

VT61

ACCEPTANCE TEST
CZVTMA0

AH-A930A-MC
COPYRIGHT © 1978
FICHE 1 OF 1

JAN 1979
digital
MADE IN USA

000000

.NLIST CND
.REPT 0

IDENTIFICATION

PRODUCT CODE: AC-A929A-MC
PRODUCT NAME: CZVTMA0 VT61 ACCPT TST
DATE: DEC 1978
MAINTENANCE: DIAGNOSTIC GROUP
AUTHOR: PAUL NELSON

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER
UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED
(WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH
SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL

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

COPYRIGHT (C) 1978 BY DIGITAL EQUIPMENT CORPORATION.

48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70

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

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 AN EOM.

THERE ARE TWO DISTINCT MODES IN WHICH THE PROGRAM CAN BE OPERATED. IN 'AUTO' MODE UP TO 2 DZ11'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 DZ11(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 DZ11(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 ADDRESS OF THE DZ11 TO BE TESTED (FORMAT IS 17XXXX, ONLY ONE ENTRY). 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. NOTE THAT IF AN ERROR IS MADE AND A ? IS TYPED, ALL DATA ENTERED IN RESPONSE TO THE QUESTION UP TO THAT POINT IS LOST. 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 THE ADDRESS ENTERED.

SIMPLY DEPRESSING A CARRIAGE RETURN WHEN UNIT ADDRESSES ARE REQUESTED WILL RESULT IN THE MAPPING AND TESTING OF ALL GOOD DZ11(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.

THE DEFAULT BAUD RATE IS 9600. IF THE OPERATOR WISHES TO CHANGE THE BAUD RATE, TYPE THE CODE FOR THE NEW BAUD RATE WHEN THE PROGRAM ASKS FOR IT DURING START-UP. TO KEEP THE DEFAULT, MERELY TYPE A CARRIAGE RETURN. IT IS RECOMMENDED THAT 9600 BAUD BE USED SINCE THE TESTS MAY TAKE A GREAT DEAL OF TIME TO RUN AT A LOWER RATE.

VALID BAUD RATE CODES

CODE=RATE	CODE-RATE
0-DEFAULT	10=1800
1-75	11=2000
2-110	12=2400
3=134.5	13=3600
4 150	14=4800
5 300	15=7200
6-600	16-9600
7-1200	17-19,200

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/RECEIVE 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 DZ11'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 (DZ11 WITH VT61 ATTACHED) AT NON-STANDARD ADDRESSES THE LOCATION 'STRTAB' CAN BE MODIFIED TO CONTAIN THE LOWEST OF THE NON-STANDARD ADDRESSES AND LOCATON '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-NUL 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

DO NOT LOOP ON THESE 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

MAINDEC-11-DZVTM-A MACY11 30A(1052) 07-NOV-78 15:00 PAGE 13-1
CZVTMA.P11 23-FEB-78 16:29

SEQ 0014

637

CAUSE A TEST EXIT.

MV
CZ

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 SHIFT AND CONTROL FUNCTIONS. THESE TESTS ARE EXPLICITELY DEFINED BY MESSAGES TO THE OPERATOR. 10 ERRORS WILL CAUSE AN AUTOMATIC EXIT FROM TEST.

.FNDR

639
640
641
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

```

.NLIST MD,MC,CND
.LIST ME
.TITLE MAINDEC-11-DZVTM-A
;*COPYRIGHT (C) 1978
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY TERMINAL DIAGNOSTICS
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
;*
.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

```

(1)	000000	PR0=	0	::PRIORITY LEVEL 0
(1)	000040	PR1=	40	::PRIORITY LEVEL 1
(1)	000100	PR2=	100	::PRIORITY LEVEL 2
(1)	000140	PR3=	140	::PRIORITY LEVEL 3
(1)	000200	PR4=	200	::PRIORITY LEVEL 4
(1)	000240	PR5=	240	::PRIORITY LEVEL 5
(1)	000300	PR6=	300	::PRIORITY LEVEL 6
(1)	000340	PR7=	340	::PRIORITY LEVEL 7

(1) :*'SWITCH REGISTER' SWITCH DEFINITIONS

(1)	100000	SW15=	100000
(1)	040000	SW14=	40000
(1)	020000	SW13=	20000
(1)	010000	SW12=	10000
(1)	004000	SW11=	4000
(1)	002000	SW10=	2000
(1)	001000	SW09=	1000
(1)	000400	SW08=	400
(1)	000200	SW07=	200
(1)	000100	SW06=	100
(1)	000040	SW05=	40
(1)	000020	SW04=	20
(1)	000010	SW03=	10
(1)	000004	SW02=	4
(1)	000002	SW01=	2
(1)	000001	SW00=	1
(1)		.EQUIV	SW09,SW9
(1)		.EQUIV	SW08,SW8
(1)		.EQUIV	SW07,SW7
(1)		.EQUIV	SW06,SW6
(1)		.EQUIV	SW05,SW5
(1)		.EQUIV	SW04,SW4
(1)		.EQUIV	SW03,SW3
(1)		.EQUIV	SW02,SW2
(1)		.EQUIV	SW01,SW1
(1)		.EQUIV	SW00,SW0

(1) :*DATA BIT DEFINITIONS (BIT00 TO BIT15)

