 032737 020000 002262 BIT #REOM,VSTAT ;;RECEIVE EOM?
3950 020140 001030 BNE XREXT ;;YES-SET UP TO EXIT
3951 020142 005337 002256 DEC DLAY ;;RUN TIMEOUT DELAY
3952 020146 001416 BEQ XRERR ;;TIME OUT OCCURRED-ERROR EXIT
3953 020150 000765 BR 1$ ;;RETURN TO CHECK RECEIVE COUNT
3954 020152 005204 2$: INC R4 ;;ADD 1 TO RECEIVE COUNTER.
3955 020154 122122 CMPB (R1)+,(R2)+ ;;COMPARE CHARACTERS
3956 020156 001407 BEQ 4$ ;;EQUAL-COMPARE AGAIN
3957 020160 020527 032672 CMP R5,#TCRLB+500 ;;ALLREADY STORED 50 ERRORS?
3958 020164 103004 BHIS 4$ ;;YES-BYPASS STORAGE
3959 020166 114125 MCVB -(R1),(R5)+ ;;STORE GOOD DATA
3960 020170 114225 MOVB -(R2),(R5)+ ;;STORE BAD DATA
3961 020172 010425 MOV R4,(R5)+ ;;LOAD RECEIVE COUNT
3962 020174 132122 BITB (R1)+,(R2)+ ;;RESET POINTERS AND
3963 020176 005303 4$: DEC R3 ;;CHECK COMPARE COUNT
3964 020200 001410 BEQ XREXT ;;ALL DONE-EXIT
3965 020202 000746 BR XRERR ;;COMPARE ANOTHER
3966 020204 062700 000002 XRERR: ADD #2,R0 ;;SET UP ERROR EXIT
3967 020210 005237 002236 INC FTLCNT ;;INCREMENT FATAL XMIT COUNT.
3968 020214 004037 017532 JSR R0,RESPTR ;;RESET INTERRUPT POINTERS.
3969 020220 000452 BR XROUT
3970 020222
3971 020222 012746 020000 XREXT: MOV #REOM,-(SP) ;;PUSH #REOM ON STACK
3972

```

```

3973 020226 005737 002270          TST      PTYPE          ;PROCESSOR TYPE
3974 020232 001407                   BEQ      3$             ;IF 0 THEN 11/45 OR 11/70
3975 020234 100403                   BMI      2$             ;IF -1 THEN 11/35 OR 11/40
3976 020236 012746 000002          MOV      #2,-(SP)      ;:PUSH #2 ON STACK
3977 020242 000405                   BR       4$             ;
3978 020244                   2$:
3979 020244 012746 000004          MOV      #4,-(SP)      ;:PUSH #4 ON STACK
3980 020250 000402                   BR       4$             ;
3981 020252                   3$:
3982 020252 012746 000010          MOV      #8,-(SP)      ;:PUSH #8. ON STACK
3983 020256 004037 022150          JSR      R0,WTBGND     ;
3984                                     ;*****
3985 020262 000431                   BR       XROUT         ;NO EOM-ISSUE ERROR AND EXIT.
3986 020264 162705 032642          SUB      #TCRLB+450,R5 ;NOW EXTRACT ERROR COUNT-IF ANY.
3987 020270 010501                   MOV      R5,R1         ;AND STORE IT IN R1
3988 020272 012705 032642          MOV      #TCRLB+450,R5 ;RELOAD ERROR POINTER
3989 020276 005701                   TST      R1            ;TEST FOR ERRORS
3990 020300 001422                   BEQ      XROUT         ;NO-CHECK STATUS AND EXIT
3991 020302 005737 020366          TST      HDFLG         ;:DATA ERROR HEADER PRINTED?
3992 020306 001003                   BNE      1$            ;YES-BYPASS HEADER PRINT
3993 020310 104012                   ERROR   12             ;PRINT DATA ERROR HEADER
3994 020312 005237 020366          INC      HDFLG         ;SET HEADER PRINT FLAG
3995 020316 004037 017110          1$: JSR      R0,CLREG     ;ERROR WAS LEGTIMATE. LOAD
3996 020322 112537 001124          MOV      (R5)+,$GDDAT ;ERROR MESSAGE AND ISSUE
3997 020326 112537 001126          MOV      (R5)+,$BDDAT ;IT.
3998 020332 012537 001120          MOV      (R5)+,$GDADR ;LOAD RECEIVE COUNT
3999 020336 104004                   ERROR   4              ;ISSUE DATA COMPARE ERROR
4000 020340 162701 000004          SUB      #4,R1         ;DECREMENT ERROR COUNT
4001 020344 001364                   BNE      1$            ;PRINT ANOTHER IF NOT AT ZERO
4002 020346 004037 022254          XROUT: JSR      R0,CKOFF ;SEE IS XOFF IS UP.
4003 020352 012746 060001          MOV      #60001,-(SP) ;:PUSH #60001 ON STACK
4004 020356 004037 016746          JSR      R0,CKSFT     ;CHECK FOR VSTAT /STATUS ERR.
4005 020362 012604                   MOV      (SP)+,R4      ;:POP STACK INTO R4
4006 020364 000200                   RTS      R0            ;EXIT SUBROUTINE
4007
4008 020366 000000          HDFLG: 0              ;INHIBIT PRINT FLAG.
4009
4010                                     ;:*****
4011
4012                                     ;SUBROUTINE TO CREATE A 'RULER' IN LOCATIONS 200
4013                                     ;TO 317.
4014
4015                                     ;:*****
4016
4017 020370 012701 032372          CRRUL: MOV      #TCRLB+200,R1 ;LOAD STARTING ADDRESS
4018 020374 012702 130461          MOV      #130461,R2   ;LOAD INITIAL RULER ASCII CODES.
4019 020400 110221                   1$: MOV      R2,(R1)+  ;STORE A RULER BYTE IN XMIT BUF.
4020 020402 022701 032512          CMP      #TCRLB+320,R1 ;RULER COMPLETE?
4021 020406 103001                   BHIS    2$            ;NO
4022 020410 000200                   RTS      R0            ;AND EXIT.
4023 020412 105202                   2$: INCB    R2         ;INCREMENT ASCII BYTE
4024 020414 122702 000272          CMPB    #272,R2       ;END OF REVERSE VIDEO?
4025 020420 001003                   BNE     3$            ;NO-SEE IF END OF NORMAL.
4026 020422 012702 030660          MOV      #030660,R2   ;SET UP TO ISSUE REVERSE 0.
4027 020426 000405                   BR       5$            ;
4028 020430 122702 000072          3$: CMPB    #72,R2     ;END OF NORMAL VIDEO?

```



```

4029 020434 001361          BNE      1$          ;NOT AT END OF A VIDEO STRING.
4030 020436 012702 130460  MOV      #130460,R2 ;YES-SET UP TO ISSUE NORMAL 0.
4031 020442 110221          5$: MOVB   R2,(R1)+    ;DO IT
4032 020444 105202          INCB   R2          ;SET BYTE TO NEXT ASCII CODE
4033 020446 000302          SWAB  R2          ;REVERSE VIDEO MODE.
4034 020450 000753          BR     1$          ;BEGIN NEXT STRING
4035
4036 ;:*****
4037 ;SUBROUTINE TO DELAY 10 M.S. TIME THE NUMBER INLOCATION
4038 ;DCOUNT. THE PROCESSOR TYPE PRE-DETERMINES THE # OF LOOPS
4039 ;REQUIRED TO DELAY 10 M.S. FOR ONE ITERATION. LOCATION
4040 ;PMULT IS PRE-LOADED WITH : 11/45 = 4, 11/40 = 2
4041 ;AND 11/10 =1.
4042 ;:*****
4043
4044 020452          DELAY:
4045 020452 010146          MOV     R1,-(SP)    ;;PUSH R1 ON STACK
4046 020454 010246          MOV     R2,-(SP)    ;;PUSH R2 ON STACK
4047 020456 013702 020512  1$: MOV     PMULT,R2   ;LOAD PROCESSOR MULTIPLIER
4048 020462 012701 002570  2$: MOV     #1400.,R1 ;LOAD 10 M.S. DELAY
4049 020466 005301          DEC     R1         ;RUN BASIC DELAY
4050 020470 001376          BNE    .-2
4051 020472 005302          DEC     R2         ;RUN MULTIPLIER DELAY
4052 020474 001372          BNE    2$
4053 020476 005337 020514  DEC     DCOUNT     ;RUN ITERATION COUNT
4054 020502 001365          BNE    1$
4055 020504 012602          MOV     (SP)+,R2   ;;POP STACK INTO R2
4056 020506 012601          MOV     (SP)+,R1   ;;POP STACK INTO R1
4057 020510 000200          RTS     R0
4058
4059 020512 000000          PMULT: 0          ;PROCESSOR MULTIPLIER
4060 020514 000000          DCOUNT: 0        ;ITERATION COUNT
4061
4062 ;:*****
4063
4064 ;SUBROUTINE TO GENERATE A INCREMENTING PATTERN AT
4065 ;(R1)+. ENTER WITH R3 EQUAL TO # OF CH. TO CREATE.
4066 ;R5 IS UTILIZED AS A WORK REGISTER.
4067
4068 ;:*****
4069
4070 020516 012705 000041  BLDINC: MOV     #41,R5 ;LOAD R5 WITH INITIAL CH.
4071 020522 110521          BLDINA: MOVB   R5,(R1)+ ;MOVE A CH. TO BUFFER
4072 020524 005303          DEC     R3         ;DECREMENT BYTE COUNT
4073 020526 001001          BNE    2$         ;NOT DONE-UPDATE PATTERN
4074 020530 000200          RTS     R0         ;EXIT-DONE.
4075 020532 105205          2$: INCB   R5         ;UPDATE CH. PATTERN
4076 020534 122705 000177  CMPB   #177,R5     ;PATTERN EXCEEDED MAX?
4077 020540 001766          BEQ    BLDINC     ;YES-RESET IT.
4078 020542 000767          BR     BLDINA     ;NO-ISSUE CURRENT PATTERN.
4079
4080
4081 ;:*****
4082
4083 ;SUBROUTINE TO FILL THE SCREEN WITH INCREMENTING DATA
4084 ;:*****

```

```

4085
4086 020544 042737 077577 002262 DATSC: BIC #77577,VSTAT ;CLEAR INTERRUPT FLAGS.
4087 020552 013701 016500 MOV TBBUF,R1
4088 020556 012703 000500 MOV #320.,R3 ;FILL XMIT BUFFER WITH INCRE-
4089 020562 004037 020516 JSR RO,BLDINC ;MENTING PATTERN
4090 020566 012737 003600 016506 10$: MOV #TOTCH,XMCNT ;SET UP TO XMIT 1920 BYTES
4091 020574 052777 040000 161162 BIS #XENA,@VJCSR
4092
4093 020602 032737 000001 002262 1$: BIT #XMDNE,VSTAT ;XMIT DONE?
4094 020610 001774 BEQ -6 ;NO
4095
4096 ;:*****
4097
4098 ;SUBROUTINE TO RESET VT61 AND DISPLAY MESSAGE
4099 ;POINTED TO BY R2.
4100
4101 ;:*****
4102
4103 020612 004037 016510 DSMES: JSR RO,RESETV ;RESET THE UNIT AND WAIT FOR XON.
4104 020616 042737 077577 002262 BIC #77577,VSTAT ;CLEAR ALL FLAGS EXCEPT XOFF AND XMKIL.
4105 020624 012737 000005 016506 MOV #5,XMCNT ;PRE-LOAD XMIT COUNT.
4106 020632 013701 016500 MOV TBBUF,R1 ;LOAD XMIT BUFFER WITH:
4107 020636 012721 000002 MOV #SOM,(R1)+ ;START OF MESSAGE
4108 020642 013721 002112 MOV ESCO,(R1)+
4109 020646 013721 002046 MOV DRECT,(R1)+ ;DISABLE RECTANGULAR MODE
4110 020652 005237 016506 1$: INC XMCNT ;INCREMENT TRANSMIT COUNT
4111 020656 112221 MOVB (R2)+,(R1)+ ;DISPLAY MESSAGE
4112 020660 001374 BNE 1$
4113 020662 112711 000004 MOVB #EOM,(R1) ;TERMINATE WITH END OF MESSAGE.
4114 020666 052777 040000 161070 BIS #XENA,@VJCSR ;XMIT IT AND WAIT FOR
4115 020674 032737 000001 002262 2$: BIT #XMDNE,VSTAT ;DONE
4116 020702 001774 BEQ 2$
4117 020704 000200 RTS RO
4118
4119 ;:*****
4120
4121 ;SUBROUTINE TO CONVERT A BINARY CHARACTER
4122 ;TO 3 OCTAL CHARACTERS. R1 CONTAINS BINARY
4123 ;NUMBER. RESULT IS STORED IN LOCATIONS SVER1,
4124 ;SVER2
4125
4126 ;:*****
4127 020706 BINOCT:
4128 020706 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
4129 020710 012705 000002 MOV #2,R5 ;LOAD ITERATION COUNT
4130 020714 000403 BR 2$ ;BYPASSS SHIFTS FOR 1ST CONVERSION
4131 020716 106201 1$: ASRB R1
4132 020720 106201 ASRB R1 ;SHIFT A CHAR INTO POSITION
4133 020722 106201 ASRB R1
4134 020724 110165 002222 2$: MOVEB R1,SVER1(R5) ;STORE THE BINARY OFFSET
4135 020730 142765 000370 002222 BICB #370,SVER1(R5) ;CLEAR NON ESSENTIAL BITS
4136 020736 152765 000060 002222 BISB #60,SVER1(R5) ;CONVERT OFFSET TO OCTAL
4137 020744 005305 DEC R5 ;DECREMENT CONVERSION COUNT
4138 020746 100363 BPL 1$ ;NOT DONE CONVERT ANOTHER
4139 020750 112737 000040 002225 MOVB #40,SVER2+1 ;LOAD A SPACE
4140 020756 012605 MOV (SP)+,R5 ;;POP STACK INTO R5

```

```

4141 020760 000200          RTS      RO
4142
4143 ;:*****
4144 ;SUBROUTINE TO CONVERT AN OCTAL CHAR. TO BINARY.  REG
4145 ;R1 CONTAINS OCTAL AND REG R2 IS BINARY ASSEMBLY AREA.
4146 ;:*****
4147
4148 020762 042701 177770 OCTBIN: BIC      #177770,R1      ;EXTRACT OCTAL COMPONENT
4149 020766 005702          TST      R2          ;FIRST CONVERSION?
4150 020770 001403          BEQ      NOSHFT       ;YES - DO NOT SHIFT
4151 020772 006302          ASL      R2          ;NO - SHIFT PREVIOUS CHAR.
4152 020774 006302          ASL      R2
4153 020776 006302          ASL      R2
4154 021000 060102          NOSHFT: ADD     R1,R2          ;ADD CURRENT CHAR.
4155 021002 000200          RTS      RO
4156 ;:*****
4157 ;SUBROUTINE TO CONVERT A BINARY POSITION TO A OCTAL #.
4158 ;:*****
4159 021004 005037 002000 CONVLN: CLR      OCTLNE
4160 021010 013737 001776 002250 MOV      TSTLNE,TPREG ;LOAD OCTLNE WITH OCTAL EQUIVALENT
4161 021016 032737 000001 002250 101$: BIT      #BIT00,TPREG ;OF TSTLNE IN BITS 8 THRU 11.
4162 021024 001005          BNE      102$
4163 021026 006037 002250 ROR      TPREG
4164 021032 105237 002001 .NCB    OCTLNE+1
4165 021036 000767          BR      101$
4166 021040 000200          102$: RTS      RO
4167
4168 ;:*****
4169 ;ROUTINE TO WAIT FOR C/R FROM VT61 UNDER TEST
4170 ;:*****
4171
4172 021042 032777 000200 160714 GTCR:  BIT      #RECDN,@VJLSR ;WAIT FOR REVEIVE DONE
4173 021050 001774          BEQ      .-6
4174 021052 127737 160710 002010 CMPB    @VRBUF,CARRT ;CHAR = CARRIAGE RETURN?
4175 021060 001370          BNE      GTCR      ;NO-KEEP LOOKING
4176 021062 000200          RTS      RO      ;YES-EXIT
4177
4178 ;:*****
4179 ;SUBROUTINE TO GET A CHARACTER (NUMERIC) FROM THE
4180 ;CONSOLE.  IF OTHER THAN A NUMERIC IS TYPED A
4181 ;"?" WILL BE ECHOED.
4182 ;:*****
4183
4184
4185 021064 004037 014726 GTNUM: JSR      RO,CONRD ;GET A CHAR
4186 021070 012601          MOV      (SP)+,R1 ;POP STACK INTO R1
4187 021072 122701 000054 CMPB    #54,R1      ;CHAR. =COMMA?
4188 021076 001411          BEQ      1$        ;YES-GO PRINT IT
4189 021100 123701 002010 CMPB    CARRT,R1    ;CHAR. = CARRIAGE RETURN?
4190 021104 001406          BEQ      1$
4191 021106 120127 000060 CMPB    R1,#60
4192 021112 103421          BLO     QUST       ;IF CHAR. IS LESS THAN 60
4193 021114 120127 000067 CMPB    R1,#67     ;OR MORE THAN 67, TYPE
4194 021120 101016          BHI     QUST       ;A QUESTION MARK
4195 021122 110137 021172 1$:  MOVB    R1,TYPNUM
4196 021126 104401 021172 TYPE    ,TYPNUM
  
```