(1)	100000	BIT15=	100000
(1)	040000	BIT14=	40000
(1)	020000	BIT13=	20000
(1)	010000	BIT12=	10000
(1)	004000	BIT11=	4000
(1)	002000	BIT10=	2000
(1)	001000	BIT09=	1000
(1)	000400	BIT08=	400
(1)	000200	BIT07=	200
(1)	000100	BIT06=	100
(1)	000040	BIT05=	40
(1)	000020	BIT04=	20
(1)	000010	BIT03=	10
(1)	000004	BIT02=	4
(1)	000002	BIT01=	2
(1)	000001	BIT00=	1
(1)		.EQUIV	BIT09,BIT9
(1)		.EQUIV	BIT08,BIT8

```

(1)      .EQUIV BIT07,BIT7
(1)      .EQUIV BIT06,BIT6
(1)      .EQUIV BIT05,BIT5
(1)      .EQUIV BIT04,BIT4
(1)      .EQUIV BIT03,BIT3
(1)      .EQUIV BIT02,BIT2
(1)      .EQUIV BIT01,BIT1
(1)      .EQUIV BIT00,BIT0
(1)
(1)      ;*BASIC 'CPU' TRAP VECTOR ADDRESSES
(1)      000004 ERRVEC= 4      ;;TIME OUT AND OTHER ERRORS
(1)      000010 RESVEC= 10     ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)      000014 TBITVEC=14     ;;'T' BIT
(1)      000014 TRTVEC= 14     ;;TRACE TRAP
(1)      000014 BPTVEC= 14     ;;BREAKPOINT TRAP (BPT)
(1)      000020 IOTVEC= 20     ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)      000024 PWRVEC= 24     ;;POWER FAIL
(1)      000030 EMTVEC= 30     ;;EMULATOR TRAP (EMT) **ERROR**
(1)      000034 TRAPVEC=34     ;;'TRAP' TRAP
(1)      000060 TKVEC= 60      ;;TTY KEYBOARD VECTOR
(1)      000064 TPVEC= 64      ;;TTY PRINTER VECTOR
(1)      000240 PIRQVEC=240    ;;PROGRAM INTERRUPT REQUEST VECTOR
689      .SBTTL TRAP CATCHER
(1)
(1)      000000      .=0
(1)      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
(1)      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
(1)      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
(1)      000174      .=174
(1)      000174 000000 DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
(1)      000176 000000 SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
690      .SBTTL ACT11 HOOKS
(1)
(2)      ;*****
(1)      ;HOOKS REQUIRED BY ACT11
(1)      000200      $SVPC=.      ;SAVE PC
(1)      000046      .-46
(1)      000046 011742 $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
(1)      000052      .=52
(1)      000052 000000 .WORD 0      ;;2)SET LOC.52 TO ZERO
(1)      000200      .=$SVPC      ;; RESTORE PC
691      000200      .-200
692      000200 000137 002426 START: JMP AUTO      ;USE AUTO SELECTION OF UNITS
693      000204 000137 002476 MSTRT: JMP MANS     ;ALLOW OPERATOR SELECTION OF UNITS/TESTS

```

```

694      .SBTTL  COMMON TAGS
(1)
(2)      ;*****
(1)      ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(1)      ;*USED IN THE PROGRAM.
(1)
(1)      001100      . =1100
(1) 001100 000000 $CMTAG:      ;; START OF COMMON TAGS
(1) 001102 000    $PASS: .WORD 0      ;; CONTAINS PASS COUNT
(1) 001103 000    $TSTNM: .BYTE 0      ;; CONTAINS THE TEST NUMBER
(1) 001104 000000 $ERFLG: .BYTE 0      ;; CONTAINS ERROR FLAG
(1) 001106 000000 $ICNT:  .WORD 0      ;; CONTAINS SUBTEST ITERATION COUNT
(1) 001110 000000 $LPADR: .WORD 0      ;; CONTAINS SCOPE LOOP ADDRESS
(1) 001112 000000 $LPERR: .WORD 0      ;; CONTAINS SCOPE RETURN FOR ERRORS
(1) 001114 000    $ERTTL: .WORD 0      ;; CONTAINS TOTAL ERRORS DETECTED
(1) 001115 001    $ITEMB: .BYTE 0      ;; CONTAINS ITEM CONTROL BYTE
(1) 001116 000000 $SERM: .BYTE 1      ;; CONTAINS MAX. ERRORS PER TEST
(1) 001120 000000 $ERRPC: .WORD 0      ;; CONTAINS PC OF LAST ERROR INSTRUCTION
(1) 001122 000000 $GDADR: .WORD 0      ;; CONTAINS ADDRESS OF 'GOOD' DATA
(1) 001124 000000 $BDADR: .WORD 0      ;; CONTAINS ADDRESS OF 'BAD' DATA
(1) 001126 000000 $GDDAT: .WORD 0      ;; CONTAINS 'GOOD' DATA
(1) 001130 000000 $BDDAT: .WORD 0      ;; CONTAINS 'BAD' DATA
(1) 001132 000000      .WORD 0      ;; RESERVED--NOT TO BE USED
(1) 001134 000    $AUTOB: .BYTE 0      ;; AUTOMATIC MODE INDICATOR
(1) 001135 000    $INTAG: .BYTE 0      ;; INTERRUPT MODE INDICATOR
(1) 001136 000000      .WORD 0
(1) 001140 177570 $SWR:  .WORD DSWR      ;; ADDRESS OF SWITCH REGISTER
(1) 001142 177570 $DISPLAY: .WORD DDISP      ;; ADDRESS OF DISPLAY REGISTER
(1) 001144 177560 $TKS: 177560      ;; TTY KBD STATUS
(1) 001146 177562 $TKB: 177562      ;; TTY KBD BUFFER
(1) 001150 177564 $TPS: 177564      ;; TTY PRINTER STATUS REG. ADDRESS
(1) 001152 177566 $TPB: 177566      ;; TTY PRINTER BUFFER REG. ADDRESS
(1) 001154 000    $NULL: .BYTE 0      ;; CONTAINS NULL CHARACTER FOR FILLS
(1) 001155 002    $FILLS: .BYTE 2      ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
(1) 001156 012    $FILLC: .BYTE 12      ;; INSERT FILL CHARS. AFTER A 'LINE FEED'
(1) 001157 000    $TPFLG: .BYTE 0      ;; 'TERMINAL AVAILABLE' FLAG (BIT<07>-0 YES)
(1) 001160 000000 $TIMES: 0      ;; MAX. NUMBER OF ITERATIONS
(1) 001162 000000 $ESCAPE: 0      ;; ESCAPE ON ERROR ADDRESS
(1) 001164 177607 $BELL: .ASCII <207><377><377>      ;; CODE FOR BELL
(1) 001170 077    $QUES: .ASCII /?/      ;; QUESTION MARK
(1) 001171 015    $CRLF: .ASCII <15>      ;; CARRIAGE RETURN
(1) 001172 000012 $LF:  .ASCII <12>      ;; LINE FEED
(2)      ;*****
  