```

4197 021132 123701 002012          CMPB   LNFED,R1
4198 021136 001406          BEQ    GTEXT
4199 021140 123701 002010          CMPB   CARRT,R1          ;IF CHAR. - C/R SET UP TO ISSUE
4200 021144 001003          BNE    GTEXT            ;LINE FEED BEFORE EXITING.
4201 021146 113701 002012          MOVB   LNFED,R1
4202 021152 000763          BR     1$
4203 021154 000200          GTEXT: RTS              ;GOOD CHAR., EXIT
4204 021156 112737 000077 021172 QUST:  MOVB   #77,TYPNUM
4205 021164 104401 021172          TYPE  ,TYPNUM          ;TYPE QUESTION MARK AND
4206 021170 000735          BR     GTNUM           ;KEEP LOOKING.
4207 021172 000          TYPNUM: .BYTE 0
4208 021173 000          .BYTE 0
4209
4210
4211 ;:*****
4212 ;SUBROUTINE TO CALCULATE CHECKSUM ON THE LOWER
4213 ;BYTE OF R5. R4 IS STORAGE FOR THE CHECKSUM
4214 ;CHARACTER. ALGORITHM FOR CHECKSUM IS ROTATE
4215 ;CURRENT ONE PLACE LEFT AND XOR NEW CHAR. CHECKSUM
4216 ;IS THE LOWER 7 BITS OF R4
4217
4218 ;:*****
4219
4220 021174 042705 177400          CALCK: BIC    #177400,R5      ;CLEAR UPPER BYTE OF R5
4221 021200 120527 000021          CMPB   R5,#XON            ;CHAR. =XON?
4222 021204 001415          BEQ    NOCALC            ;YES DO NOT CALCULATE CHECKSUM
4223 021206 120527 000023          CMPB   R5,#XOFF          ;CHAR =XOFF?
4224 021212 001412          BEQ    NOCALC            ;YES DO NOT CALCULATE CHECKSUM
4225
4226 021214 000241          CLC                      ;INSURE CARRY BIT INITIALLY CLEAR
4227 021216 105704          TSTB   R4                ;SET UP TO ROTATE R4
4228 021220 100001          BPL    1$                ;A FULL 8 BYTES
4229
4230 021222 000261          SEC                      ;R4 WAS NEG. SO ROTATE A ONE
4231 021224 106104          1$:  ROLB   R4              ;INTO LOW ORDER BIT.
4232 021226 010403          MOV    R4,R3
4233 021230 040503          BIC    R5,R3              ;NOT A AND B
4234 021232 040405          BIC    R4,R5              ;NOT B AND A
4235 021234 050305          BIS    R3,R5              ;ORED
4236 021236 010504          MOV    R5,R4              ;EQUAL NEW CHECKSUM
4237 021240 000200          NOCALC: RTS              R0
4238 ;:*****
4239
4240 ;SUBROUTINE TO LOAD XMIT BUFFER FROM R0 THRU R1
4241 ;:*****
4242
4243 021242 112021          LDBUF: MOVB   (R0)+,(R1)+    ;LOAD A BYTE
4244 021244 001376          BNE    .-2                ; UNTIL ZERO BYTE FOUND.
4245 021246 000200          RTS    R0
4246 ;:*****
4247 ;SUBROUTINE TO CHECK THE VT61 FOR A PERIPHERAL ABORT.
4248 ;:*****
4249
4250 021250 032737 010000 002262 CKABRT: BIT    #PABRT,VSTAT    ;ABORT FLAG RECEIVED?
4251 021256 001446          BEQ    2$                ;NO-EXIT
4252 021260 010037 001124          MOV    R0,$GDDAT

```

```

4253 021264 162737 000004 001124      SUB      #4,$GDDAT      ;POINT ERR PC TO MAIN ROUTINE.
4254 021272 013737 002262 001126      MOV      VSTAT,$BDDAT
4255 021300 104020      ERROR    20           ;ISSUE PERIPHERAL ABORT ERROR
4256
4257 021302 013701 016500      MOV      TBBUF,R1
4258 021306 004037 021242      JSR      RO,LDBUF      ;LOAD THE XMIT BUFFER WITH:
4259 021312      033      117      137      .BYTE    .ESC,.O,.IABT,.ESC,.O,.RABT
4260 021315      033      117      140
4261 021320      033      117      145
4262 021323      007      000      000      .BYTE    .ESC,.O,.UNLKKB,.BEL,0,0
4263 021326 012737 000007 016506      MOV      #7,XMCNT      ;SET UP TO XMIT 7 BYTES.
4264 021334 004037 017132      JSR      RO,XMREC      ;XMIT AND RECEIVE.
4265 021340 000240      NOP
4266 021342 123727 016226 000170      CMPB     STRO,#NABRT    ;ABORT FLAG CLEARED?
4267 021350 001411      BEQ      2$           ;YES-EXIT
4268 021352 010037 001124      MOV      RO,$GDDAT      ;NO-SET UP AND ISSUE A CANT
4269 021356 162737 000004 001124      SUB      #4,$GDDAT      ;CLEAR ABORT FLAG ERROR MESSAGE.
4270 021364 013737 002262 001126      MOV      VSTAT,$BDDAT
4271 021372 104021      ERROR    21
4272 021374 000200      2$:      RTS      RO
4273
4274      ;:*****
4275      ;SUBROUTINE TO COMPARE RECEIVED KEYBOARD DATA WITH
4276      ;DATA EXPECTED. ERRORS ARE REPORTED AS POSITIONAL
4277      ;ERROR ONLY.
4278      ;:*****
4279
4280
4281 021376 105077 011270      CKKBD:  CLR      @ABUFP      ;CLEAR RECEIVE BYTE
4282 021402 005037 002220      CLR      CHR      ;CLEAR INPUT STORAGE.
4283 021406 105777 011260      KBDLP:  TSTB     @ABUFP      ;WAIT FOR A INPUT.
4284 021412 001775      BEQ
4285
4286 021414 117737 011252 002220      MOV      @ABUFP,CHR      ;STORE IT AND
4287 021422 105077 011244      CLR      @ABUFP      ;CLEAR THE INPUT AREA.
4288 021426 123714 002220      1$:      CMPB     CHR,(R4)      ;RECEIVED EQUAL EXPECTED?
4289 021432 001500      BEQ      GDSTRK      ;NO-UPDATE POINTERS.
4290 021434 005237 002246      INC      BUBCT      ;INCREMENT ERROR COUNT.
4291 021440 023727 002246 000012      CMP      BUBCT,#10.    ;COUNT = 10?
4292 021446 103075      BHIS     CNTF      ;YES-EXIT SUBROUTINE.
4293 021450 010401      MOV      R4,R1
4294 021452 166501 012276      SUB      DTIBL(R5),R1  ;EXTRACT KEY POSITION FROM ROW LOC.
4295 021456 005201      INC      R1           ;CONVERT LOGICAL POS. TO ACTUAL.
4296 021460 004037 020706      JSR      RO,BINOC      ;GET KEY POSITION IN OCTAL.
4297 021464 113737 002224 002222      MOV      SVER2,SVER1   ;RE-ASSEMBLE OCTAL BYTES.
4298 021472 123727 002223 000060      CMPB     SVER1+1,#60   ;POSITION LESS THAN 8?
4299 021500 001413      BEQ      LDPOS      ;YES-GO LOAD IT.
4300 021502 123727 002222 000062      CMPB     SVER1,#62     ;POSITION GREATER THAN 8 AND LESS THAN 12?
4301 021510 103404      BLO      BOROW      ;YES-SET UP TO BORROW.
4302 021512 162737 000002 002222      SUB      #2,SVER1      ;NO-JUST SUBTRACT 2.
4303 021520 000403      BR      LDPOS
4304 021522 162737 000370 002222      BOROW:  SUB      #370,SVER1 ;SUBTRACT AND BORROW.
4305 021530 113737 002222 031303      LDPOS:  MOV      SVER1,KYSTRK+1 ;LOAD THE CONVERTED DECIMAL #.
4306 021536 113737 002223 031302      MOV      SVER1+1,KYSTRK
4307 021544 012703 031225      DMPOCT: MOV      #DKBERR,R3
4308 021550 004037 017472      JSR      RO,LDXMIT ;ISSUE BODY OF KEYBOARD ERROR.

```

```

4309 021554 111401          MOVB   (R4)   ,R1
4310 021556 004037 020706   JSR    R0    ,BINOCT
4311 021562 012703 002222   MOV    #SVER1 ,R3
4312 021566 004037 017472   JSR    R0    ,LDXMIT ;CONVERT AND ISSUE GOOD CHAR.
4313 021572 012703 031334   MOV    #DSPC6,R3
4314 021576 004037 017472   JSR    R0,LDXMIT ;INSERT 6 SPACES IN MESSAGE.
4315 021602 113701 002220   MOVB   CHR0   ,R1
4316 021606 004037 020706   JSR    R0    ,BINOCT
4317 021612 012703 002222   MOV    #SVER1 ,R3
4318 021616 004037 017472   JSR    R0    ,LDXMIT ;CONVERT AND ISSUE RECEIVED CHAR.
4319 021622 012703 001171   MOV    #SCLF  ,R3
4320 021626 004037 017472   JSR    R0    ,LDXMIT ;ISSUE C/R AND L/F.
4321 021632 000665          BR     KBDLP ;LOOK FOR SAME KEY AGAIN.
4322
4323 021634 005204          GDSTRK: INC   R4           ;INCREMENT KEYBOARD ROW COUNTER.
4324 021636 105714          TSTB   (R4)           ;REACHED END OF ROW?
4325 021640 001262          BNE    KBDLP         ;NO-LOOK FOR NEXT INPUT
4326 021642 000200          CNTF:  RTS   R0       ;YES-EXIT.
4327
4328 ;:*****
4329
4330 ;SUBROUTINE TO LOOP DATA THROUGH HOST COMPUTER. ALL
4331 ;FUNCTIONS ARE ALLOWED, BUT BLOCK TRANSMITS WHICH

```

```
4332 ;EXCEED 552 BYTES WILL RESULT IN THE TERMINATION
4333 ;OF THE OPERATION AFTER 552 RECEIVED BYTES.
4334
4335 .;*****
4336
4337 021644 005237 022146 LOOP: INC XMZER ;SET UP TO XMIT NULLS.
4338 021650 012737 032672 016174 MOV #TCRLB+500,REBUF ;RESET BUFFER POINTERS
4339 021656 012737 031472 016500 MOV #RCRLB,TBBUF
4340 021664 004037 017532 JSR RO,RESPTR ;RELOAD ALL INTERRUPT POINTERS
4341 021670 042737 077577 002262 BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
4342 021676 013704 016176 LOOPT: MOV RBUF,R4 ;SET UP RECEIVE FLAG
4343 021702 032737 000001 002262 LOOPTA: BIT #XMDNE,VSTAT ;XMIT COMPLETE?
4344 021710 001407 BEQ LOOPR ;NO
```

| | | | | | | | | |
|------|--------|--------|--------|--------|---------|------|----------------|-------------------------------------|
| 4345 | 021712 | 042737 | 000001 | 002262 | | BIC | #XMDNE,VSTAT | :YES RESET FLAG |
| 4346 | 021720 | 013737 | 016172 | 016176 | | MOV | RBBUF,RBUFP | :RESET THE REC. BUFFER POINTER |
| 4347 | 021726 | 000763 | | | | BR | LOOP | |
| 4348 | 021730 | 032737 | 001400 | 002262 | LOOPR: | BIT | #EPL+ESC,VSTAT | :RECEIVED AN ESC OR EPL? |
| 4349 | 021736 | 001004 | | | | BNE | LPSTR | :YES-GO CHECK IT |
| 4350 | 021740 | 023704 | 016176 | | | CMP | RBUFP,R4 | :RECEIVED A DISPLAY CHAR? |
| 4351 | 021744 | 001756 | | | | BEQ | LOOP | :NO-LOOP |
| 4352 | 021746 | 000426 | | | | BR | BUMPCT | |
| 4353 | 021750 | 117777 | 010716 | 174220 | LPSTR: | MOVB | @ABUFP,@RBUFP | :YES LOAD IT IN THE BUFFER |
| 4354 | 021756 | 005237 | 016176 | | | INC | RBUFP | :AND INCREMENT BUFFER POINTER |
| 4355 | 021762 | 005037 | 016200 | | | CLR | ESAMB | :CLEAR ESC ASSEMBLY WORD |
| 4356 | 021766 | 042737 | 001400 | 002262 | | BIC | #EPL+ESC,VSTAT | :CLEAR THE FLAGS |
| 4357 | 021774 | 005237 | 016506 | | | INC | XMCNT | :INCREMENT XMIT COUNT |
| 4358 | 022000 | 123777 | 002170 | 010664 | | CMPB | ESCN,@ABUFP | :CHAR. A ESC(033)? |
| 4359 | 022006 | 001733 | | | | BEQ | LOOP | :YES WAIT FOR NEXT PART OF FUNCTION |
| 4360 | 022010 | 113777 | 002012 | 174160 | | MOVB | LNFEED,@RBUFP | :CHAR. WAS EPL ADD A LINE FEED. |
| 4361 | 022016 | 005237 | 016176 | | | INC | RBUFP | |
| 4362 | 022022 | 000407 | | | | BR | FRCECT | :AND ISSUE THEM. |
| 4363 | 022024 | 023727 | 016506 | 000764 | BUMPCT: | CMP | XMCNT,#500. | :BUFFER ABOUT FILLED? |
| 4364 | 022032 | 103403 | | | | BLO | FRCECT | :NO |
| 4365 | 022034 | 005337 | 016176 | | 1\$: | DEC | RBUFP | :YES-RESET THE RECEIVE POINTER |


```

4366 022040 000716          BR      LOOPT
4367 022042 005237 016506  FRCECT: INC    XMCNT          ;INCREMENT THE XMIT COUNT
4368 022046 023727 016506 000002  CMP    XMCNT,#2      ;FIRST CHAR TO XMIT?
4369 022054 101003          BHI    XMWT          ;NO
4370 022056 052777 040000 157700  BIS    #XENA,@VJCSR  ;YES-SET THE XMIT ENABLE
4371 022064 004037 022074  XMWT: JSR    RO,EXTST ;LOOK FOR END OF TEST COMMAND.
4372 022070 000702          BR     LOOPT        ;NONE FOUND.
4373 022072 000200          RTS    RO          ;AND EXIT
4374
4375
4376          ;:*****
4377          ;SUBROUTINE TO CHECK FOR END OF TEST COMMAND. THE CONTROL
4378          ;C KEY EXITS ALL TESTS.
4379          ;:*****
4380 022074 127727 010572 000003  EXTST: CMPB   @ABUFP,#3 ;LOOK FOR CONTROL C.
4381 022102 001020          BNE   NOROUT
4382
4383 022104 012737 032171 016174  ABSXT: MOV    #RCRLB+477,REBUF ;RESET THE BUFFERS
4384 022112 012737 032172 016500  MOV    #TCRLB,TBBUF
4385 022120 004037 017532          JSR    RO,RESPTR   ;RESET ALL POINTERS
4386 022124 012702 030113          MOV    #DEXT,R2
4387 022130 004037 020612          JSR    RO,DSMES    ;ISSUE EXIT MESSAGE
4388 022134 005037 022146          CLR    XMZER       ;CLEAR THE ZERO TRANSMIT FLAG.
4389 022140 062700 000002          ADD    #2,RO      ;SET UP TEST EXIT.
4390 022144 000200  NOROUT: RTS    RO          ;EXIT SUBROUTINE.
4391
4392 022146 000000  XMZER: .WORD  0
4393          ;:*****
4394          ;SUB-ROUTINE TO LOOK FOR VSTAT BIT ON THE STACK
4395          ;DELAY FACTOR IS FIRST WORD ON THE STACK AND VSTAT BIT
4396          ;IS THE SECOND. MIN. DELAY IS 4 U.S FOR A MOS 11/45.
4397          ;:*****
4398
4399          WTBGND:
4400 022150 012637 002260          MOV    (SP)+,ROSVE  ;:POP STACK INTO ROSVE
4401 022154 012637 022252          MOV    (SP)+,VDLAY ;:POP STACK INTO VDLAY
4402 022160 012637 022250          MOV    (SP)+,VBIT  ;:POP STACK INTO VBIT
4403 022164 012737 000002 002272  MOV    #2,DLYCNT   ;DO DELAY TWICE (REV. B0)
4404 022172 005037 002256          CLR    DLAY
4405 022176 033737 022250 002262  1$: BIT    VBIT,VSTAT ;SENSED THE CONDITION?
4406 022204 001015          BNE   FNDBT       ;YES-EXIT.
4407 022206 005337 002256          DEC    DLAY       ;NO-RUN DELAY.
4408 022212 001371          BNE   ?$
4409 022214 005337 002272          DEC    DLYCNT     ;LOOPED TWICE? (REV. B0)
4410 022220 001364          BNE   1$          ;IF NO, DO IT (REV. B0)
4411 022222 005337 022252          DEC    VDLAY     ;DELAY FACTOR EXPIRED?
4412 022226 001361          BNE   1$          ;NO-LOOP
4413 022230 104011          ERROR  11        ;DELAY EXPIRED-ISSUE HUNG NIT
4414 022232 005237 002236          INC    FTLCNT    ;INCREMENT FATAL XMIT COUNT.
4415 022236 000401          BR     TIMEXT
4416 022240 005720  FNDBT: TST    (RO)+ ;SET UP FOR NORMAL EXIT
4417 022242          TIMEXT:
4418 022242 013746 002260          MOV    ROSVF,-(SP) ;:PUSH ROSVE ON STACK
4419 022246 000200          RTS    RO
4420 022250 000000  VBIT:  0
4421 022252 000000  VDLAY:  0
  
```