```

000377

```

(1) .SBTTL ERROR POINTER TABLE
(1)
(1) ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1) ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1) ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1) ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1) ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1) ;*      EM      ;;POINTS TO THE ERROR MESSAGE
(1) ;*      DH      ;;POINTS TO THE DATA HEADER
(1) ;*      DT      ;;POINTS TO THE DATA
(1) ;*      DF      ;;POINTS TO THE DATA FORMAT
(1)
(1) $ERRTB:
695      ;GENERAL ESCAPE SEQUENCE ERROR MESSAGE
696
697 001174 026035      EM1      ;AN ESCAPE SEQUENCE TO VT61 FAILED.
698 001176 026122      DH1      ;TEST#,ERROR PC,2 SEQUENCE BYTES,2 SEQUENCE BYTES.
699 001200 001424      DT0
700 001202 001444      DF0
701
702      ;RECEIVE STATUS ERROR MESSAGE
703
704 001204 026165      EM2      ;RECEIVE STATUS ERROR
705 001206 026215      DH2      ;ADDRESS,STATUS ,ERR. BITS,CHAR.
706 001210 001454      DT2
707 001212 001444      DF0
708
709
710      ;RECIEVE SOFTWARE STATUS ERROR MESSAGE.
711
712 001214 026254      EM3      ;SOFTWARE (VSTAT) STATUS ERROR
713 001216 026315      DH3      ;PASS#,TEST#,GOOD ,STATUS,RECEIVED STATUS
714 001220 001502      DT4
715 001222 001545      DF6
716
717      ;DATA ERROR
718
719 001224 026364      EM4      ;DATA EXPECTED DOES NOT MATCH RECEIVE DATA.
720 001226 026430      DH4      ;TEST#,REC.CNT.,EXPECTED DATA, RECEIVE DATA
721 001230 001514      DT5
722 001232 001444      DF0
723
724      ;RECEIVE BYTE COUNT ERROR
725
726 001234 026476      EM5      ;BYTES EXPECTED DOES NOT EQUAL BYTES RECEIVED.
727 001236 026555      DH5      ;BYTES EXPECTED, BYTES RECEIVED
728 001240 001436      DT1
729 001242 001452      DF2
730
731      ;GENERAL DIRECT CURSOR ADDRESS FAILURE
732
733 001244 026606      EM6      ;CURSOR POSITION ERROR
734 001246 026641      DH6      ;GD LINE, GD COL., BD LINE, BAD COL.
735 001250 001454      DT2
  
```

736	001252	001476	DF3	
737				
738				;DIRECT CURSOR ADDRESS ERROR
739				
740	001254	026706	EM7	;DIRECT CURSOR ADDRESS ERROR
741	001256	027006	DH10	;PASS#,TEST#,BD. ROW,BD. COL.
742	001260	001502	DT4	
743	001262	001545	DF6	
744				
745				
746				;LAST TEST-SELF TEST FAILED
747				
748	001264	027210	EM10	;VT61 FAILED SELF-TEST FUNCTION
749	001266	027565	DH11	;CSR, VECTOR
750	001270	001436	DT1	
751	001272	001450	DF1	
752				
753				;VT61 FAIL/HUNG ERROR MESSAGE
754	001274	027045	EM11	;LAST TRANSMISSION TO VT61 CAUSED VT61 TO FAIL/HANG
755	001276	026751	DH7	;PASS#,TEST#,ERROR PC
756	001300	001466	DT3	
757	001302	001536	DF4	
758				
759				;GENERAL TEST FAILURE-PRECEEDS DATA/POSITION ERROR
760				
761	001304	027133	EM12	;VT61 UNDERR TEST FAILED-ERROR DATA FOLLOWS
762	001306	026751	DH7	;PASS#,TEST#,ERROR PC.
763	001310	001466	DT3	
764	001312	001536	DF4	
765				
766				;RECEIVE CHECKSUM ERROR
767				
768	001314	027364	EM13	;VT61 RECEIVER CHECKSUM ERROR
769	001316	027251	DH12	;PASS#,TEST#,GD.CKSUM,BD CKSUM
770	001320	001502	DT4	
771	001322	001545	DF6	
772				
773				;TRANSMITTER CHECKSUM ERROR
774				
775	001324	027433	EM14	;VT61 TRANSMITTER CHECKSUM ERROR
776	001326	027251	DH12	
777	001330	001502	DT4	
778	001332	001545	DF6	
779				
780				
781				
782				;XOFF FAILED TO HALT BLOCK XMIT
783				
784	001334	027660	EM15	;XOFF TO VT61 FAILED TO HALT BLOCK XMIT
785	001336	030375	DH13	;PASS,TEST,VSTAT
786	001340	001526	DT6	
787	001342	001536	DF4	
788				
789				;XON FAILED TO RESTART BLOCK XMIT
790				
791	001344	027731	EM16	;XON TO VT61 FAILED TO RESTART BLOCK XMIT

792	001346	030375				DH13	
793	001350	001526				DT6	
794	001352	001536				DF4	
795							
796							;NO XON AFTER UNIT WAS RESET
797							
798	001354	030004				EM17	;NO XON AFTER UNIT WAS RESET.
799	001356	026751				DH7	;PASS#,TEST#,ERROR PC
800	001360	001526				DT6	
801	001362	001536				DF4	
802							
803							;PERIPHERAL ABORT ERROR
804							
805	001364	030062				EM20	;LAST PERIPHERAL OPERATION ABORTED.
806	001366	030427				DH14	;PASS,TEST,ERROR PC, VSTAT
807	001370	001502				DT4	
808	001372	001545				DF6	
809							
810							;CANT CLEAR PERIPHERAL ABORT FLAG.
811							
812	001374	030126				EM21	;COULD NOT CLEAR LAST ABORT FLAG.
813	001376	030427				DH14	
814	001400	001502				DT4	
815	001402	001545				DF6	
816							
817							;MAINTENANCE MODE DID NOT FORCE A SOM/EOM.
818							
819	001404	030171				EM22	;SOM OR EOM NOT REC. IN MAINT. MODE.
820	001406	026315				DH3	;PASS#,TEST#,EXP.STAT, ACT.STAT
821	001410	001502				DT4	
822	001412	001545				DF6	
823							
824							;LINE FEED OR CURSOR RIGHT AT ROW 23 DID NOT CAUSE A SCROLL.
825							
826	001414	030257				EM23	;NO SCROLL FROM LINE FEED OR CURSOR RIGHT.
827	001416	026751				DH7	
828	001420	001526				DT6	
829	001422	001536				DF4	
830							
831	001424	002424	001116	001124	DT0:	.WORD	TSTNM,\$ERRPC,\$GDDAT,\$BDDAT,0
	001432	001126	000000				
832	001436	001124	001126	000000	DT1:	.WORD	\$GDDAT,\$BDDAT,0
833	001444	000	000	000	DF0:	.BYTE	0,0,0,0
	001447	000					
834	001450	000	000		DF1:	.BYTE	0,0
835							
836	001452	001	001		DF2:	.BYTE	1,1 ;DECIMAL TYPE
837							
838	001454	001120	001124	001122	DT2:	.WORD	\$GDADR,\$GDDAT,\$BDADR,\$BDDAT,0
	001462	001126	000000				
839	001466	001100	002424	001116	DT3:	.WORD	\$PASS,TSTNM,\$ERRPC,0
	001474	000000					
840	001476	001	001	001	DF3:	.BYTE	1,1,1,1
	001501	001					
841	001502	001100	002424	001124	DT4:	.WORD	\$PASS,TSTNM,\$GDDAT,\$BDDAT,0
	001510	001126	000000				

842	001514	002424	001120	001124	DT5:	.WORD	TSTNM,\$GDADR,\$GDDAT,\$BDDAT,0
	001522	001126	000000				
843	001526	001100	002424	001120	DT6:	.WORD	\$PASS,TSTNM,\$GDADR,0
	001534	000000					
844	001536	001	000	000	DF4:	.BYTE	1,0,0
845	001541	000	000	001	DF5:	.BYTE	0,0,1,1
	001544	001					
846	001545	001	000	000	DF6:	.BYTE	1,0,0,0
	001550	000					
847		001552				.EVEN	

; INSTRUCTION DEFINITIONS

848					POP2SP	=22626	
849		022626			PUSH2SP	=24646	
850		024646					
851							
852		000024			MAXTST	=24	
853							

; MAX. NO. OF TESTS

;; *****
 ;: DEFINITION SOFTWARE STATUS(VSTAT) REGISTER BITS
 ;; *****

858	100000	RXOFF	=100000	; SET FOR XOFF, CLEARED FOR XON
859	040000	RSOM	=040000	; SET FOR SOM (START OF MESSAGE).
860	020000	REOM	=020000	; SET FOR EOM (END OF MESSAGE).
861	010000	PABRT	=010000	; SET FOR A PERIPHERAL ABORT.
862	004000	RSTT	=004000	; SET FOR RECEIVE STATUS ERROR.
863	002000	CKSUM	=002000	; SET TO CALCULATE 61 REC. CHECKSUM
864	001000	EPL	=001000	; SET WHEN END OF LINE DETECTED
865	000400	FSC	=000400	; SET WHEN OCTAL 33 RECEIVED.
866	000200	XMKIL	=000200	; SET WHEN TRANSMIT KILLED.
867	000100	TXSUM	=000100	; SET TO CALCULATE 61 XMIT CHECKSUM
868	000040	REVID	=000040	; SET WHEN REVERSE VIDEO MODE RECEIVED.
869	000020	COMGP	000020	; SET TO CONVERT REC. CHAR. BY -137.
870	000010	ILLNE	=000010	; SET FOR REC. INT. ON NON-SELECTED LINE.
871	000004	CURPOS	=000004	; SET WHEN CURSOR POS. RECEIVED
872	000002	TRMID	=000002	; SET WHEN TERMINAL I.D. RECEIVED.
873	000001	XMDNE	000001	; SET UPON TRANSMIT COMPLETE

;; *****
 ;: DEFINITION OF DZ11 CONTROL BITS
 ;; *****

879	100000	TRDY	100000	; XMIT READY
880	040000	XENA	=040000	; XMIT INT. ENABLE
881	000040	XSCN	=000040	; XMIT SCAN ENABLE.
882	000200	RECDN	=000200	; RECEIVER DONE.
883	000100	RENA	=000100	; REC. INT. ENABLE.
884	000070	SCL	=000070	; STOP CODE = 2; CHAR. LENGTH = 8
885	010000	RXE	=010000	; RECEIVER ENABLE
886	000020	MCLR	=000020	; MASTER CLEAR.
887	000010	MAINT	=000010	; MAINTENANCE MODE.
888	000040	RSCN	=000040	
889	040010	TOMB	=040010	; MAINT. MODE AND XMIT INT. ENABLE.
890	000040	SCAN	=000040	; REC. AND XMIT SCAN ENABLES.
891	100000	RRDY	100000	; REG. 2, REC. BUFFER READY FLAG.
892				
893	003600	TOTCH	1920.	; TOTAL CHARACTERS ON SCREEN

```

894          003601      TOTC1  =1921.                ;TOTAL SCREEN +1
895                                     ;*****
896                                     ;FOLLOWING ARE DZ11 ADDRESS, VECTOR AND LINE STORAGE TABLES
897                                     ;*****
898 001552 000004      VVECT:  .BLKW  4                ;GOOD DZ11 VECTOR TABLE
899 001562 000004      DZTBL:  .BLKW  4                ;GOOD DZ11 ADDRESS TABLE
900 001572 000020      INTAB:  .BLKW 20                ;TABLE OF POSSIBLE DZ11 ADDRESSES
901 001632 000050      DZLNE:  .BLKW 40.              ;TABLE OF DZ11 LINES.
902 001752 000050      MZLNE:  .BLKW 40.              ;TABLE OF MODEM PRESENT FLAGS
903 002072 000000      SCRCH:  .WORD  0                ;SCRATCH LOCATION.
904 002074 000000      MODEM:  .WORD  0                ;MODEM PRESENT FLAG WORD
905
906                                     ;*****
907                                     ;CURRENT POINTERS FOR ADDRESSES, VECTORS AND LINES
908                                     ;*****
909 002076 000000      VECPT:  .WORD                    ;VECTOR INDEX
910 002100 000000      DZAPT:  .WORD                    ;ADDRESS INDEX
911 002102 000000      LNFTPT: .WORD                    ;DZ11 LINE POINTER.
912 002104 000000      MNEPT:  .WORD                    ;MODEM FLAG TABLE POINTER
913                                     ;*****
914 002106 160010      SIRTAB: .WORD 160010            ;ADDRESS TABLES FOR DZ11 INTERFACES
915 002110 164000      ENDTAB: .WORD 164000            ;BEGINNING OF FLOATING ADD.
916                                     ;*****
917                                     ;VT61 ADDRESSES IN TABLE REFLECT UNIT UNDER TEST
918                                     ;*****
919 002112 000000      VZCSR:  .WORD  0                ;DZ11 CONTROL AND STATUS.
920 002114 000000      VRBUF:  .WORD  0                ;RECEIVE DATA BUFFER
921 002116 000000      VXTCR:  .WORD  0                ;XMIT LINE CONTROL.
922 002120 000000      VXBUF:  .WORD  0                ;XMITTER DATA BUFFER
923 002122 000000      VECT:   .WORD  0                ;VECTOR FOR UNIT UNDER TEST
924 002124 000000      TSTLNE: .WORD  0                ;DZ11 LINE UNDER TEST.
925 002126 000000      OCTLNE: .WORD  0                ;OCT. EQUIV. OF TSTLNE (BIT8-11)
926 002130 000000      CRCSR:  .WORD  0                ;CONSOLE RECEIVE CSR
927 002132 000000      CRBUF:  .WORD  0                ;CONSOLE DATA BUFFER
928 002134 000000      BDRATE: .WORD  0                ;DZ BAUD RATE
929 002136 001000      X:      1000                    ;EXPERIMENTAL LOCATION
930 002140 000000      Y:      0                      ;EXPERIMENTAL LOCATION
931 002142 000000      NULL:   0                      ;NULL CHARACTER FLAG FOR TRANSMIT INTERRUPT ROUTINE
932
933                                     ;*****
934                                     ;TABLE OF VT61 COMMAND AND SEQUENCES
935                                     ;*****
936
937
938 000007      .BEL      =007
939 002144 000007      BEL:   .WORD  007                ;BELL
940 000015      .CARRT    =015
941 002146 000015      CARRT: .WORD  015                ;CARRIAGE RETURN
942 000012      .LNFED    =012
943 002150 000012      LNFED: .WORD  012                ;LINE FEED
944 000011      .TAB      =011
945 002152 000011      TAB:   .WORD  011                ;TAB
946
947 002154 000001      .WORD  01                      ;TABLE DELIMITER (ESCN)
948
949                                     ;*****