```

4422 ;:*****
4423 ;:SUBROUTINE TO LOOK FOR XOFF BEFORE EXITING A RECEIVE ROUTINE.
4424 ;:*****
4425
4426 022254 005037 002256 CKOFF: CLR DLAY
4427 022260 032737 100000 002262 1$: BIT #RXOFF,VSTAT ;IS XOFF SET?
4428 022266 001403 BEQ 2$ ;NO-EXIT
4429 022270 005337 002256 DEC DLAY ;RUN DELAY.
4430 022274 001371 BNE 1$
4431 022276 000200 2$: RTS RO
4432
4433 .SBTTL SCOPE HANDLER ROUTINE
4434
4435 ;:*****
4436 ;:THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
4437 ;:AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
4438 ;:AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
4439 ;:THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4440 ;:SW14=1 LOOP ON TEST
4441 ;:SW11=1 INHIBIT ITERATIONS
4442 ;:SW09=1 LOOP ON ERROR
4443 ;:SW08=1 LOOP ON TEST IN SWR<7:0>
4444 ;:CALL
4445 ;* SCOPE ;:SCOPE=IOT
4446
4447 022300 $SCOPE:
4448 022300 004037 014760 JSR RO,MONIT
4449 022304 032777 040000 156626 1$: BIT #BIT14,@SWR ;:LOOP ON PRESENT TEST?
4450 022312 001111 BNE $OVER ;:YES IF SW14=1
4451 ;:*****START OF CODE FOR THE XOR TESTER*****
4452 022314 000416 $XTSTR: BR 6$ ;:IF RUNNING ON THE 'XOR' TESTER CHANGE
4453 ;:THIS INSTRUCTION TO A 'NOP' (NOP=240)
4454 022316 013746 000004 MOV @#ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
4455 022322 012737 022342 000004 MOV #5$,@#ERRVEC ;:SET FOR TIMEOUT
4456 022330 005737 177060 TST @#177060 ;:TIME OUT ON XOR?
4457 022334 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
4458 022340 000463 BR $SVLAD ;:GO TO THE NEXT TEST
4459 022342 022626 5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
4460 022344 012637 000004 MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
4461 022350 000423 BR 7$ ;:LOOP ON THE PRESENT TEST
4462 022352 6$: ;:*****END OF CODE FOR THE XOR TESTER*****
4463 022352 032777 000400 156560 BIT #BIT08,@SWR ;:LOOP ON SPEC. TEST?
4464 022360 001404 BEQ 2$ ;:BR IF NO
4465 022362 127737 156552 001102 CMPB @SWR,$STNM ;:ON THE RIGHT TEST? SWR<7:0>
4466 022370 001462 BEQ $OVER ;:BR IF YES
4467 022372 105737 001103 2$: TSTB $ERFLG ;:HAS AN ERROR OCCURRED?
4468 022376 001421 BEQ 3$ ;:BR IF NO
4469 022400 123737 001115 001103 CMPB $ERMAX,$ERFLG ;:MAX. ERRORS FOR THIS TEST OCCURRED?
4470 022406 101015 BHI 3$ ;:BR IF NO
4471 022410 032777 001000 156522 BIT #BIT09,@SWR ;:LOOP ON ERROR?
4472 022416 001404 BEQ 4$ ;:BR IF NO
4473 022420 013737 001110 001106 7$: MOV $LPERR,$LPADR ;:SET LOCP ADDRESS TO LAST SCOPE
4474 022426 000443 BR $OVER
4475 022430 105037 001103 4$: CLRB $ERFLG ;:ZERO THE ERROR FLAG
4476 022434 005037 001160 CLR $TIMES ;:CLEAR THE NUMBER OF ITERATIONS TO MAKE
4477 022440 000415 BR 1$ ;:ESCAPE TO THE NEXT TEST
  
```

```

4478 022442 032777 004000 156470 3$: BIT #BIT11,@SWR ;;INHIBIT ITERATIONS?
4479 022450 001011 BNE 1$ ;;BR IF YES
4480 022452 005737 001100 TST $PASS ;;IF FIRST PASS OF PROGRAM
4481 022456 001406 BEQ 1$ ;; INHIBIT ITERATIONS
4482 022460 005237 001104 INC $ICNT ;;INCREMENT ITERATION COUNT
4483 022464 023737 001160 001104 CMP $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
4484 022472 002021 BGE $OVER ;;BR IF MORE ITERATION REQUIRED
4485 022474 012737 000001 001104 1$: MOV #1,$ICNT ;;REINITIALIZE THE ITERATION COUNTER
4486 022502 013737 022552 001160 MOV $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
4487 022510 105237 001102 $SVLAD: INCB $STNM ;;COUNT TEST NUMBERS
4488 022514 011637 001106 MOV (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
4489 022520 011637 001110 MOV (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
4490 022524 005037 001162 CLR $ESCAPE ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
4491 022530 112737 000001 001115 MOVB #1,$ERMAX ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
4492 022536 013777 001102 156376 $OVER: MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER
4493 022544 013716 001106 MOV $LPADR,(SP) ;;FUDGE RETURN ADDRESS
4494 022550 000002 RTI ;;FIXES PS
4495 022552 000005 $MXCNT: 5 ;;MAX. NUMBER OF ITERATIONS
4496 .SBTTL ERROR HANDLER ROUTINE
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510 022554 $ERROR:
4511 022554 105237 001103 7$: INCB $ERFLG ;;SET THE ERROR FLAG
4512 022560 001775 BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
4513 022562 013777 001102 156352 MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
4514 022570 032777 002000 156342 BIT #BIT10,@SWR ;;BELL ON ERROR?
4515 022576 001402 BEQ 1$ ;;NO - SKIP
4516 022600 104401 001164 TYPE , $BELL ;;RING BELL
4517 022604 005237 001112 1$: INC $ERTTL ;;COUNT THE NUMBER OF ERRORS
4518 022610 011637 001116 MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
4519 022614 162737 000002 001116 SUB #2,$ERRPC
4520 022622 117737 156270 001114 MOVB @$ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
4521 022630 032777 020000 156302 BIT #BIT13,@SWR ;;SKIP TYPEOUT IF SET
4522 022636 001004 BNE 20$ ;;SKIP TYPEOUTS
4523 022640 004737 023142 JSR PC,$ERRTYP ;;GO TO USER ERROR ROUTINE
4524 022644 104401 001171 TYPE , $CRLF
4525 022650 20$:
4526 022650 005777 156264 2$: TST @SWR ;;HALT ON ERROR
4527 022654 100001 BPL 3$ ;;SKIP IF CONTINUE
4528 022656 000000 HALT ;;HALT ON ERROR!
4529 022660 032777 001000 156252 3$: BIT #BIT09,@SWR ;;LOOP ON ERROR SWITCH SET?
4530 022666 001402 BEQ 4$ ;;BR IF NO
4531 022670 013716 001110 MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
4532 022674 005737 001162 4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
4533 022700 001402 BEQ 5$ ;;BR IF NONE

```

```

*****
*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
*AND GO TO $ERRTYP ON ERROR
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
*SW15=1 HALT ON ERROR
*SW13=1 INHIBIT ERROR TYPEOUTS
*SW10=1 BELL ON ERROR
*SW09=1 LOOP ON ERROR
*CALL
* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER

```

```

4534 022702 013716 001162          MOV    $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
4535 022706                                5$:
4536 022706 022737 011246 000042    CMP    #SENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
4537 022714 001001                                BNE    6$              ;;BRANCH IF NO
4538 022716 000000                                HALT                                ;;YES
4539 022720                                6$:
4540 022720 000002          RTI                    ;;RETURN
4541                                .SBTTL  TYPE ROUTINE
4542
4543                                ;*****
4544                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
4545                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
4546                                ;*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
4547                                ;*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
4548                                ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
4549                                ;*
4550                                ;*CALL:
4551                                ;*1) USING A TRAP INSTRUCTION
4552                                ;*      TYPE      ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
4553                                ;*OR
4554                                ;*      TYPE
4555                                ;*      MESADR
4556                                ;*
4557
4558 022722 105737 001157    $TYPE: TSTB    $STPFLG          ;;IS THERE A TERMINAL?
4559 022726 100002          BPL     1$                  ;;BR IF YES
4560 022730 000000          HALT                        ;;HALT HERE IF NO TERMINAL
4561 022732 000407          BR     3$                  ;;LEAVE
4562 022734 010046    1$:  MOV     RO,-(SP)          ;;SAVE RO
4563 022736 017600 000002    MOV     @2(SP),RO          ;;GET ADDRESS OF ASCIZ STRING
4564 022742 112046    2$:  MOVVB  (RO)+,-(SP)        ;;PUSH CHARACTER TO BE TYPED ONTO STACK
4565 022744 001005          BNE    4$                  ;;BR IF IT ISN'T THE TERMINATOR
4566 022746 005726          TST   (SP)+               ;;IF TERMINATOR POP IT OFF THE STACK
4567 022750 012600    60$: MOV     (SP)+,RO          ;;RESTORE RO
4568 022752 062716 000002    3$:  ADD     #2,(SP)           ;;ADJUST RETURN PC
4569 022756 000002          RTI                        ;;RETURN
4570 022760 122716 000011    4$:  CMPB   #HT,(SP)           ;;BRANCH IF <HT>
4571 022764 001430          BEQ   8$                  ;;BRANCH IF NOT <CRLF>
4572 022766 122716 000200    CMPB   #CRLF,(SP)
4573 022772 001006          BNE   5$
4574 022774 005726          TST   (SP)+               ;;POP <CR><LF> EQUIV
4575 022776 104401          TYPE                                ;;TYPE A CR AND LF
4576 023000 001171          $CRLF
4577 023002 105037 023136    CLRB   $CHARCNT          ;;CLEAR CHARACTER COUNT
4578 023006 000755          BR    2$                  ;;GET NEXT CHARACTER
4579 023010 004737 023072    5$:  JSR    PC,$TYPEC          ;;GO TYPE THIS CHARACTER
4580 023014 123726 001156    6$:  CMPB   $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
4581 023020 001350          BNE   2$                  ;;IF NO GO GET NEXT CHAR.
4582 023022 013746 001154    MOV    $NULL,-(SP)        ;;GET # OF FILLER CHARS. NEEDED
4583                                ;;AND THE NULL CHAR.
4584 023026 105366 000001    7$:  DECB   1(SP)              ;;DOES A NULL NEED TO BE TYPED?
4585 023032 002770          BLT   6$                  ;;BR IF NO--GO POP THE NULL OFF OF STACK
4586 023034 004737 023072    JSR    PC,$TYPEC          ;;GO TYPE A NULL
4587 023040 105337 023136    DECB   $CHARCNT          ;;DO NOT COUNT AS A COUNT
4588 023044 000770          BR    7$                  ;;LOOP
4589

```

```

4590 ;HORIZONTAL TAB PROCESSOR
4591
4592 023046 112716 000040 8$: MOVB #' (SP) ;;REPLACE TAB WITH SPACE
4593 023052 004737 023072 9$: JSR PC,$TYPEC ;;TYPE A SPACE
4594 023056 132737 000007 023136 BITB #7,$CHARCNT ;;BRANCH IF NOT AT
4595 023064 001372 BNE 9$ ;;TAB STOP
4596 023066 005726 TST (SP)+ ;;POP SPACE OFF STACK
4597 023070 000724 BR 2$ ;;GET NEXT CHARACTER
4598 023072 105777 156052 $TYPEC: TSTB @STPS ;;WAIT UNTIL PRINTER IS READY
4599 023076 100375 BPL $TYPEC
4600 023100 116677 000002 156044 MOVB 2(SP),@STPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
4601 023106 122766 000015 000002 CMPB #CR,2(SP) ;;IS CHARACTER A CARRIAGE RETURN?
4602 023114 001003 BNE 1$ ;;BRANCH IF NO
4603 023116 105037 023136 CLR$ $CHARCNT ;;YES--CLEAR CHARACTER COUNT
4604 023122 000406 BR $TYPEX ;;EXIT
4605 023124 122766 000012 000002 1$: CMPB #LF,2(SP) ;;IS CHARACTER A LINE FEED?
4606 023132 001402 BEQ $TYPEX ;;BRANCH IF YES
4607 023134 105227 INCB (PC)+ ;;COUNT THE CHARACTER
4608 023136 000000 $CHARCNT: .WORD 0 ;;CHARACTER COUNT STORAGE
4609 023140 000207 $TYPEX: RTS PC
  
```

4610 .SBTTL ERROR MESSAGE TIMEOUT ROUTINE

```

4611
4612
4613 ;*****
4614 ;*THIS ROUTINE USES THE 'ITEM CONTROL BYTE' ($ITEMB) TO DETERMINE WHICH
4615 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE 'ERROR TABLE' ($ERRTB),
4616 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
4617
  
```

```

4618 023142 $ERRTYP:
4619 023142 104401 001171 TYPE $CRLF ;;'CARRIAGE RETURN' & 'LINE FEED'
4620 023146 010040 MOV R0,-(SP) ;;SAVE R0
4621 023150 005000 CLR R0 ;;PICKUP THE ITEM INDEX
4622 023152 153700 001114 BISB @#$ITEMB,R0
4623 023156 001004 BNE 1$ ;;IF ITEM NUMBER IS ZERO, JUST
4624 ;;TYPE THE PC OF THE ERROR
4625 023160 013746 001116 MOV $ERRPC,-(SP) ;;SAVE $ERRPC FOR TIMEOUT
4626 ;;ERROR ADDRESS
4627 023164 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4628 023166 000445 BR 10$ ;;GET OUT
4629 023170 005300 1$: DEC R0 ;;ADJUST THE INDEX SO THAT IT WILL
4630 023172 006300 ASL R0 ;; WORK FOR THE ERROR TABLE
4631 023174 006300 ASL R0
4632 023176 006300 ASL R0
4633 023200 062700 001174 ADD # $ERRTB,R0 ;;FORM TABLE POINTER
4634 023204 012037 023214 MOV (R0)+,2$ ;;PICKUP 'ERROR MESSAGE' POINTER
4635 023210 001404 BEQ 3$ ;;SKIP TIMEOUT IF NO POINTER
4636 023212 104401 TYPE ;;TYPE THE 'ERROR MESSAGE'
4637 023214 000000 2$: .WORD 0 ;;'ERROR MESSAGE' POINTER GOES HERE
4638 023216 104401 001171 TYPE $CRLF ;;'CARRIAGE RETURN' & 'LINE FEED'
4639 023222 012037 023232 3$: MOV (R0)+,4$ ;;PICKUP 'DATA HEADER' POINTER
4640 023226 001404 BEQ 5$ ;;SKIP TIMEOUT IF 0
4641 023230 104401 TYPE ;;TYPE THE 'DATA HEADER'
4642 023232 000000 4$: .WORD 0 ;;'DATA HEADER' POINTER GOES HERE
4643 023234 104401 001171 TYPE $CRLF ;;'CARRIAGE RETURN' & 'LINE FEED'
4644 023240 010146 5$: MOV R1,-(SP) ;;SAVE R1
4645 023242 012001 MOV (R0)+,R1 ;;PICKUP 'DATA TABLE' POINTER
  
```