```

950		000110	.CHOM	=110		
951	002156	000110	CHOM:	.WORD	110	;HOME CURSOR H
952						
953		000103	.CRT	=103		
954	002160	000103	CRT:	.WORD	103	;CURSOR RIGHT C
955						
956		000102	.CDWN	=102		
957	002162	000102	CDWN:	.WORD	102	;CURSOR DOWN B
958						
959		000104	.CLFT	=104		
960	002164	000104	CLFT:	.WORD	104	;CURSOR LEFT D
961						
962		000101	.CUP	=101		
963	002166	000101	CUP:	.WORD	101	;CURSOR UP A
964						
965		000112	.EOS	=112		
966	002170	000112	EOS:	.WORD	112	;ERASE TO END OF SCREEN J
967						
968						
969			;*****			
970	002172	000002		.WORD	2	;TABLE DELIMITER (ESCO)
971			;*****			
972						
973						
974		000101	.EMAIN	=101		
975	002174	000101	EMAIN:	.WORD	101	;ENTER MAINTENANCE MODE A
976		000141	.DMAIN	=141		
977	002176	000141	DMAIN:	.WORD	141	;EXIT MAINTENANCE MODE SA
978						
979		000105	.LKKB	=105		
980	002200	000105	LKKB:	.WORD	105	;LOCK KEYBOARD E
981		000145	.UNLKKB	=145		
982	002202	000145	UNLKKB:	.WORD	145	;UNLOCK KEYBOARD SE
983						
984		000103	.DRECT	=103		
985	002204	000103	DRECT:	.WORD	103	;ENABLE LINEAR MODE C
986						
987		000133	.CLRCK	=133		
988	002206	000133	CLRCK:	.WORD	133	;CLEAR RECEIVER CHECKSUM [
989						
990		000134	.CLTCK	=134		
991	002210	000134	CLTCK:	.WORD	134	;CLEAR TRANSMITTER CHECKSUM
992						
993						
994		000112	.EEMP	=112		
995	002212	000112	EEMP:	.WORD	112	;ENABLE REVERSE VIDEO J
996		000152	.DEMP	=152		
997	002214	000152	DEMP:	.WORD	152	;DISABLE REVERSE VIDEO SJ
998						
999		000137	.IABT	=137		
1000	002216	000137	IABT:	.WORD	137	;INITIALIZE ABORT FLAG -
1001						
1002			;*****			
1003	002220	000003		.WORD	3	;TABLE DELIMITER (ESCAPE P)
1004			;*****			
1005						

1006		000131	.EAPNT	=131		
1007	002222	000131	EAPNT:	.WORD	131	;ENABLE AUTO PRINT MODE Y
1008		000171	.DAPNT	=171		
1009	002224	000171	DAPNT:	.WORD	171	;DISABLE AUTO PRINT MODE SY
1010						
1011		000111	.EINST	=111		
1012	002226	000111	EINST:	.WORD	111	;ENABLE INSERT I
1013		000151	.ERPL	=151		
1014	002230	000151	ERPL:	.WORD	151	;ENABLE REPLACE SI
1015						
1016						
1017						
1018	002232	000004				
1019			::*****			
1020				.WORD	4	;TABLE DELIMITER (I/O)
1021			::*****			
1021		054433	.DCRAD	=054433		
1022	002234	054433	DCRAD:	.WORD	054433	;DIRECT CURSOR ADDRESSING
1023		067467	.R23C79	=067467		
1024	002236	067467	R23C79:	.WORD	067467	;CURSOR TO LOWER RIGHT
1025	002240	000000		.WORD	0	
1026						
1027	002242	047433	RCUR:	.WORD	047433	;DIRECT CURSOR ADDRESSING
1028		000131	.Y	=131		
1029		000131	.RDCUR	=00131		
1030	002244	000131	RDCUR:	.WORD	00131	;READ CURSOR POSITION Y
1031	002246	000000		.WORD	0	
1032						
1033		000117	.O	=117		
1034	002250	047433	[ESC:	.WORD	047433	;ESCAPE O
1035		000126	.XMTAL	=000126		
1036	002252	000126	XMTAL:	.WORD	000126	;TRANSMIT ALL V
1037	002254	000000		.WORD	0	
1038						
1039	002256	047433		.WORD	047433	;ESCAPE O
1040		000127	.TUCH	=127		
1041	002260	000127	TUCH:	.WORD	127	;XMIT CHARACTER AT CURSOR. W
1042	002262	000000		.WORD	0	
1043						
1044	002264	047433		.WORD	047433	;ESCAPE O
1045		000135	.TXRCK	=135		
1046	002266	000135	TXRCK:	.WORD	135	;XMIT RECIEVER CHECKSUM]
1047	002270	000000		.WORD	0	
1048						
1049	002272	047433		.WORD	047433	;ESCAPE O
1050		000136	.TXTCK	=136		
1051	002274	000136	TXCK:	.WORD	136	;XMIT TRANSMITTER CHECKSUM
1052	002276	000000		.WORD	0	
1053						
1054	002300	147433		.WORD	147433	;ESCAPE O
1055		000140	.RABT	=140		
1056	002302	000140	RABT:	.WORD	140	;READ THE ABORT FLAG. \
1057	002304	000000		.WORD	0	
1058						
1059			::*****			
1060	002306	177777		.WORD	-1	;END OF TABLE TERMINATOR
1061			::*****			

```

1062
1063 ::*****
1064 ;PERIPHERAL COMMANDS
1065 ::*****
1066
1067 .EPNT =127
1068 002310 000127 EPNT: .WORD 127 ;ENABLE PRINT MODE. W
1069 000130 .DPNT =130
1070 002312 000130 DPNT: .WORD 130 ;DISABLE PRINT MODE X
1071
1072 .CPYSC =135 ;COPY SCREEN J
1073 000136 .ENAC =136 ;ENABLE AUTO COPY MODE ESC ^
1074 000137 .DISAC =137 ;DISABLE AUTO COPY MODE ESC -
1075 000150 .PSCRN =000150 ;PRINT THE SCREEN H/SH
1076
1077
1078 ::*****
1079 ;ESCAPE CODE EQUIVALENCES AND IDENTIFIERS
1080 ::*****
1081
1082 .ESC =033 ;PRIMARY ESCAPE CODE.
1083 .P =120
1084 002314 050033 ESCP: .WORD 050033 ;ESCAPE P
1085 000124 .TSTER =124
1086 002316 000124 TSTER: .WORD 124 ;TEST TERMINAL(ESC O T)
1087 002234 ESCYI =DCRAD ;ESCYI EQUALS DCRAD/DCRADI
1088 000057 SLSH =000057 ;SLASH CODE FOR TERMINAL IDENT ESC.
1089 000106 CKGP =106 ;ENABLE REC.TO SUB 137 FROM ALL REC DATA
1090 000107 NCKGP =107 ;ENABLE NORMAL RECEIVED DATA.
1091 000171 CPABRT =171 ;COPIER ABORT
1092 000172 PRABRT =172 ;PRINTER ABORT
1093 000170 NABRT =170 ;NO ABORT SX
1094 002320 000000 IDENT: .WORD 0 ;VT61 IDENT CODE
1095 002250 ESCOI =ESCO
1096 002314 ESCPI =ESCP
1097 002322 ESCZI =ESCZ
1098 055033 .ESCZ =055033
1099 002322 055033 ESCZ: .WORD 055033 ;OCTAL EQUIV. OF ESZ SEQUENCE
1100 000122 .RESET =122
1101 002324 000122 RESET: .WORD 122 ;VT61 INITIALIZE R
1102
1103 002326 000033 ESCN: .WORD 000033 ;ESCAPE N-FLAG
1104 002330 020041 R01C00: .WORD 020041 ;ROW1,COL. 0
1105 002332 032041 R01C20: .WORD 032041 ;ROW01,COLUMN 20
1106 002334 020066 R22C00: .WORD 020066 ;ROW22,COL.00
1107 002336 020054 R12C00: .WORD 020054 ;RCW 12,COLUMN 00
1108 020067 .R23C00 =020067
1109 002340 020067 R23C00: .WORD 020067 ;ROW23,COL.00
1110 025440 .R00C11 =025440
1111 002342 025440 R00C11: .WORD 025440 ;ROW,COL.11
1112 032040 .R00C20 =032040
1113 002344 032040 R00C20: .WORD 032040 ;ROW 0,COLUMN 20
1114 002346 024040 R00C08: .WORD 024040 ;ROW 00,COLUMN 8
1115 020040 CUHME: .WORD 020040 ;OCTAL EQUIV. OF CURSOR HOME.
1116 002352 067440 R00C80: .WORD 067440 ;ROW 0,COLUMN 80.
1117 002354 067067 R23C78: .WORD 067067 ;ROW 23,COL. 78.