```

4646 023244 001415          BEQ      9$          ;;BR IF NO DATA TO BE TYPED
4647 023246 012000          MOV      (R0)+,R0      ;;PICKUP 'DATA FORMAT' POINTER
4648 023250 105720          6$: TSTB   (R0)+          ;;'OCTAL' OR 'DECIMAL'
4649 023252 001003          BNE     7$          ;;BR IF DECIMAL
4650 023254 013146          MOV      @(R1)+,-(SP)  ;;SAVE @(R1)+ FOR TYPEOUT
4651 023256 104402          TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4652 023260 000402          BR      8$
4653 023262                    7$:
4654 023262 013146          MOV      @(R1)+,-(SP)  ;;SAVE @(R1)+ FOR TYPEOUT
4655 023264 104405          TYPDS                    ;;GO TYPE--DECIMAL ASCII WITH SIGN
4656 023266 005711          8$: TST      (R1)          ;;IS THERE ANOTHER NUMBER?
4657 023270 001403          BEQ     9$          ;;BR IF NO
4658 023272 104401 023312  TYPE     ,11$         ;;TYPE TWO(2) SPACES
4659 023276 000764          BR      6$          ;;LOOP

```

```

4660
4661 023300 012601          9$: MOV      (SP)+,R1    ;;RESTORE R1
4662 023302 012600          10$: MOV     (SP)+,R0   ;;RESTORE R0
4663 023304 104401 001171  TYPE     ,SCLRF       ;;'CARRIAGE RETURN' & 'LINE FEED'
4664 023310 000207          RTS     PC           ;;RETURN
4665 023312 020040 000          11$: .ASCIZ  / /       ;;TWO(2) SPACES
4666 023316

```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

```

4667
4668
4669 *****
4670 *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
4671 *OCTAL (ASCII) NUMBER AND TYPE IT.
4672 *$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
4673 *CALL:

```

```

4674 *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
4675 *      TYPOS                    ;;CALL FOR TYPEOUT
4676 *      .BYTE   N                ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
4677 *      .BYTE   M                ;;M=1 OR 0
4678 *                                  ;;1=TYPE LEADING ZEROS
4679 *                                  ;;0=SUPPRESS LEADING ZEROS

```

```

4680
4681 *$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
4682 *$TYPOS OR $TYPOC

```

```

4683 *CALL:
4684 *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
4685 *      TYPON                    ;;CALL FOR TYPEOUT

```

```

4686 *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

```

```

4687 *CALL:
4688 *      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
4689 *      TYPOC                    ;;CALL FOR TYPEOUT

```

```

4690
4691
4692 023316 017646 000000          $TYPOS: MOV     @(SP),-(SP)  ;;PICKUP THE MODE
4693 023322 116637 000001 023541  MOVB    1(SP),%OFILL      ;;LOAD ZERO FILL SWITCH
4694 023330 112637 023543          MOVB    (SP)+,%OMODE+1  ;;NUMBER OF DIGITS TO TYPE
4695 023334 062716 000002          ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
4696 023340 000406          BR      $TYPON
4697 023342 112737 000001 023541  $TYPOC: MOVB    #1,%OFILL  ;;SET THE ZERO FILL SWITCH
4698 023350 112737 000006 023543  MOVB    #6,%OMODE+1     ;;SET FOR SIX(6) DIGITS
4699 023356 112737 000005 023540  $TYPON: MOVB    #5,%OCNT  ;;SET THE ITERATION COUNT
4700 023364 010346          MOV     R3,-(SP)       ;;SAVE R3
4701 023366 010446          MOV     R4,-(SP)       ;;SAVE R4

```

```

4702 023370 010546          MOV      R5,-(SP)          ;;SAVE R5
4703 023372 113704 023543  MOVB     $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
4704 023376 005404          NEG      R4
4705 023400 062704 000006  ADD      #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
4706 023404 110437 023542  MOVB     R4,$OMODE       ;;SAVE IT FOR USE
4707 023410 113704 023541  MOVB     $OFILL,R4       ;;GET THE ZERO FILL SWITCH
4708 023414 016605 000012  MOV      12(SP),R5       ;;PICKUP THE INPUT NUMBER
4709 023420 005003          CLR      R3             ;;CLEAR THE OUTPUT WORD
4710 023422 006105          1$:    ROL      R5             ;;ROTATE MSB INTO 'C'
4711 023424 000404          BR       3$             ;;GO DO MSB
4712 023426 006105          2$:    ROL      R5             ;;FORM THIS DIGIT
4713 023430 006105          ROL      R5
4714 023432 006105          ROL      R5
4715 023434 010503          MOV      R5,R3
4716 023436 006103          3$:    ROL      R3             ;;GET LSB OF THIS DIGIT
4717 023440 105337 023542  DECB     $OMODE          ;;TYPE THIS DIGIT?
4718 023444 100016          BPL      7$             ;;BR IF NO
4719 023446 042703 177770  BIC      #177770,R3      ;;GET RID OF JUNK
4720 023452 001002          BNE      4$             ;;TEST FOR 0
4721 023454 005704          TST      R4             ;;SUPPRESS THIS 0?
4722 023456 001403          BEQ      5$             ;;BR IF YES
4723 023460 005204          4$:    INC      R4             ;;DON'T SUPPRESS ANYMORE 0'S
4724 023462 052703 000060  BIS      #'0,R3         ;;MAKE THIS DIGIT ASCII
4725 023466 052703 000040  5$:    BIS      #' ,R3         ;;MAKE ASCII IF NOT ALREADY
4726 023472 110337 023536  MOVB     R3,8$          ;;SAVE FOR TYPING
4727 023476 104401 023536  TYPE     ,8$           ;;GO TYPE THIS DIGIT
4728 023502 105337 023540  7$:    DECB     $OCNT      ;;COUNT BY 1
4729 023506 003347          BGT      2$             ;;BR IF MORE TO DO
4730 023510 002402          BLT      6$             ;;BR IF DONE
4731 023512 005204          INC      R4             ;;INSURE LAST DIGIT ISN'T A BLANK
4732 023514 000744          BR       2$             ;;GO DO THE LAST DIGIT
4733 023516 012605          6$:    MOV      (SP)+,R5     ;;RESTORE R5
4734 023520 012604          MOV      (SP)+,R4     ;;RESTORE R4
4735 023522 012603          MOV      (SP)+,R3     ;;RESTORE R3
4736 023524 016666 000002 000004  MOV      2(SP),4(SP)    ;;SET THE STACK FOR RETURNING
4737 023532 012616          MOV      (SP)+,(SP)
4738 023534 000002          RTI
4739 023536 000          8$:    .BYTE    0           ;;RETURN
4740 023537 000          .BYTE    0           ;;STORAGE FOR ASCII DIGIT
4741 023540 000          $OCNT:  .BYTE    0           ;;TERMINATOR FOR TYPE ROUTINE
4742 023541 000          $OFILL: .BYTE    0           ;;OCTAL DIGIT COUNTER
4743 023542 000000          $OMODE: .WORD    0           ;;ZERO FILL SWITCH
4744          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
4745
4746          ;;*****
4747          ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
4748          ;;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
4749          ;;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
4750          ;;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
4751          ;;*REPLACED WITH SPACES.
4752          ;;*CALL:
4753          ;;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
4754          ;;*      TYPDS          ;;GO TO THE ROUTINE
4755
4756          $TYPDS: MOV      R0,-(SP)      ;;PUSH R0 ON STACK
4757          023544 010046

```

```

4758 023546 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
4759 023550 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
4760 023552 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
4761 023554 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
4762 023556 012746 020200  MOV      #20200,-(SP)   ;;SET BLANK SWITCH AND SIGN
4763 023562 016605 000020  MOV      20(SP),R5     ;;GET THE INPUT NUMBER
4764 023566 100004      BPL      1$           ;;BR IF INPUT IS POS.
4765 023570 005405      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
4766 023572 112766 000055 000001  MOVB     #'-,1(SP)     ;;MAKE THE ASCII NUMBER NEG.
4767 023600 005000      CLR      R0           ;;ZERO THE CONSTANTS INDEX
4768 023602 012703 023760  MOV      #SDBLK,R3     ;;SETUP THE OUTPUT POINTER
4769 023606 112723 000040  MOVB     #' ,(R3)+     ;;SET THE FIRST CHARACTER TO A BLANK
4770 023612 005002      CLR      R2           ;;CLEAR THE BCD NUMBER
4771 023614 016001 023750  MOV      $DTBL(R0),R1  ;;GET THE CONSTANT
4772 023620 160105      3$:     SUB      R1,R5     ;;FORM THIS BCD DIGIT
4773 023622 002402      BLT     4$           ;;BR IF DONE
4774 023624 005202      INC     R2           ;;INCREASE THE BCD DIGIT BY 1
4775 023626 000774      BR      3$
4776 023630 060105      4$:     ADD      R1,R5     ;;ADD BACK THE CONSTANT
4777 023632 005702      TST     R2           ;;CHECK IF BCD DIGIT=0
4778 023634 001002      BNE     5$           ;;FALL THROUGH IF 0
4779 023636 105716      TSTB    (SP)         ;;STILL DOING LEADING 0'S?
4780 023640 100407      BMI     7$           ;;BR IF YES
4781 023642 106316      5$:     ASLB    (SP)         ;;MSD?
4782 023644 103003      BCC     6$           ;;BR IF NO
4783 023646 116663 000001 177777  MOVB     1(SP),-1(R3)  ;;YES--SET THE SIGN
4784 023654 052702 000060  6$:     BIS     #'0,R2     ;;MAKE THE BCD DIGIT ASCII
4785 023660 052702 000040  7$:     BIS     #' ,R2     ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
4786 023664 110223      MOVB     R2,(R3)+     ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
4787 023666 005720      TST     (R0)+        ;;JUST INCREMENTING
4788 023670 020027 000010  CMP      R0,#10      ;;CHECK THE TABLE INDEX
4789 023674 002746      BLT     2$           ;;GO DO THE NEXT DIGIT
4790 023676 003002      BGT     8$           ;;GO TO EXIT
4791 023700 010502      MOV     R5,R2        ;;GET THE LSD
4792 023702 000764      BR      6$           ;;GO CHANGE TO ASCII
4793 023704 105726      8$:     TSTB    (SP)+     ;;WAS THE LSD THE FIRST NON-ZERO?
4794 023706 100003      BPL     9$           ;;BR IF NO
4795 023710 116663 177777 177776  MOVB     -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
4796 023716 105013      9$:     CLRB    (R3)      ;;SET THE TERMINATOR
4797 023720 012605      MOV     (SP)+,R5     ;;POP STACK INTO R5
4798 023722 012603      MOV     (SP)+,R3     ;;POP STACK INTO R3
4799 023724 012602      MOV     (SP)+,R2     ;;POP STACK INTO R2
4800 023726 012601      MOV     (SP)+,R1     ;;POP STACK INTO R1
4801 023730 012600      MOV     (SP)+,R0     ;;POP STACK INTO R0
4802 023732 104401 023760  TYPE     ,SDBLK      ;;NOW TYPE THE NUMBER
4803 023736 016666 000002 000004  MOV     2(SP),4(SP)   ;;ADJUST THE STACK
4804 023744 012616      MOV     (SP)+,(SP)
4805 023746 000002      RTI
4806 023750 023420      $DTBL: 10000.
4807 023752 001750      1000.
4808 023754 000144      100.
4809 023756 000012      10.
4810 023760 000004      $SDBLK: .BLKW 4
4811      .SBTTL POWER DOWN AND UP ROUTINES
4812
4813      ;;*****

```



```

4814                                     :POWER DOWN ROUTINE
4815 023770 012737 024142 000024 $PWRDN: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
4816 023776 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
4817 024004 010046      MOV    R0,-(SP) ;;PUSH R0 ON STACK
4818 024006 010146      MOV    R1,-(SP) ;;PUSH R1 ON STACK
4819 024010 010246      MOV    R2,-(SP) ;;PUSH R2 ON STACK
4820 024012 010346      MOV    R3,-(SP) ;;PUSH R3 ON STACK
4821 024014 010446      MOV    R4,-(SP) ;;PUSH R4 ON STACK
4822 024016 010546      MOV    R5,-(SP) ;;PUSH R5 ON STACK
4823 024020 017746 155114      MOV    @SWR,-(SP) ;;PUSH @SWR ON STACK
4824 024024 010637 024146      MOV    SP,$SAVR6 ;;SAVE SP
4825 024030 012737 024042 000024      MOV    #$PWRUP,@#PWRVEC ;;SET UP VECTOR
4826 024036 000000      HALT
4827 024040 000776      BR      -2 ;;HANG UP
4828
4829                                     :*****
4830                                     :POWER UP ROUTINE
4831 024042 012737 024142 000024 $PWRUP: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
4832 024050 013706 024146      MOV    $SAVR6,SP ;;GET SP
4833 024054 005037 024146      CLR    $SAVR6 ;;WAIT LOOP FOR THE TTY
4834 024060 005237 024146 1$: INC    $SAVR6 ;;WAIT FOR THE INC
4835 024064 001375      BNE    1$ ;;OF WORD
4836 024066 012677 155046      MOV    (SP)+,@SWR ;;POP STACK INTO @SWR
4837 024072 012605      MOV    (SP)+,R5 ;;POP STACK INTO R5
4838 024074 012604      MOV    (SP)+,R4 ;;POP STACK INTO R4
4839 024076 012603      MOV    (SP)+,R3 ;;POP STACK INTO R3
4840 024100 012602      MOV    (SP)+,R2 ;;POP STACK INTO R2
4841 024102 012601      MOV    (SP)+,R1 ;;POP STACK INTO R1
4842 024104 012600      MOV    (SP)+,R0 ;;POP STACK INTO R0
4843 024106 012737 023770 000024      MOV    #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
4844 024114 012737 000340 000026      MOV    #340,@#PWRVEC+2 ;;PRIO:7
4845 024122 104401      TYPE   ;;REPORT THE POWER FAILURE
4846 024124 024150 $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
4847 024126 042766 000020 000002      BIC    #20,2(SP) ;;CLEAR 'T' BIT
4848 024134 005037 011320      CLR    $TBIT ;;CLEAR THE 'T' BIT FLAG
4849 024140 000002      RTI
4850 024142 000000 $ILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
4851 024144 000776      BR      -2 ;;BEFORE THE POWER DOWN WAS COMPLETE
4852 024146 000000 $SAVR6: 0 ;;PUT THE SP HERE
4853 024150 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER'
4854 024156 000122
4855
4856                                     .EVEN
4857                                     .SBTTL TRAP DECODER
4858
4859                                     :*****
4860                                     :*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
4861                                     :*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
4862                                     :*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
4863                                     :*GO TO THAT ROUTINE.
4864 024160 010046 $TRAP: MOV    R0,-(SP) ;;SAVE R0
4865 024162 016600 000002      MOV    2(SP),R0 ;;GET TRAP ADDRESS
4866 024166 005740      TST    -(R0) ;;BACKUP BY 2
4867 024170 111000      MOVB   (R0),R0 ;;GET RIGHT BYTE OF TRAP
4868 024172 006300      ASL    R0 ;;POSITION FOR INDEXING
4869 024174 016000 024214      MOV    $TRPAD(R0),R0 ;;INDEX TO TABLE

```

4870 024200 000200 RTS R0 ;;GO TO ROUTINE

4871
 4872
 4873 ;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO

4874
 4875 024202 011646 \$TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
 4876 024204 016666 000004 000002 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
 4877 024212 000002 RTI ;;RESTORE THE PSW

4878
 4879 .SBTTL TRAP TABLE

4880
 4881 ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 4882 ;*BY THE 'TRAP' INSTRUCTION.

4883
 4884 : ROUTINE
 4885 :
 4886 \$TRPAD: .WORD \$TRAP2
 4887 \$TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
 4888 \$TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 4889 \$TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
 4890 \$TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
 4891 \$TYPDS ;;CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)

4892
 4893
 4894
 4895 024230 003340 003732 004140 TSTADD: TST1,TST2,TST3
 4896
 4897 024236 004334 004466 004726 TST4,TST5,TST6
 4898
 4899 024244 005156 005732 006462 TST7,TST10,TST11
 4900
 4901 024252 006674 007050 007276 TST12,TST13,TST14
 4902
 4903 024260 007520 010256 010474 TST15,TST16,TST17
 4904
 4905 024266 010642 011342 011646 TST20,TST21,TST22
 4906
 4907 024274 012044 012116 TST23,TST24

4908
 4909
 4910
 4911 024300 042523 020124 052126 STUPM: .ASCII /SET VT61S TO FULL DUPLEX, /<15><12>
 4912 024306 030466 020123 047524
 4913 024314 020040 052506 046114
 4914 024322 042040 050125 042514
 4915 024330 026130 006440 012
 4916 024335 071 030066 041060 .ASCIIZ /9600BAUD, REMOTE,PARITY MATCHED TO INTERFACE/<15><12>
 4917 024342 052501 026104 051040
 4918 024350 046505 052117 026105
 4919 024356 040520 044522 054524
 4920 024364 046440 052101 044103
 4921 024372 042105 052040 020117
 4922 024400 047111 042524 043122
 4923 024406 041501 006505 000012

4924
 4925

| | | | | | | |
|------|--------|--------|--------|--------|----------------|--|
| 4926 | | | | | | |
| 4927 | 024414 | 005015 | 042101 | 051104 | DUNTST: .ASCIZ | <15><12>/ADDRESSES WITH RESPONSIVE VT61S ARE:/<15><12> |
| 4928 | 024422 | 051505 | 042523 | 020123 | | |
| 4929 | 024430 | 044527 | 044124 | 051040 | | |
| 4930 | 024436 | 051505 | 047520 | 051516 | | |
| 4931 | 024444 | 053111 | 020105 | 052126 | | |
| 4932 | 024452 | 030466 | 020123 | 051101 | | |
| 4933 | 024460 | 035105 | 005015 | 000 | | |
| 4934 | 024465 | 116 | 020117 | 052126 | NOVT: .ASCIZ | /NO VT61 RESPONDED TO ESCZ SEQ. AUTO RETRY IN 30 SEC./<15><12> |
| 4935 | 024472 | 030466 | 051040 | 051505 | | |
| 4936 | 024500 | 047520 | 042116 | 042105 | | |
| 4937 | 024506 | 052040 | 020117 | 051505 | | |
| 4938 | 024514 | 055103 | 051440 | 050505 | | |
| 4939 | 024522 | 020056 | 052501 | 047524 | | |
| 4940 | 024530 | 051040 | 052105 | 054522 | | |
| 4941 | 024536 | 044440 | 020116 | 030063 | | |
| 4942 | 024544 | 051440 | 041505 | 006456 | | |
| 4943 | 024552 | 000012 | | | | |
| 4944 | | | | | | |
| 4945 | | | | | | |
| 4946 | 024554 | 005015 | 045104 | 030461 | DLERR: .ASCIZ | <15><12>/DJ11 FAILED AT ADDRESS/ |
| 4947 | 024562 | 043040 | 044501 | 042514 | | |
| 4948 | 024570 | 020104 | 052101 | 040440 | | |
| 4949 | 024576 | 042104 | 042522 | 051523 | | |
| 4950 | 024604 | 000 | | | | |
| 4951 | | | | | | |
| 4952 | 024605 | 115 | 047101 | 040525 | DMANA: .ASCII | /MANUAL TEST SELECTED -/<15><12> |
| 4953 | 024612 | 020114 | 042524 | 052123 | | |
| 4954 | 024620 | 051440 | 046105 | 041505 | | |
| 4955 | 024626 | 042524 | 020104 | 006455 | | |
| 4956 | 024634 | 012 | | | | |
| 4957 | 024635 | 105 | 052116 | 051105 | .ASCIZ | /ENTER ADDRESSES OF VT61S TO BE TESTED/<15><12> |
| 4958 | 024642 | 040440 | 042104 | 042522 | | |
| 4959 | 024650 | 051523 | 051505 | 047440 | | |
| 4960 | 024656 | 020106 | 052126 | 030466 | | |
| 4961 | 024664 | 020123 | 047524 | 041040 | | |
| 4962 | 024672 | 020105 | 042524 | 052123 | | |
| 4963 | 024700 | 042105 | 005015 | 000 | | |
| 4964 | | | | | | |
| 4965 | 024705 | 105 | 052116 | 051105 | DMANB: .ASCIZ | /ENTER TESTS TO BE RUN/<15><12> |
| 4966 | 024712 | 052040 | 051505 | 051524 | | |
| 4967 | 024720 | 052040 | 020117 | 042502 | | |
| 4968 | 024726 | 051040 | 047125 | 005015 | | |
| 4969 | 024734 | 000 | | | | |
| 4970 | 024735 | 105 | 052116 | 051105 | DMANL: .ASCIZ | /ENTER LINES TO BE TESTED IN BINARY FORMAT(I.E.0=1,10=2000)/<15><12> |
| 4971 | 024742 | 046040 | 047111 | 051505 | | |
| 4972 | 024750 | 052040 | 020117 | 042502 | | |
| 4973 | 024756 | 052040 | 051505 | 042524 | | |
| 4974 | 024764 | 020104 | 047111 | 041040 | | |
| 4975 | 024772 | 047111 | 051101 | 020131 | | |
| 4976 | 025000 | 047506 | 046522 | 052101 | | |
| 4977 | 025006 | 044450 | 042456 | 030056 | | |
| 4978 | 025014 | 030475 | 030454 | 036460 | | |
| 4979 | 025022 | 030062 | 030060 | 006451 | | |
| 4980 | 025030 | 000012 | | | | |
| 4981 | | | | | | |

| | | | | | | |
|------|--------|--------|--------|--------|------|---|
| 4982 | 025032 | 047101 | 042440 | 041523 | EM1: | .ASCIZ /AN ESC SEQ. TO THE VT61 FAILED - OCTAL EQUIV. IS: /<15><12> |
| 4983 | 025040 | 051440 | 050505 | 020056 | | |
| 4984 | 025046 | 047524 | 052040 | 042510 | | |
| 4985 | 025054 | 053040 | 033124 | 020061 | | |
| 4986 | 025062 | 043040 | 044501 | 042514 | | |
| 4987 | 025070 | 020104 | 020055 | 041517 | | |
| 4988 | 025076 | 040524 | 020114 | 050505 | | |
| 4989 | 025104 | 044525 | 027126 | 044440 | | |
| 4990 | 025112 | 035123 | 005015 | 000 | | |
| 4991 | 025117 | 124 | 051505 | 021524 | DH1: | .ASCIZ /TEST# ERR PC BYTE 1+2 BYTE 3+4 /<15><12> |
| 4992 | 025124 | 020040 | 051105 | 020122 | | |
| 4993 | 025132 | 041520 | 020040 | 054502 | | |
| 4994 | 025140 | 042524 | 030440 | 031053 | | |
| 4995 | 025146 | 041040 | 052131 | 020105 | | |
| 4996 | 025154 | 025463 | 006464 | 000012 | | |
| 4997 | | | | | | |
| 4998 | 025162 | 042522 | 042503 | 053111 | EM2: | .ASCIZ /RECEIVE STATUS ERROR. /<15><12> |
| 4999 | 025170 | 020105 | 052123 | 052101 | | |
| 5000 | 025176 | 051525 | 042440 | 051122 | | |
| 5001 | 025204 | 051117 | 006456 | 000012 | | |
| 5002 | 025212 | 042101 | 027104 | 020040 | DH2: | .ASCIZ /ADD. STAT. ERR.BITS CHAR. /<15><12> |
| 5003 | 025220 | 052123 | 052101 | 020056 | | |
| 5004 | 025226 | 042440 | 051122 | 041056 | | |
| 5005 | 025234 | 052111 | 020123 | 041440 | | |
| 5006 | 025242 | 040510 | 027122 | 005015 | | |
| 5007 | 025250 | 000 | | | | |
| 5008 | | | | | | |
| 5009 | 025251 | 123 | 043117 | 053524 | EM3: | .ASCIZ /SOFTWARE (VSTAT) STATUS ERROR. /<15><12> |
| 5010 | 025256 | 051101 | 020105 | 053050 | | |
| 5011 | 025264 | 052123 | 052101 | 020051 | | |
| 5012 | 025272 | 052123 | 052101 | 051525 | | |
| 5013 | 025300 | 042440 | 051122 | 051117 | | |
| 5014 | 025306 | 006456 | 000012 | | | |
| 5015 | 025312 | 050040 | 051501 | 021523 | DH3: | .ASCIZ / PASS#, TEST#, EXP.STAT, ACT.STAT /<15><12> |
| 5016 | 025320 | 020054 | 052040 | 051505 | | |
| 5017 | 025326 | 021524 | 020054 | 042440 | | |
| 5018 | 025334 | 050130 | 051456 | 040524 | | |
| 5019 | 025342 | 026124 | 020040 | 041501 | | |
| 5020 | 025350 | 027124 | 052123 | 052101 | | |
| 5021 | 025356 | 005015 | 000 | | | |
| 5022 | | | | | | |
| 5023 | 025361 | 107 | 027104 | 042040 | EM4: | .ASCIZ /GD. DATA DOES NOT MATCH REC. DATA /<15><12> |
| 5024 | 025366 | 052101 | 020101 | 047504 | | |
| 5025 | 025374 | 051505 | 047040 | 052117 | | |
| 5026 | 025402 | 046440 | 052101 | 044103 | | |
| 5027 | 025410 | 051040 | 041505 | 020056 | | |
| 5028 | 025416 | 040504 | 040524 | 005015 | | |
| 5029 | 025424 | 000 | | | | |
| 5030 | 025425 | 124 | 051505 | 021524 | DH4: | .ASCIZ /TEST# ,REC.CNT.,GD. DATA, REC. DATA /<15><12> |
| 5031 | 025432 | 026040 | 042522 | 027103 | | |
| 5032 | 025440 | 047103 | 027124 | 043454 | | |
| 5033 | 025446 | 027104 | 042040 | 052101 | | |
| 5034 | 025454 | 026101 | 051040 | 041505 | | |
| 5035 | 025462 | 020056 | 040504 | 040524 | | |
| 5036 | 025470 | 005015 | 000 | | | |
| 5037 | | 025474 | | | | .EVEN |

| | | | | | | |
|------|--------|--------|--------|--------|-------|--|
| 5038 | | | | | | |
| 5039 | 025474 | 054502 | 042524 | 020123 | EM5: | .ASCIZ /BYTES EXPECTED DOES NOT EQUAL BYTES RECEIVED/<15><12> |
| 5040 | 025502 | 054105 | 042520 | 052103 | | |
| 5041 | 025510 | 042105 | 042040 | 042517 | | |
| 5042 | 025516 | 020123 | 047516 | 020124 | | |
| 5043 | 025524 | 050505 | 040525 | 020114 | | |
| 5044 | 025532 | 054502 | 042524 | 020123 | | |
| 5045 | 025540 | 042522 | 042503 | 053111 | | |
| 5046 | 025546 | 042105 | 005015 | 000 | | |
| 5047 | 025553 | 102 | 052131 | 051505 | DH5: | .ASCIZ /BYTES EXP., BYTES REC./<15><12> |
| 5048 | 025560 | 042440 | 050130 | 026056 | | |
| 5049 | 025566 | 041040 | 052131 | 051505 | | |
| 5050 | 025574 | 051040 | 041505 | 006456 | | |
| 5051 | 025602 | 000012 | | | | |
| 5052 | | | | | | |
| 5053 | 025604 | 052503 | 051522 | 051117 | EM6: | .ASCIZ /CURSOR POSITIONING ERROR/<15><12> |