```

1118	000040	.R00	=40	;ROW 0
1119	000041	.R01	=41	;ROW 1
1120	000054	.R12	=54	;ROW 12
1121	000066	.R22	=66	;ROW 22
1122	000067	.R23	=67	;ROW 23
1123	000040	.C00	=40	;COLUMN 0
1124	000043	.C03	=43	;COL. 3
1125	000050	.C08	=50	;COL. 8
1126	000053	.C11	=53	;COL. 11
1127	000064	.C20	=64	;COL. 20
1128	000065	.C21	=65	;COL. 21
1129	000110	.C40	=110	;COL. 40
1130	000157	.C79	=157	;COL. 79
1131				
1132		;*****		
1133		;TEMPORARY STORAGE LOCATIONS AND		
1134		;SPECIAL RECEIVE CODE EQUIVALENCES.		
1135		;*****		
1136	000002	SOM	=02	;START OF MESSAGE
1137	000004	EOM	=04	;END OF MESSAGE
1138	000023	XOFF	=23	;TURN OFF TRANSMISSION
1139	000021	XON	=21	;TURN ON TRANSMISSION
1140	002356	CHRD:	.WORD 0	;STORAGE FOR SINGLE CH. READ
1141	002360	SVER1:	.WORD	;TEMP. STORAGE R1.
1142	002362	SVER2:	.WORD	;TEMP. STORAGE R2.
1143	002364	ZERO:	.WORD 0	;MUST BE LEFT AS ZERO.
1144	002366	TYP6:	.WORD 3000	;TYPE 6 OCTAL CHAR-NO ZEROS
1145	002370	TSTPTR:	.WORD 0	;TEST POINTER IN MANUAL SELECT MODE
1146	002372	MODE:	.WORD 0	;BYTE0=TESTING MODE,BYTE1=INTERFACE TYPE
1147	002374	FTLCNT:	.WORD 0	;COUNT OF INCOMPLETE XMIT.
1148	002376	ALWCNT:	.WORD 10.	;# OF ALLOWABLE INCOMPLETE XMIT.
1149	002400	ONE:	.WORD 1	
1150	002402	TOADD:	.WORD	
1151	002404	BUBCT:	.WORD	
1152	002406	TPREG:	0	
1153	002410	PRESC:	.WORD	;PRIMARY ESC COMMAND
1154	002412	ESSEQ:	.WORD	;SEQUENCE ASSEMBLY AREA
1155	002414	DLAY:	.WORD	
1156	002416	ROSVE:	.WORD	;TEMP STORAGE FOR R0 ONLY.
1157	002420	VSTAT:	.WORD 0	
1158	002422	BLKM:	.WORD 0	;FLAG LOCATION FOR BLOCK MODE XMIT.
1159	002424	TSTNM:	.WORD 0	;DISPLAY STORAGE FOR TEST NUMBER.
1160				
1161				
1163		;*****		
1164		;AUTOMATIC SELECTION OF UNITS. TESTS 1 THROUGH 33 WILL BE		
1165		;REPITIVELY EXECUTED FOR ALL UNITS.		
1166		;*****		
1167				
1168	002426	012737	007000	002134 AUTO: MOV #7000,BDRATE ;SET DEFAULT BAUD RATE TO 9600
1169	002434	005037	001100	CLR \$PASS ;CLEAR PASS COUNT
1170	002440	005037	002372	CLR MODE ;ZERO THE MODE SWITCH
1171	002444	000137	013046	JMP SETA ;DO VECTOR SETUP
1172	002450	004037	003032	AUTOA: JSR R0,GBDRT ;SEE IF OPERATOR WANTS TO CHANGE BAUD
1173	002454	004037	013576	AUTOB: JSR R0,TRPVEC ;GO FIND GOOD DZ11S
1174	002460	004037	013704	JSR R0,CDEV ;CHECK DZ11S FOUND

```

1175 002464 004037 014416      JSR    R0,INITA      ;INSURE VT61S ON DZ11
1176 002470 000137 003102      JMP    MODCK        ;VT61 PRESENT -BEGIN TESTING
1177 002474 000767              BR      AUTOB          ;NO VT61 FOUND LOOP IN CHECKING
1178
1179      ;:*****
1180
1181      ;MANUAL UNIT AND TEST SELECTION.  UNITS CAN BE
1182      ;SELECTED VIA CONSOLE OR AUTO SELECTION CAN
1183      ;BE UTILIZED.  TESTS ENTERED VIA CONSOLE WILL
1184      ;BE EXECUTED IN THE ORDER ENTERED.
1185
1186      ;:*****
1187
1188 002476 012737 007000 002134  MANS:  MOV    #7000,BDRATE  ;SET DEFAULT BAUD RATE TO 9600
1189 002504 012737 000001 002372      MOV    #1,MODE      ;SET MODE TO MANUAL SELECT.
1190 002512 000137 013046              JMP    SETA          ;GO SET UP CONSTANTS
1191 002516 004037 003032      MANS:  JSR    R0,GBDRT      ;SEE IF OPERATOR WANTS TO CHANGE BAUD
1192 002522 104401 025521              TYPE    ,DMANA
1193 002526 004037 013576              JSR    R0,TRPVEC      ;FIND GOOD DZ11'S
1194 002532 012703 001572              MOV    #INTAB,R3
1195 002536 005002      BLDADD:  CLR    R2
1196
1197 002540 004037 022010      BLDADA:  JSR    R0,GTNUM      ;GET A KEYBOARD INPUT
1198 002544 120127 000054      CMPB    R1,#54      ;CHAR. = COMMA?
1199 002550 001004              BNE     1$          ;NO
1200 002552 004037 013532      JSR    R0,TMNA      ;YES-VERIFY THIS ADDRESS,
1201 002556 010223              MOV     R2,(R3)+      ;STORE THIS ADDRESS
1202 002560 000766              BR      BLDADD      ;AND LOOK FOR ANOTHER ADDRESS.
1203 002562 120137 002150      1$:   CMPB    R1,LFED      ;CHAR. = LINE FEED?
1204 002566 001025              BNE     3$          ;NO
1205 002570 005702              TST     R2          ;ANY ENTRIES CREATED?
1206 002572 001414              BEQ     2$          ;NO USE AUTO SELECTION OF UNITS
1207 002574 004037 013532      JSR    R0,TMNA      ;YES-VERIFY THIS ADDRESS,
1208 002600 010223              MOV     R2,(R3)+      ;STORE LAST ADDRESS,
1209 002602 013723 002364      MOV     ZERO,(R3)+  ;AND SET A TERMINATOR IN TABLE.
1210 002606 004037 013704      JSR    R0,CDEV      ;CHECK DZ11 ON VT 61 SELECTED
1211 002612 005737 001562      TST     DZTBL      ;ANY DZ11S GOOD?
1212 002616 001737              BEQ     MANSA      ;NO-BACK TO SQUARE ONE
1213 002620 000137 002650      JMP     BLDLNE      ;YES- GO GET TESTS
1214 002624 004037 013704      2$:   JSR    R0,CDEV      ;CHECK DZ11'S
1215 002630 004037 014416      JSR    R0,INITA      ;VERIFY DZ11 HAVE VT61 ATTACHED
1216 002634 000137 002650      JMP     BLDLNE      ;BEGIN LINE SELECTION
1217 002640 000726              BR      MANSA      ;NO UNIT FOUND-LOOP
1218 002642 004037 021706      3$:   JSR    R0,OCTBIN      ;KEEP BUILDING ADDRESS
1219 002646 000734              BR      BLDADA
1220 002650 104401 025651      BLDLNE:  TYPE    ,DMANL      ;TYPE ENTER , LINES MESSAGE.
1221 002654 012703 001632      MOV     #DZLNE,R3      ;SET FIRST LINE ADDRESS.
1222 002660 005002      BLDLNA:  CLR    R2
1223 002662 004037 022010      10$:  JSR    R0,GTNUM      ;GET A KEYBOARD INPUT
1224 002666 120127 000054      CMPB    R1,#54      ;CHAR. = COMMA?
1225 002672 001002              BNE     1$          ;NO
1226 002674 010223              MOV     R2,(R3)+      ;YES - STORE THIS ADDRESS
1227 002676 000770              BR      BLDLNA      ;AND LOOK FOR ANOTHER LINE ENTRY.
1228 002700 120137 002150      1$:   CMPB    R1,LFED      ;CHAR. = LINE FEED?
1229 002704 001403              BEQ     2$          ;YES-SET TERMINATIONS AND EXIT.
1230 002706 004037 021706      JSR    R0,OCTBIN      ;NO-KEEP BUILDING ADDRESS