| 5054 | 025612 | 050040 | 051517 | 052111 | | |
| 5055 | 025620 | 047511 | 044516 | 043516 | | |
| 5056 | 025626 | 042440 | 051122 | 051117 | | |
| 5057 | 025634 | 005015 | 000 | | | |
| 5058 | 025637 | 107 | 020104 | 044514 | DH6: | .ASCIZ /GD LINE GD COL. BD LINE BD COL/<15><12> |
| 5059 | 025644 | 042516 | 020040 | 042107 | | |
| 5060 | 025652 | 041440 | 046117 | 020056 | | |
| 5061 | 025660 | 020040 | 042102 | 046040 | | |
| 5062 | 025666 | 047111 | 020105 | 041040 | | |
| 5063 | 025674 | 020104 | 047503 | 006514 | | |
| 5064 | 025702 | 000012 | | | | |
| 5065 | | | | | | |
| 5066 | 025704 | 044504 | 042522 | 052103 | EM7: | .ASCIZ /DIRECT CURSOR ADDRESSING FAILURE/<15><12> |
| 5067 | 025712 | 041440 | 051125 | 047523 | | |
| 5068 | 025720 | 020122 | 042101 | 051104 | | |
| 5069 | 025726 | 051505 | 044523 | 043516 | | |
| 5070 | 025734 | 043040 | 044501 | 052514 | | |
| 5071 | 025742 | 042522 | 005015 | 000 | | |
| 5072 | 025747 | 120 | 051501 | 021523 | DH7: | .ASCIZ /PASS# TEST # ERROR PC /<15><12> |
| 5073 | 025754 | 020040 | 042524 | 052123 | | |
| 5074 | 025762 | 021440 | 020040 | 051105 | | |
| 5075 | 025770 | 047522 | 020122 | 041520 | | |
| 5076 | 025776 | 020040 | 006440 | 000012 | | |
| 5077 | 026004 | 040520 | 051523 | 020043 | DH10: | .ASCIZ /PASS# TEST# BD.ROW BD.COL/<15><12> |
| 5078 | 026012 | 052040 | 051505 | 021524 | | |
| 5079 | 026020 | 020040 | 042102 | 051056 | | |
| 5080 | 026026 | 053517 | 020040 | 042102 | | |
| 5081 | 026034 | 041456 | 046117 | 005015 | | |
| 5082 | 026042 | 000 | | | | |
| 5083 | | | | | | |
| 5084 | 026043 | 114 | 051501 | 020124 | EM11: | .ASCIZ /LAST TRANSMISSION TO VT61 CAUSED UNIT TO FAIL-HANG./<15><12> |
| 5085 | 026050 | 051124 | 047101 | 046523 | | |
| 5086 | 026056 | 051511 | 044523 | 047117 | | |
| 5087 | 026064 | 052040 | 020117 | 052126 | | |
| 5088 | 026072 | 030466 | 041440 | 052501 | | |
| 5089 | 026100 | 042523 | 020104 | 047125 | | |
| 5090 | 026106 | 052111 | 052040 | 020117 | | |
| 5091 | 026114 | 040506 | 046111 | 044055 | | |
| 5092 | 026122 | 047101 | 027107 | 005015 | | |
| 5093 | 026130 | 000 | | | | |

| | | | | | |
|------|--------|--------|--------|--------|---|
| 5094 | | | | | |
| 5095 | 026131 | 126 | 033124 | 020061 | EM12: .ASCIZ /VT61 UNDER TEST FAILED- ERROR DATA FOLLOWS/<15><12> |
| 5096 | 026136 | 047125 | 042504 | 020122 | |
| 5097 | 026144 | 042524 | 052123 | 043040 | |
| 5098 | 026152 | 044501 | 042514 | 026504 | |
| 5099 | 026160 | 042440 | 051122 | 051117 | |
| 5100 | 026166 | 042040 | 052101 | 020101 | |
| 5101 | 026174 | 047506 | 046114 | 053517 | |
| 5102 | 026202 | 006523 | 000012 | | |
| 5103 | | | | | |
| 5104 | 026206 | 052126 | 030466 | 043040 | EM10: .ASCIZ /VT61 FAILED SELF TEST FUNCTION/<15><12> |
| 5105 | 026214 | 044501 | 042514 | 020104 | |
| 5106 | 026222 | 042523 | 043114 | 052040 | |
| 5107 | 026230 | 051505 | 020124 | 052506 | |
| 5108 | 026236 | 041516 | 044524 | 047117 | |
| 5109 | 026244 | 005015 | 000 | | |
| 5110 | | | | | |
| 5111 | | | | | |
| 5112 | 026247 | 120 | 051501 | 021523 | DH12: .ASCIZ /PASS#, TEST#, GD.CKSUM, BD.CKSUM/<15><12> |
| 5113 | 026254 | 020054 | 052040 | 051505 | |
| 5114 | 026262 | 021524 | 020054 | 042107 | |
| 5115 | 026270 | 041456 | 051513 | 046525 | |
| 5116 | 026276 | 020054 | 042102 | 041456 | |
| 5117 | 026304 | 051513 | 046525 | 005015 | |
| 5118 | 026312 | 000 | | | |
| 5119 | | | | | |
| 5120 | 026313 | 124 | 051505 | 044524 | DABRT: .ASCIZ /TESTING ABORTED-TOO MANY FATAL XMITTS/<15><12> |
| 5121 | 026320 | 043516 | 040440 | 047502 | |
| 5122 | 026326 | 052122 | 042105 | 052055 | |
| 5123 | 026334 | 047517 | 046440 | 047101 | |
| 5124 | 026342 | 020131 | 040506 | 040524 | |
| 5125 | 026350 | 020114 | 046530 | 052111 | |
| 5126 | 026356 | 006523 | 000012 | | |
| 5127 | | | | | |
| 5128 | 026362 | 052126 | 030466 | 051040 | EM13: .ASCIZ /VT61 RECEIVER CHECKSUM COMPARE ERROR/<15><12> |
| 5129 | 026370 | 041505 | 044505 | 042526 | |
| 5130 | 026376 | 020122 | 044103 | 041505 | |
| 5131 | 026404 | 051513 | 046525 | 041440 | |
| 5132 | 026412 | 046517 | 040520 | 042522 | |
| 5133 | 026420 | 042440 | 051122 | 051117 | |
| 5134 | 026426 | 005015 | 000 | | |
| 5135 | | | | | |
| 5136 | 026431 | 126 | 033124 | 020061 | EM14: .ASCIZ /VT61 TRANSMITTER CHECKSUM COMPARE ERROR/<15><12> |
| 5137 | 026436 | 051124 | 047101 | 046523 | |
| 5138 | 026444 | 052111 | 042524 | 020122 | |
| 5139 | 026452 | 044103 | 041505 | 051513 | |
| 5140 | 026460 | 046525 | 041440 | 046517 | |
| 5141 | 026466 | 040520 | 042522 | 042440 | |
| 5142 | 026474 | 051122 | 051117 | 005015 | |
| 5143 | 026502 | 000 | | | |
| 5144 | | | | | |
| 5145 | | 026504 | | | .EVEN |
| 5146 | 026504 | 047125 | 052111 | 052440 | DVUNIT: .ASCII /UNIT UNDER TEST /<15><12> |
| 5147 | 026512 | 042116 | 051105 | 052040 | |
| 5148 | 026520 | 051505 | 020124 | 005015 | |
| 5149 | 026526 | 041522 | 051123 | 020040 | .ASCIZ /RCR VECT. LINE IDENT/<15><12> |

| | | | | | |
|------|--------|--------|--------|--------|--|
| 5150 | 026534 | 053040 | 041505 | 027124 | |
| 5151 | 026542 | 020040 | 046040 | 047111 | |
| 5152 | 026550 | 020105 | 044440 | 042504 | |
| 5153 | 026556 | 052116 | 005015 | 000 | |
| 5154 | 026563 | 040 | 041522 | 051123 | DH11: .ASCIZ / RCSR VECT./<15><12> |
| 5155 | 026570 | 020040 | 053040 | 041505 | |
| 5156 | 026576 | 027124 | 005015 | 000 | |
| 5157 | 026603 | 120 | 044522 | 052116 | DPRTR: .ASCIZ /PRINTER IS ATTACHED/<15><12> |
| 5158 | 026610 | 051105 | 044440 | 020123 | |
| 5159 | 026616 | 052101 | 040524 | 044103 | |
| 5160 | 026624 | 042105 | 005015 | 000 | |
| 5161 | 026631 | 103 | 050117 | 042511 | DCOPYR: .ASCIZ /COPIER IS ATTACHED/<15><12> |
| 5162 | 026636 | 020122 | 051511 | 040440 | |
| 5163 | 026644 | 052124 | 041501 | 042510 | |
| 5164 | 026652 | 006504 | 000012 | | |
| 5165 | 026656 | 047530 | 043106 | 052040 | EM15: .ASCIZ /XOFF TO VT61 FAILED TO HALT BLOCK XMIT/<15><12> |
| 5166 | 026664 | 020117 | 052126 | 030466 | |
| 5167 | 026672 | 043040 | 044501 | 042514 | |
| 5168 | 026700 | 020104 | 047524 | 044040 | |
| 5169 | 026706 | 046101 | 020124 | 046102 | |
| 5170 | 026714 | 041517 | 020113 | 046530 | |
| 5171 | 026722 | 052111 | 005015 | 000 | |
| 5172 | 026727 | 130 | 047117 | 052040 | EM16: .ASCIZ /XON TO VT61 FAILED TO RESTART BLOCK XMIT/<15><12> |
| 5173 | 026734 | 020117 | 052126 | 030466 | |
| 5174 | 026742 | 043040 | 044501 | 042514 | |
| 5175 | 026750 | 020104 | 047524 | 051040 | |
| 5176 | 026756 | 051505 | 040524 | 052122 | |
| 5177 | 026764 | 041040 | 047514 | 045503 | |
| 5178 | 026772 | 054040 | 044515 | 006524 | |
| 5179 | 027000 | 000012 | | | |
| 5180 | 027002 | 047516 | 054040 | 047117 | EM17: .ASCIZ /NO XON RECEIVED WITHIN 3 SEC. AFTER A RESET/<15><12> |
| 5181 | 027010 | 051040 | 041505 | 044505 | |
| 5182 | 027016 | 042526 | 020104 | 044527 | |
| 5183 | 027024 | 044124 | 047111 | 031440 | |
| 5184 | 027032 | 051440 | 041505 | 020056 | |
| 5185 | 027040 | 043101 | 042524 | 020122 | |
| 5186 | 027046 | 020101 | 042522 | 042523 | |
| 5187 | 027054 | 006524 | 000012 | | |
| 5188 | 027060 | 040514 | 052123 | 050040 | EM20: .ASCIZ /LAST PERIPHERAL OPERATION ABORTED/<15><12> |
| 5189 | 027066 | 051105 | 050111 | 042510 | |
| 5190 | 027074 | 040522 | 020114 | 050117 | |
| 5191 | 027102 | 051105 | 052101 | 047511 | |
| 5192 | 027110 | 020116 | 041101 | 051117 | |
| 5193 | 027116 | 042524 | 006504 | 000012 | |
| 5194 | 027124 | 047503 | 046125 | 020104 | EM21: .ASCIZ /COULD NOT CLEAR LAST ABORT FLAG./<15><12> |
| 5195 | 027132 | 047516 | 020124 | 046103 | |
| 5196 | 027140 | 040505 | 020122 | 040514 | |
| 5197 | 027146 | 052123 | 040440 | 047502 | |
| 5198 | 027154 | 052122 | 043040 | 040514 | |
| 5199 | 027162 | 027107 | 005015 | 000 | |
| 5200 | 027167 | 123 | 046517 | 047440 | EM22: .ASCIZ /SOM OR EOM NOT RECEIVED DURING MAINT. MODE TRANSMIT/<15><12> |
| 5201 | 027174 | 020122 | 047505 | 020115 | |
| 5202 | 027202 | 047516 | 020124 | 042522 | |
| 5203 | 027210 | 042503 | 053111 | 042105 | |
| 5204 | 027216 | 042040 | 051125 | 047111 | |
| 5205 | 027224 | 020107 | 040515 | 047111 | |

| | | | | | |
|------|--------|--------|--------|--------|---|
| 5206 | 027232 | 027124 | 046440 | 042117 | |
| 5207 | 027240 | 020105 | 051124 | 047101 | |
| 5208 | 027246 | 046523 | 052111 | 005015 | |
| 5209 | 027254 | 000 | | | |
| 5210 | 027255 | 114 | 047111 | 020105 | EM23: .ASCIZ /LINE FEED OR CURSOR RIGHT ISSUED FROM ROW 23 DID NOT CAUSE SCREEN TO SC |
| 5211 | 027262 | 042506 | 042105 | 047440 | |
| 5212 | 027270 | 020122 | 052503 | 051522 | |
| 5213 | 027276 | 051117 | 051040 | 043511 | |
| 5214 | 027304 | 052110 | 044440 | 051523 | |
| 5215 | 027312 | 042525 | 020104 | 051106 | |
| 5216 | 027320 | 046517 | 051040 | 053517 | |
| 5217 | 027326 | 031040 | 020063 | 044504 | |
| 5218 | 027334 | 020104 | 047516 | 020124 | |
| 5219 | 027342 | 040503 | 051525 | 020105 | |
| 5220 | 027350 | 041523 | 042522 | 047105 | |
| 5221 | 027356 | 052040 | 020117 | 041523 | |
| 5222 | 027364 | 047522 | 046114 | 005015 | |
| 5223 | 027372 | 000 | | | |
| 5224 | 027373 | 120 | 051501 | 020123 | DH13: .ASCIZ /PASS , TEST , VSTAT/<15><12> |
| 5225 | 027400 | 020054 | 020040 | 042524 | |
| 5226 | 027406 | 052123 | 026040 | 020040 | |
| 5227 | 027414 | 053040 | 052123 | 052101 | |
| 5228 | 027422 | 005015 | 000 | | |
| 5229 | 027425 | 120 | 051501 | 026123 | DH14: .ASCIZ /PASS, TEST, ERR PC, VSTAT/<15><12> |
| 5230 | 027432 | 020040 | 052040 | 051505 | |
| 5231 | 027440 | 026124 | 020040 | 042440 | |
| 5232 | 027446 | 051122 | 050040 | 026103 | |
| 5233 | 027454 | 020040 | 053040 | 052123 | |
| 5234 | 027462 | 052101 | 005015 | 000 | |
| 5235 | | | | | |
| 5236 | 027467 | 105 | 041523 | 000040 | DESC: .ASCIZ /ESC / |
| 5237 | | | | | |
| 5238 | | | | | |
| 5239 | | | | | |
| 5240 | 027474 | 042513 | 041131 | 040517 | DKYBD: .ASCII /KEYBOARD TEST/<15><12> |
| 5241 | 027502 | 042122 | 052040 | 051505 | |
| 5242 | 027510 | 006524 | 012 | | |
| 5243 | 027513 | 113 | 054505 | 052123 | .ASCII /KEYSTROKES ECHO:/<15><12> |
| 5244 | 027520 | 047522 | 042513 | 020123 | |
| 5245 | 027526 | 041505 | 047510 | 006472 | |
| 5246 | 027534 | 012 | | | |
| 5247 | 027535 | 101 | 042040 | 051511 | .ASCII /A DISPLAY CHAR. = A DISPLAY CHAR./<15><12> |
| 5248 | 027542 | 046120 | 054501 | 041440 | |
| 5249 | 027550 | 040510 | 027122 | 036440 | |
| 5250 | 027556 | 040440 | 042040 | 051511 | |
| 5251 | 027564 | 046120 | 054501 | 041440 | |
| 5252 | 027572 | 040510 | 027122 | 005015 | |
| 5253 | 027600 | 031463 | 036440 | 042440 | .ASCII /33 = ESC/<15><12> |
| 5254 | 027606 | 041523 | 005015 | | |
| 5255 | 027612 | 032461 | 036440 | 041440 | .ASCII /15 = C-R/<15><12> |
| 5256 | 027620 | 051055 | 005015 | | |
| 5257 | 027624 | 031061 | 036440 | 046040 | .ASCII /12 = L-f/<15><12> |
| 5258 | 027632 | 043055 | 005015 | | |
| 5259 | 027636 | 033460 | 036440 | 041040 | .ASCII /07 = BELL/<15><12> |
| 5260 | 027644 | 046105 | 006514 | 012 | |
| 5261 | 027651 | 061 | 020060 | 020075 | .ASCII /10 = TAB/<15><12> |

| | | | | | |
|------|--------|--------|--------|--------|---|
| 5262 | 027656 | 040524 | 006502 | 012 | |
| 5263 | 027663 | 116 | 047117 | 042055 | .ASCIZ /NON-DISPLAY CHAR.= OCTAL EQUIV/<15><12> |
| 5264 | 027670 | 051511 | 046120 | 054501 | |
| 5265 | 027676 | 041440 | 040510 | 027122 | |
| 5266 | 027704 | 020075 | 041517 | 040524 | |
| 5267 | 027712 | 020114 | 050505 | 044525 | |
| 5268 | 027720 | 006526 | 000012 | | |
| 5269 | | | | | |
| 5270 | 027724 | 040524 | 020102 | 000 | DTAB: .ASCIZ /TAB / |
| 5271 | 027731 | 103 | 051055 | 000040 | DCR: .ASCIZ /C-R / |
| 5272 | 027736 | 026514 | 020106 | 000 | DLF: .ASCIZ /L-F / |
| 5273 | 027743 | 102 | 046105 | 020114 | DBELL: .ASCIZ /BELL / |
| 5274 | 027750 | 000 | | | |
| 5275 | | | | | |
| 5276 | 027751 | 114 | 047517 | 020120 | DLOOP: .ASCII /LOOP TEST - LOOP COMMANDS AND DATA THRU/<15><12> |
| 5277 | 027756 | 042524 | 052123 | 026440 | |
| 5278 | 027764 | 046040 | 047517 | 020120 | |
| 5279 | 027772 | 047503 | 046515 | 047101 | |
| 5280 | 030000 | 051504 | 040440 | 042116 | |
| 5281 | 030006 | 042040 | 052101 | 020101 | |
| 5282 | 030014 | 044124 | 052522 | 005015 | |
| 5283 | 030022 | 047510 | 052123 | 041040 | .ASCII /HOST BACK TO VT61 UNDER TEST. /<15><12> |
| 5284 | 030030 | 041501 | 020113 | 047524 | |
| 5285 | 030036 | 053040 | 033124 | 020061 | |
| 5286 | 030044 | 047125 | 042504 | 020122 | |
| 5287 | 030052 | 042524 | 052123 | 020056 | |
| 5288 | 030060 | 005015 | | | |
| 5289 | 030062 | 047503 | 052116 | 047522 | DCNTZ: .ASCIZ /CONTROL C EXITS TEST./<15><12> |
| 5290 | 030070 | 020114 | 020103 | 042440 | |
| 5291 | 030076 | 044530 | 051524 | 052040 | |
| 5292 | 030104 | 051505 | 027124 | 005015 | |
| 5293 | 030112 | 000 | | | |
| 5294 | | | | | |
| 5295 | 030113 | 105 | 044530 | 020124 | DEXT: .ASCIZ /EXIT TEST./ |
| 5296 | 030120 | 042524 | 052123 | 000056 | |
| 5297 | | | | | |
| 5298 | 030126 | 051120 | 047111 | 042524 | DPRNT: .ASCII /PRINTER TEST -/<15><12> |
| 5299 | 030134 | 020122 | 042524 | 052123 | |
| 5300 | 030142 | 026440 | 005015 | | |
| 5301 | 030146 | 031461 | 020062 | 047503 | .ASCII /132 COLUMNS OF A SLIDING PATTERN WILL BE/ |
| 5302 | 030154 | 052514 | 047115 | 020123 | |
| 5303 | 030162 | 043117 | 040440 | 051440 | |
| 5304 | 030170 | 044514 | 044504 | 043516 | |
| 5305 | 030176 | 050040 | 052101 | 042524 | |
| 5306 | 030204 | 047122 | 053440 | 046111 | |
| 5307 | 030212 | 020114 | 042502 | | |
| 5308 | 030216 | 047503 | 052116 | 047111 | .ASCII /CONTINUOUSLY OUTPUTTED TO PRINTER/<15><12> |
| 5309 | 030224 | 052517 | 046123 | 020131 | |
| 5310 | 030232 | 052517 | 050124 | 052125 | |
| 5311 | 030240 | 042524 | 020104 | 047524 | |
| 5312 | 030246 | 050040 | 044522 | 052116 | |
| 5313 | 030254 | 051105 | 005015 | | |
| 5314 | 030260 | 040503 | 027122 | 051040 | DCRST: .ASCIZ /CAR. RET. TO START/<15><12> |
| 5315 | 030266 | 052105 | 020056 | 047524 | |
| 5316 | 030274 | 051440 | 040524 | 052122 | |
| 5317 | 030302 | 005015 | 000 | | |

| | | | | | |
|------|--------|--------|--------|--------|--|
| 5318 | | | | | |
| 5319 | 030305 | 114 | 051501 | 020124 | DEVERR: .ASCIZ /LAST XMIT CAUSED VT61 HANG/<15><12> |
| 5320 | 030312 | 046530 | 052111 | 041440 | |
| 5321 | 030320 | 052501 | 042523 | 020104 | |
| 5322 | 030326 | 052126 | 030466 | 044040 | |
| 5323 | 030334 | 047101 | 006507 | 000012 | |
| 5324 | 030342 | 000077 | | | QMRK: .ASCIZ /?/ |
| 5325 | 030344 | 051120 | 042117 | 041525 | DKBD: .ASCII /PRODUCTION KEYBOARD TEST. 10 ERRORS CAUSES TEST EXIT./<15><12> |
| 5326 | 030352 | 044524 | 047117 | 045440 | |
| 5327 | 030360 | 054505 | 047502 | 051101 | |
| 5328 | 030366 | 020104 | 042524 | 052123 | |
| 5329 | 030374 | 020056 | 030061 | 042440 | |
| 5330 | 030402 | 051122 | 051117 | 020123 | |
| 5331 | 030410 | 040503 | 051525 | 051505 | |
| 5332 | 030416 | 052040 | 051505 | 020124 | |
| 5333 | 030424 | 054105 | 052111 | 006456 | |
| 5334 | 030432 | 012 | | | |
| 5335 | 030433 | 104 | 050105 | 042522 | .ASCIZ /DEPRESS KEYS FROM LEFT TO RIGHT/<15><12> |
| 5336 | 030440 | 051523 | 045440 | 054505 | |
| 5337 | 030446 | 020123 | 051106 | 046517 | |
| 5338 | 030454 | 046040 | 043105 | 020124 | |
| 5339 | 030462 | 047524 | 051040 | 043511 | |
| 5340 | 030470 | 052110 | 005015 | 000 | |
| 5341 | 030475 | 104 | 050105 | 042522 | DLSHFT: .ASCIZ /DEPRESS LEFT SHIFT KEY AND THE 'A' KEY /<15><12> |
| 5342 | 030502 | 051523 | 046040 | 043105 | |
| 5343 | 030510 | 020124 | 044123 | 043111 | |
| 5344 | 030516 | 020124 | 042513 | 020131 | |
| 5345 | 030524 | 047101 | 020104 | 044124 | |
| 5346 | 030532 | 020105 | 040442 | 020042 | |
| 5347 | 030540 | 042513 | 020131 | 005015 | |
| 5348 | 030546 | 000 | | | |
| 5349 | 030547 | 104 | 050105 | 042522 | DTOP: .ASCIZ /DEPRESS KEYS IN TOP ROW/<15><12> |
| 5350 | 030554 | 051523 | 045440 | 054505 | |
| 5351 | 030562 | 020123 | 047111 | 052040 | |
| 5352 | 030570 | 050117 | 051040 | 053517 | |
| 5353 | 030576 | 005015 | 000 | | |
| 5354 | | | | | |
| 5355 | 030601 | 104 | 050105 | 042522 | DRSHFT: .ASCIZ /DEPRESS RIGHT SHIFT KEY AND THE 'A' KEY /<15><12> |
| 5356 | 030606 | 051523 | 051040 | 043511 | |
| 5357 | 030614 | 052110 | 051440 | 044510 | |
| 5358 | 030622 | 052106 | 045440 | 054505 | |
| 5359 | 030630 | 040440 | 042116 | 052040 | |
| 5360 | 030636 | 042510 | 021040 | 021101 | |
| 5361 | 030644 | 045440 | 054505 | 006440 | |
| 5362 | 030652 | 000012 | | | |
| 5363 | 030654 | 042504 | 051120 | 051505 | DSEC: .ASCIZ /DEPRESS KEYS IN SECOND ROW/<15><12> |
| 5364 | 030662 | 020123 | 042513 | 051531 | |
| 5365 | 030670 | 044440 | 020116 | 042523 | |
| 5366 | 030676 | 047503 | 042116 | 051040 | |
| 5367 | 030704 | 053517 | 005015 | 000 | |
| 5368 | | | | | |
| 5369 | 030711 | 104 | 050105 | 042522 | DTHRD: .ASCIZ /DEPRESS KEYS IN THIRD ROW BEGINNING WITH 'A' /<15><12> |
| 5370 | 030716 | 051523 | 045440 | 054505 | |
| 5371 | 030724 | 020123 | 052040 | 052040 | |
| 5372 | 030732 | 050110 | 042122 | 051040 | |
| 5373 | 030740 | 053517 | 041040 | 043505 | |

| | | | | | |
|------|--------|--------|--------|--------|---|
| 5374 | 030746 | 047111 | 044516 | 043516 | |
| 5375 | 030754 | 053440 | 052111 | 020110 | |
| 5376 | 030762 | 040447 | 006447 | 000012 | |
| 5377 | 030770 | 042504 | 051120 | 051505 | DCONT: .ASCIZ /DEPRESS CONTROL KEY ,AND THE 'A' KEY /<15><12> |
| 5378 | 030776 | 020123 | 047503 | 052116 | |
| 5379 | 031004 | 047522 | 020114 | 042513 | |
| 5380 | 031012 | 020131 | 040454 | 042116 | |
| 5381 | 031020 | 052040 | 042510 | 021040 | |
| 5382 | 031026 | 021101 | 045440 | 054505 | |
| 5383 | 031034 | 006440 | 000012 | | |
| 5384 | 031040 | 042504 | 051120 | 051505 | DBOT: .ASCIZ /DEPRESS KEYS IN FORTH ROW EXCEPT SHIFT KEYS/<15><12> |
| 5385 | 031046 | 020123 | 042513 | 051531 | |
| 5386 | 031054 | 044440 | 020116 | 047506 | |
| 5387 | 031062 | 052122 | 020110 | 047522 | |
| 5388 | 031070 | 020127 | 054105 | 042503 | |
| 5389 | 031076 | 052120 | 051440 | 044510 | |
| 5390 | 031104 | 052106 | 045440 | 054505 | |
| 5391 | 031112 | 006523 | 000012 | | |
| 5392 | 031116 | 042504 | 051120 | 051505 | DSPCE: .ASCIZ /DEPRESS SPACE BAR/<15><12> |
| 5393 | 031124 | 020123 | 050123 | 041501 | |
| 5394 | 031132 | 020105 | 040502 | 006522 | |
| 5395 | 031140 | 000012 | | | |
| 5396 | | | | | |
| 5397 | 031142 | 042504 | 051120 | 051505 | DKPD: .ASCIZ /DEPRESS KEYPAD KEYS,LEFT TO RIGHT, TOP TO BOTTOM/<15><12> |
| 5398 | 031150 | 020123 | 042513 | 050131 | |
| 5399 | 031156 | 042101 | 045440 | 054505 | |
| 5400 | 031164 | 026123 | 042514 | 052106 | |
| 5401 | 031172 | 052040 | 020117 | 044522 | |
| 5402 | 031200 | 044107 | 026124 | 052040 | |
| 5403 | 031206 | 050117 | 052040 | 020117 | |
| 5404 | 031214 | 047502 | 052124 | 046517 | |
| 5405 | 031222 | 005015 | 000 | | |
| 5406 | | | | | |
| 5407 | 031225 | 113 | 054505 | 047502 | DKBERR: .ASCII /KEYBOARD ERROR,KEY POSITION IN ROW SHOULD BE / |
| 5408 | 031232 | 051101 | 020104 | 051105 | |
| 5409 | 031240 | 047522 | 026122 | 042513 | |
| 5410 | 031246 | 020131 | 047520 | 044523 | |
| 5411 | 031254 | 044524 | 047117 | 044440 | |
| 5412 | 031262 | 020116 | 047522 | 020127 | |
| 5413 | 031270 | 044123 | 052517 | 042114 | |
| 5414 | 031276 | 041040 | 020105 | | |
| 5415 | 031302 | 020040 | 005015 | | KYSTRK: .ASCII / /<15><12> |
| 5416 | 031306 | 041517 | 040524 | 020114 | .ASCIZ /OCTAL GD, OCTAL BAD/<15><12> |
| 5417 | 031314 | 042107 | 020054 | 041517 | |
| 5418 | 031322 | 040524 | 020114 | 040502 | |
| 5419 | 031330 | 006504 | 000012 | | |
| 5420 | 031334 | 020040 | 020040 | 020040 | DSPC6: .ASCIZ / / |
| 5421 | 031342 | 000 | | | |
| 5422 | | | | | |
| 5423 | 031343 | 036 | 076 | 020 | ROW1: .BYTE 36,76,20,13,32,12,54,44,14,41,71,57,63,64,3,114,0 |
| 5424 | 031346 | 013 | 032 | 012 | |
| 5425 | 031351 | 054 | 047 | 014 | |
| 5426 | 031354 | 041 | 071 | 057 | |
| 5427 | 031357 | 063 | 064 | 003 | |
| 5428 | 031362 | 114 | 000 | | |
| 5429 | | | | | |

| | | | | | | | |
|------|--------|--------|-----|-----|--------|----------------|---|
| 5430 | 031364 | 026 | 056 | 030 | ROW2: | .BYTE | 26,56,30,73,52,22,55,34,24,31,51,77,62,61,2,0 |
| 5431 | 031367 | 073 | 052 | 022 | | | |
| 5432 | 031372 | 055 | 034 | 024 | | | |
| 5433 | 031375 | 031 | 051 | 077 | | | |
| 5434 | 031400 | 062 | 061 | 002 | | | |
| 5435 | 031403 | 000 | | | | | |
| 5436 | | | | | | | |
| 5437 | 031404 | 046 | 040 | 053 | ROW3: | .BYTE | 46,40,53,23,72,42,45,74,11,21,47,27,66,0 |
| 5438 | 031407 | 023 | 072 | 042 | | | |
| 5439 | 031412 | 045 | 074 | 011 | | | |
| 5440 | 031415 | 021 | 047 | 027 | | | |
| 5441 | 031420 | 066 | 000 | | | | |
| 5442 | | | | | | | |
| 5443 | 031422 | 016 | 070 | 060 | ROW4: | .BYTE | 16,70,60,50,33,43,25,35,75,65,37,115,67,0 |
| 5444 | 031425 | 050 | 033 | 043 | | | |
| 5445 | 031430 | 025 | 035 | 075 | | | |
| 5446 | 031433 | 065 | 037 | 115 | | | |
| 5447 | 031436 | 067 | 000 | | | | |
| 5448 | | | | | | | |
| 5449 | 031440 | 001 | 000 | | CNTRA: | .BYTE | 01,0 |
| 5450 | | | | | | | |
| 5451 | 031442 | 101 | 000 | | SHFTA: | .BYTE | 101,0 |
| 5452 | | | | | | | |
| 5453 | 031444 | 015 | 000 | | SPCB: | .BYTE | 15,0 |
| 5454 | | | | | | | |
| 5455 | 031446 | 113 | 004 | 103 | KYPD: | .BYTE | 113,04,103,104,1,112,101,102,6,7,106,100,5 |
| 5456 | 031451 | 104 | 001 | 112 | | | |
| 5457 | 031454 | 101 | 102 | 006 | | | |
| 5458 | 031457 | 007 | 106 | 100 | | | |
| 5459 | 031462 | 005 | | | | | |
| 5460 | 031463 | 010 | 105 | 107 | | .BYTE | 10,105,107,110,17,111,0 |
| 5461 | 031466 | 110 | 017 | 111 | | | |
| 5462 | 031471 | 000 | | | | | |
| 5463 | | | | | | | |
| 5464 | | | | | | | |
| 5465 | 031472 | 000500 | | | RCRLB: | .EVEN .BLKB | 500 ;RECEIVE CIRCULAR BUFFER |
| 5466 | | | | | | | |
| 5467 | 032172 | 000500 | | | TCRLB: | .BLKB | 500 ;TRANSMIT CIRCULAR BUFFER |
| 5468 | 032672 | 000000 | | | ABJFP: | .WORD | 0 |
| 5469 | 032674 | 000062 | | | ABBUF: | .BLKB | 50. |
| 5470 | 032756 | 000000 | | | | | |
| 5471 | | 000001 | | | | .END | |

| | | | | | | | | | | | | | | |
|---------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| ABBUF | 032674 | 1513 | 3415 | 3417 | 5469# | | | | | | | | | |
| ABSXT | 022104 | 4383# | | | | | | | | | | | | |
| ABUFP | 032672 | 1513* | 1799 | 1831 | 1834 | 1846 | 1849 | 2080 | 2081 | 2088 | 2686 | 2689 | 2693* | 3414* |
| | | 3415 | 3417* | 3418* | 3659* | 3660 | 3772 | 3776 | 4281* | 4283 | 4286 | 4287* | 4353 | 4358 |
| | | 4380 | 5468# | | | | | | | | | | | |
| AESCO | 016104 | 3487 | 3519# | | | | | | | | | | | |
| AESCP | 016076 | 3485 | 3516# | | | | | | | | | | | |
| ALEXT | 013076 | 2956 | 2961# | | | | | | | | | | | |
| ALWCNT | 002240 | 1308# | 1616 | 1885 | 2348 | 3330 | | | | | | | | |
| AOUT | 013670 | 3011 | 3087# | | | | | | | | | | | |
| ASESC | 015720 | 3421 | 3484# | | | | | | | | | | | |
| ASTRT | 003540 | 1516 | 1529# | 2602 | 2692 | 2772 | 2798 | 2835 | | | | | | |
| AUTO | 002274 | 795 | 1334# | | | | | | | | | | | |
| AUTOA | 002304 | 1336# | 1340 | 1437 | 2945 | | | | | | | | | |
| BASC3 | 004504 | 1762 | 1763# | | | | | | | | | | | |
| BBLUP | 014534 | 3144 | 3259# | | | | | | | | | | | |
| BCLR = | 000020 | 1053# | | | | | | | | | | | | |
| BDAD | 013624 | 3076 | 3078# | | | | | | | | | | | |
| BDEXT | 013066 | 2952 | 2959# | | | | | | | | | | | |
| BDEXTA | 013064 | 2953 | 2958# | | | | | | | | | | | |
| BEL | 002006 | 1099# | 1535 | 2708 | | | | | | | | | | |
| BINOCT | 020706 | 2724 | 4127# | 4296 | 4310 | 4316 | | | | | | | | |
| BIT0 = | 000001 | 760# | | | | | | | | | | | | |
| BIT00 = | 000001 | 750# | 760 | 1505 | 2593 | 3045 | 3420 | 4161 | | | | | | |
| BIT01 = | 000002 | 749# | 759 | 1508 | 1580 | 3420 | 3490 | | | | | | | |
| BIT02 = | 000004 | 748# | 758 | | | | | | | | | | | |
| BIT03 = | 000010 | 747# | 757 | 3420 | 3486 | 3496 | | | | | | | | |
| BIT04 = | 000020 | 746# | 756 | | | | | | | | | | | |
| BIT05 = | 000040 | 745# | 755 | | | | | | | | | | | |
| BIT06 = | 000100 | 744# | 754 | | | | | | | | | | | |
| BIT07 = | 000200 | 743# | 753 | 1924 | 3377 | | | | | | | | | |
| BIT08 = | 000400 | 742# | 752 | 4463 | | | | | | | | | | |
| BIT09 = | 001000 | 741# | 751 | 4471 | 4529 | | | | | | | | | |
| BIT1 = | 000002 | 759# | | | | | | | | | | | | |
| BIT10 = | 002000 | 740# | 4514 | | | | | | | | | | | |
| BIT11 = | 004000 | 739# | 4478 | | | | | | | | | | | |
| BIT12 = | 010000 | 738# | 2647 | | | | | | | | | | | |
| BIT13 = | 020000 | 737# | 4521 | | | | | | | | | | | |
| BIT14 = | 040000 | 736# | 4449 | | | | | | | | | | | |
| BIT15 = | 100000 | 735# | | | | | | | | | | | | |
| BIT2 = | 000004 | 758# | | | | | | | | | | | | |
| BIT3 = | 000010 | 757# | | | | | | | | | | | | |
| BIT4 = | 000020 | 756# | | | | | | | | | | | | |
| BIT5 = | 000040 | 755# | | | | | | | | | | | | |
| BIT6 = | 000100 | 754# | | | | | | | | | | | | |
| BIT7 = | 000200 | 753# | | | | | | | | | | | | |
| BIT8 = | 000400 | 752# | | | | | | | | | | | | |
| BIT9 = | 001000 | 751# | | | | | | | | | | | | |
| BLDADA | 002356 | 1358# | 1380 | | | | | | | | | | | |
| BLDADD | 002354 | 1356# | 1363 | 2960 | | | | | | | | | | |
| BLDINA | 020522 | 2763 | 4071# | 4078 | | | | | | | | | | |
| BLDINC | 020516 | 1731 | 1772 | 2016 | 2224 | 2568 | 4070# | 4077 | 4089 | | | | | |
| BLDLNA | 002476 | 1383# | 1388 | | | | | | | | | | | |
| BLDLNE | 002466 | 1374 | 1377 | 1381# | | | | | | | | | | |
| BLDTST | 002542 | 1397# | | | | | | | | | | | | |
| BLKM | 002264 | 1318# | 1463* | 1764* | 2010* | 2055* | 2386* | 2569* | 3574 | 3577* | 3582* | 3609 | 3754* | 3930* |

| | | | | | | | | | | | | | | |
|----------|---------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| | | 2336* | 2454* | 2505* | 2551* | 2676* | 2746* | 2790* | 2814* | 2896* | 4473* | 4488* | 4493 | 4495 |
| \$LPERR | 001110 | 810# | 2897* | 4473 | 4489* | 4495 | 4531 | | | | | | | |
| \$MAIL = | ***** U | 2914 | 4488 | 4526 | 4564 | | | | | | | | | |
| \$MXCNT | 022552 | 4486 | 4495# | | | | | | | | | | | |
| \$NULL | 001154 | 830# | 4582 | 4611 | | | | | | | | | | |
| \$NWTST= | 000000 | 1530# | 1631# | 1677# | 1723# | 1759# | 1815# | 1866# | 2003# | 2119# | 2175# | 2215# | 2274# | 2333# |
| | | 2451# | 2502# | 2548# | 2673# | 2743# | 2787# | 2811# | | | | | | |
| \$SOCNT | 023540 | 4699* | 4728* | 4741# | | | | | | | | | | |
| \$SOMODE | 023542 | 4694* | 4698* | 4703 | 4706* | 4717* | 4743# | | | | | | | |
| \$OVER | 022536 | 4450 | 4466 | 4474 | 4484 | 4492# | | | | | | | | |
| \$PASS | 001100 | 805# | 1003 | 1007 | 1011 | 2618* | 2619* | 2627 | 2659 | 4480 | 4496 | | | |
| \$POWER | 024150 | 4846 | 4853# | | | | | | | | | | | |
| \$PWRDN | 023770 | 2876 | 4815# | 4843 | | | | | | | | | | |
| \$PWRMG | 024124 | 4846# | | | | | | | | | | | | |
| \$PWRUP | 024042 | 4825 | 4831# | | | | | | | | | | | |
| \$QUES | 001170 | 837# | 4541 | 4611 | | | | | | | | | | |
| \$RDCHR= | ***** U | 4894 | | | | | | | | | | | | |
| \$RDDEC= | ***** U | 4894 | | | | | | | | | | | | |
| \$RDLIN= | ***** U | 4894 | | | | | | | | | | | | |
| \$RDOCT= | ***** U | 4894 | | | | | | | | | | | | |
| \$RTNAD | 011316 | 2658# | | | | | | | | | | | | |
| \$RTRN | 011312 | 2634 | 2653# | 2884 | 2886* | 2891* | | | | | | | | |
| \$R2A = | ***** U | 4894 | | | | | | | | | | | | |
| \$SAVRE= | ***** U | 4894 | | | | | | | | | | | | |
| \$SAVR6 | 024146 | 4824* | 4832 | 4833* | 4834* | 4852# | | | | | | | | |
| \$SCOPE | 022300 | 2870 | 3091 | 4447# | | | | | | | | | | |
| \$SETUP= | 000037 | 1329# | 2616 | 2869 | 2870 | 2872 | 2874 | 2876 | 2878 | 2879 | 2880 | 2882 | 2896 | 2918 |
| | | 2921 | 4448 | 4511 | 4529 | 4536 | 4848 | | | | | | | |
| \$STUP = | 177777 | 1329# | | | | | | | | | | | | |
| \$SVLAD | 022510 | 4458 | 4487# | | | | | | | | | | | |
| \$SVPC = | 000200 | 788# | 793 | | | | | | | | | | | |
| \$SWR = | 177400 | 644# | 653 | 657 | 658 | 659 | 660 | 661 | 662 | 663 | 664 | 834 | 835 | 836 |
| | | 1532 | 1633 | 1679 | 1725 | 1761 | 1817 | 1868 | 2005 | 2121 | 2177 | 2217 | 2276 | 2335 |
| | | 2453 | 2504 | 2550 | 2610 | 2617 | 2632 | 2645 | 2659 | 2675 | 2745 | 2789 | 2813 | 2879 |
| | | 2880 | 2882 | 2896 | 2897 | 4439 | 4440 | 4441 | 4442 | 4443 | 4449 | 4461 | 4463 | 4464 |
| | | 4467 | 4468 | 4469 | 4476 | 4477 | 4478 | 4489 | 4492 | 4495 | 4502 | 4503 | 4504 | 4505 |
| | | 4506 | 4514 | 4521 | 4526 | 4529 | 4541 | 4847 | | | | | | |
| \$SWRMK= | 000000 | 664 | 665 | 4443 | 4444 | 4465 | | | | | | | | |
| \$TBIT | 011320 | 2649* | 2659# | 2895* | 4848* | | | | | | | | | |
| \$TIMES | 001160 | 834# | 1532* | 1633* | 1679* | 1725* | 1761* | 1817* | 1868* | 2005* | 2121* | 2177* | 2217* | 2276* |
| | | 2335* | 2453* | 2504* | 2550* | 2617* | 2675* | 2745* | 2789* | 2813* | 2879* | 4476* | 4483 | 4486* |
| | | 4495 | | | | | | | | | | | | |
| \$TKB | 001146 | 827# | 3316 | 3341 | | | | | | | | | | |
| \$TKS | 001144 | 826# | 3314 | 3339 | | | | | | | | | | |
| \$TN = | 000025 | 644# | 653 | 1530 | 1532# | 1631 | 1633# | 1677 | 1679# | 1723 | 1725# | 1759 | 1761# | 1815 |
| | | 1817# | 1866 | 1868# | 2003 | 2005# | 2119 | 2121# | 2175 | 2177# | 2215 | 2217# | 2274 | 2276# |
| | | 2333 | 2335# | 2451 | 2453# | 2502 | 2504# | 2548 | 2550# | 2673 | 2675# | 2743 | 2745# | 2787 |
| | | 2789# | 2811 | 2813# | | | | | | | | | | |
| \$TPB | 001152 | 829# | 4600* | 4611 | | | | | | | | | | |
| \$TPFLG | 001157 | 833# | 4558 | 4611 | | | | | | | | | | |
| \$TPS | 001150 | 828# | 4598 | 4611 | | | | | | | | | | |
| \$TRAP | 024160 | 2874 | 4864# | | | | | | | | | | | |
| \$TRAP2 | 024202 | 4875# | 4886 | | | | | | | | | | | |
| \$TRP = | 000006 | 4879# | 4888# | 4889# | 4890# | 4891# | 4892# | | | | | | | |
| \$TRPAD | 024214 | 4869 | 4886# | | | | | | | | | | | |
| \$TSTNM | 001102 | 806# | 1537 | 2616* | 3630 | 4438 | 4465 | 4487* | 4492 | 4496 | 4513 | 4541 | | |

| | | | | | | | | | | | | | | | | | | | | |
|------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|--|--|--|--|--|--|--|
| \$TYPBN= ***** U | 4892 | | | | | | | | | | | | | | | | | | | |
| \$TYPDS 023544 | 4756# | 4891 | | | | | | | | | | | | | | | | | | |
| \$TYPE 022722 | 4558# | 4879 | 4887 | | | | | | | | | | | | | | | | | |
| \$TYPEC 023072 | 4579 | 4586 | 4593 | 4598# | 4599 | | | | | | | | | | | | | | | |
| \$TYPEX 023140 | 4604 | 4606 | 4609# | | | | | | | | | | | | | | | | | |
| \$TYPOC 023342 | 4697# | 4888 | | | | | | | | | | | | | | | | | | |
| \$TYPON 023356 | 4696 | 4699# | 4890 | | | | | | | | | | | | | | | | | |
| \$TYPOS 023316 | 4692# | 4889 | | | | | | | | | | | | | | | | | | |
| \$XTSTR 022314 | 4452# | | | | | | | | | | | | | | | | | | | |
| \$SGET4= 000001 | 2632# | 2637 | | | | | | | | | | | | | | | | | | |
| \$OFILL 023541 | 4693* | 4697* | 4707 | 4742# | | | | | | | | | | | | | | | | |
| \$40CAT= ***** U | 4449 | 4523 | | | | | | | | | | | | | | | | | | |
| . = 032760 | 777# | 781# | 788 | 789# | 791# | 793# | 794# | 803# | 840 | 1018# | 1066# | 1067# | 1068# | | | | | | | |
| | 1069# | 1565 | 1574 | 1977 | 2092 | 2659 | 2664 | 2769 | 2867 | 2896 | 2897 | 2923# | 3022 | | | | | | | |
| | 3292 | 3299 | 3302 | 3315 | 4050 | 4094 | 4173 | 4244 | 4284 | 4495 | 4496 | 4541 | 4611 | | | | | | | |
| | 4666# | 4810# | 4827 | 4851 | 5037# | 5145# | 5465# | 5467# | 5469# | | | | | | | | | | | |
| .BEL = 000007 | 1098# | 2133 | 2286 | 2461 | 4261 | | | | | | | | | | | | | | | |
| .CARRT= 000015 | 1100# | | | | | | | | | | | | | | | | | | | |
| .CDWN = 000102 | 1116# | 2127 | 2236 | 2282 | | | | | | | | | | | | | | | | |
| .CHOM = 000110 | 1110# | 2233 | 2286 | 2304 | 2513 | | | | | | | | | | | | | | | |
| .CLFT = 000104 | 1119# | 2130 | 2282 | | | | | | | | | | | | | | | | | |
| .CLRCK= 000133 | 1147# | 2103 | | | | | | | | | | | | | | | | | | |
| .CLTCK= 000134 | 1150# | 2058 | | | | | | | | | | | | | | | | | | |
| .CPYSC= 000135 | 1232# | 2596 | | | | | | | | | | | | | | | | | | |
| .CRT = 000103 | 1113# | 2127 | | | | | | | | | | | | | | | | | | |
| .CUP = 000101 | 1122# | 2133 | | | | | | | | | | | | | | | | | | |
| .COO = 000040 | 1283# | 2282 | | | | | | | | | | | | | | | | | | |
| .CO3 = 000043 | 1284# | | | | | | | | | | | | | | | | | | | |
| .CO8 = 000050 | 1285# | | | | | | | | | | | | | | | | | | | |
| .C11 = 000053 | 1286# | | | | | | | | | | | | | | | | | | | |
| .C20 = 000064 | 1287# | 2459 | | | | | | | | | | | | | | | | | | |
| .C21 = 000065 | 1288# | | | | | | | | | | | | | | | | | | | |
| .C40 = 000110 | 1289# | | | | | | | | | | | | | | | | | | | |
| .C79 = 000157 | 1290# | 2301 | | | | | | | | | | | | | | | | | | |
| .DAPNT= 000171 | 1168# | | | | | | | | | | | | | | | | | | | |
| .DCRAD= 054433 | 1181# | | | | | | | | | | | | | | | | | | | |
| .DEMP = 000152 | 1156# | | | | | | | | | | | | | | | | | | | |
| .DISAC= 000137 | 1234# | | | | | | | | | | | | | | | | | | | |
| .DMAIN= 000141 | 1136# | 2731 | 2840 | | | | | | | | | | | | | | | | | |
| .DPNT = 000130 | 1229# | | | | | | | | | | | | | | | | | | | |
| .DRECT= 000103 | 1144# | 2103 | | | | | | | | | | | | | | | | | | |
| .EAPNT= 000131 | 1166# | | | | | | | | | | | | | | | | | | | |
| .EEMP = 000112 | 1154# | 1985 | | | | | | | | | | | | | | | | | | |
| .EINST= 000111 | 1171# | 2255 | 2258 | | | | | | | | | | | | | | | | | |
| .EMAIN= 000101 | 1134# | | | | | | | | | | | | | | | | | | | |
| .ENAC = 000136 | 1233# | | | | | | | | | | | | | | | | | | | |
| .EOS = 000112 | 1125# | 2513 | | | | | | | | | | | | | | | | | | |
| .EPNT = 000127 | 1227# | | | | | | | | | | | | | | | | | | | |
| .ERPL = 000151 | 1173# | 2255 | | | | | | | | | | | | | | | | | | |
| .ESC = 000033 | 1242# | 1985 | 2058 | 2103 | 2127 | 2130 | 2133 | 2233 | 2236 | 2258 | 2282 | 2286 | 2301 | | | | | | | |
| | 2304 | 2459 | 2461 | 2513 | 2731 | 2840 | 4259 | 4261 | | | | | | | | | | | | |
| .ESCZ = 055033 | 1258# | | | | | | | | | | | | | | | | | | | |
| .IABT = 000137 | 1159# | 4259 | | | | | | | | | | | | | | | | | | |
| .LKKB = 000105 | 1139# | | | | | | | | | | | | | | | | | | | |
| .LNFED= 000012 | 1102# | 2286 | 2461 | | | | | | | | | | | | | | | | | |
| .O = 000117 | 1193# | 1985 | 2058 | 2103 | 2127 | 2130 | 2133 | 2233 | 2236 | 2286 | 2301 | 2304 | 2461 | | | | | | | |

| | | | | | | |
|----------------|-------|------|------|------|------|------|
| .P = 000120 | 2513 | 2731 | 2840 | 4259 | 4261 | |
| .PSCRN= 000150 | 1243# | 2258 | | | | |
| .RABT = 000140 | 1235# | | | | | |
| .RDCUR= 000131 | 1215# | 4259 | | | | |
| .RESET= 000122 | 1189# | 2127 | 2130 | 2133 | 2301 | 2461 |
| .R00 = 000040 | 1260# | | | | | |
| .R00C1= 025440 | 1278# | 2459 | | | | |
| .R00C2= 032040 | 1270# | | | | | |
| .R01 = 000041 | 1272# | | | | | |
| .R12 = 000054 | 1279# | | | | | |
| .R22 = 000066 | 1280# | | | | | |
| .R23 = 000067 | 1281# | | | | | |
| .R23C0= 020067 | 1282# | 2282 | 2301 | | | |
| .R23C7= 067467 | 1268# | | | | | |
| .TAB = 000011 | 1183# | | | | | |
| .TCUCH= 000127 | 1104# | | | | | |
| .TSTER= 000124 | 1200# | 2233 | 2236 | 2286 | 2304 | |
| .TXRCK= 000135 | 1245# | | | | | |
| .TXTCK= 000136 | 1205# | | | | | |
| .UNLKK= 000145 | 1210# | 2058 | | | | |
| .XMTAL= 000126 | 1141# | 4261 | | | | |
| .Y = 000131 | 1195# | 2058 | 2513 | | | |
| | 1188# | 2282 | 2301 | 2459 | | |

| | | | | | | | | | | | | | | | |
|----------|-------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| \$\$ESCA | 775# | | | | | | | | | | | | | | |
| \$\$NEWT | 775# | 1530 | 1631 | 1677 | 1723 | 1759 | 1815 | 1866 | 2003 | 2119 | 2175 | 2215 | 2274 | 2333 | 2451 |
| | 2502 | 2548 | 2673 | 2743 | 2787 | 2811 | | | | | | | | | |
| \$\$SET | 4879# | 4888 | 4889 | 4890 | 4891 | | | | | | | | | | |
| \$\$SETU | 2882# | | | | | | | | | | | | | | |
| \$\$SKIP | 775# | | | | | | | | | | | | | | |
| .EQUAT | 644# | 665 | | | | | | | | | | | | | |
| .HEADE | 644# | | | | | | | | | | | | | | |
| .SETUP | 644# | 1329 | | | | | | | | | | | | | |
| .SWRHI | 644# | 653 | | | | | | | | | | | | | |
| .SWRLO | 644# | 665# | | | | | | | | | | | | | |
| .SACT1 | 644# | 784 | | | | | | | | | | | | | |
| .SCATC | 644# | 775 | | | | | | | | | | | | | |
| .SCMTA | 644# | 797 | | | | | | | | | | | | | |
| .SEOP | 644# | 2604 | | | | | | | | | | | | | |
| .SERRO | 644# | 4496 | | | | | | | | | | | | | |
| .SERRT | 644# | 4611 | | | | | | | | | | | | | |
| .SPOWE | 644# | 4811 | | | | | | | | | | | | | |
| .\$SAVE | 644# | | | | | | | | | | | | | | |
| .\$SCOP | 644# | 4433 | | | | | | | | | | | | | |
| .\$TRAP | 644# | 4856 | | | | | | | | | | | | | |
| .\$TYPD | 644# | 4744 | | | | | | | | | | | | | |
| .\$TYPE | 644# | 4541 | | | | | | | | | | | | | |
| .\$TYPO | 644# | 4667 | | | | | | | | | | | | | |

. ABS. 032760 000

ERRORS DETECTED: 0

CZVTJB.BIN,CZVTJB.LST/CRF/SOL/NL:TOC=CZVTJB.P11

RUN-TIME: 69 49 5 SECONDS

RUN-TIME RATIO: 300/124=2.4

CORE USED: 21K (41 PAGES)