```

```

1231 002712 000763
1232 002714 010223
1233 002716 012723 177777
1234 002722 005013
1235
1236 002724 104401 025621
1237 002730 012703 001572
1238 002734 005004
1239 002736 005002
1240 002740 004037 022010
1241 002744 120127 000054
1242 002750 001011
1243 002752 122702 000024
1244 002756 002422
1245 002760 110223
1246 002762 005204
1247 002764 020437 000040
1248 002770 001444
1249 002772 000761
1250 002774 120137 002150
1251 003000 001006
1252 003002 110223
1253 003004 105013
1254 003006 112737 000001 002372
1255 003014 000432
1256
1257 003016 004037 021706
1258 003022 000746
1259
1260 003024 104401 026031
1261 003030 000735
1262
1263
1264
1265
1266
1267
1268 003032 005002
1269 003034 104401 025746
1270 003040 004037 022010
1271 003044 123701 002150
1272 003050 001403
1273 003052 004037 021706
1274 003056 000770
1275 003060 005702
1276 003062 001406
1277 003064 020227 000017
1278 003070 003360
1279 003072 000302
1280 003074 010237 002134
1281 003100 000200
1282
1283
1284
1285
1286

2$: BR 10$
MOV R2,(R3)+ ;STORE LAST ADDRESS.
MOV #-1,(R3)+ ;STORE END OF ADD. TERMINATOR.
CLR (R3) ;STORE LAST LINE TERMINATOR

BLDTST: TYPE ,DMANB ;TYPE 2ND PART OF MANUAL MESSAGE
MOV #INTAB,R3 ;USE INTAB AS TEST # STORAGE.
CLR R4 ;CLEAR TEST COUNTER
11$: CLR R2 ;CLEAR ASSEMBL WORD
10$: JSR R0,GTNUM ;GET A NUMERIC CHAR.
CMPB R1,#54 ;CHAR.=COMMA?
BNE 1$ ;NO
CMPB #MAXTST,R2 ;GREATER THAN THE NUMBER OF AVAILABLE TESTS?
BLT 12$ ;BR IF YES
MOVB R2,(R3)+ ;YES STORE A TEST #
INC R4 ;AND INCREMENT TEST COUNT.
CMP R4,32. ;COUNT =32?
BEQ MODCK ;YES ACCEPT NO MORE ENTRIES.
BR 11$ ;NO KEEP LOOKING
1$: CMPB R1,LNFED ;CHAR. = LINE FEED?
BNE 2$ ;NO
MOVB R2,(R3)+ ;LOAD THE LAST TEST
CLRB (R3) ;AND INSERT TEST TABLE TERMINATOR
MOVB #1,MODE ;SET MODE SWITCH TO MANUAL
BR MODCK ;AND BEGIN TESTING.

2$: JSR R0,OCTBIN ;CONVERT CHAR.
BR 10$

12$: TYPE ,QUES
BR BLDTST ;GO BACK AND GET ANOTHER RESPONSE

;*****
;THIS ROUTINE ASKS FOR A BAUD RATE OTHER THAN DEFAULT.
;OPERATOR TYPES CR TO KEEP DEFAULT (9600)
;*****

GBDRT: CLR R2
TYPE ,BRATE ;ASK FOR BAUD RATE (9600 IS DEFAULT)
1$: JSR R0,GTNUM ;GET THE NUMBER
CMPB LNFED,R1 ;LINE FEED?
BEQ 2$ ;BR IF SO
JSR R0,OCTBIN ;CONVERT CHARACTER
BR 1$ ;GET ANOTHER
2$: TST R2 ;WAS A RATE SPECIFIED?
BEQ 3$ ;BR IF NOT
CMP R2,#17 ;SEE IF VALID BAUD RATE
BGT GBDRT ;BR IF NOT
SWAB R2 ;MOVE INTO HIGH BYTE
MOV R2,BDRATE ;STORE IT
3$: RTS R0

;*****
;THIS ROUTINE LOOKS FOR THE OPERATIONAL MODE REQUESTED AND
;SELECTS THE NEXT UNIT TO BE TESTED.

```

```

1287                                     ;MODE 0 = ACCEPTANCE TYPE TEST
1288                                     ;MODE 1 = OPERATOR SELECTION OF UNITS AND SEQUENCE OF TESTS.
1289                                     ;*****
1290 003102 012737 001562 002100 MODCK: MOV #DZTBL,DZAPT ;INITIAL SETUP OF ADDRESS
1291 003110 012737 001552 002076      MOV #VVECT,VECPT ;AND VECTOR POINTERS.
1292 003116 012737 001632 002102      MOV #DZLNE,LNEPT ;LOAD LINE POINTER.
1293 003124 012737 001752 002104      MOV #MZLNE,MNEPT ;LOAD MODEM FLAG POINTER
1294 003132 012701 002112      MODCO: MOV #VZCSR,R1 ;LOAD ADDRESS DESTINATION
1295 003136 013702 002100      MOV DZAPT,R2 ;LOAD CURRENT ADDRESS POINTER
1296 003142 017703 176730      MOV @VECT,R3 ;LOAD CURRENT VECTOR POINTER
1297 003146 005712      TST (R2) ;ALL UNITS CHECKED?
1298 003150 001013      BNE 1$ ;NO - CONTINUE
1299 003152 005737 002372      TST MODE ;CHECK MODE
1300 003156 001002      BNE 10$
1301 003160 000137 011636      JMP ENDP$ ;GO TO END OF PASS
1302 003164 105777 177200      10$: TSTB @TSTPTR ;MANUAL LOOP REQUESTED?
1303 003170 100001      BPL 2$ ;NO
1304 003172 000743      BR MODCK ;YES-RESTART COMPLETE TEST.
1305 003174 000137 002516      2$: JMP MANSA ;GO RESTART MANUAL MODE
1306 003200 004037 015172      1$: JSR R0,LDADD ;NO-LOAD NEXT ADDRESSES
1307 003204 010237 002100      MOV R2,DZAPT ;SAVE ADDRESS POINTER.
1308 003210 010337 002122      MOV R3,VECT ;STORE VECT. OF UNIT UNDER TEST
1309 003214 012723 016176      MOV #INTRC,(R3)+ ;YES - NOW SET UP RECEIVE VECTOR
1310 003220 012723 000340      MOV #340,(R3)+ ;AND SET RECEIVER PSW TO 7
1311 003224 012723 017164      MOV #INTXM,(R3)+ ;SET UP TRANSMIT VECTOR
1312 003230 012723 000340      MOV #340,(R3)+ ;AND SET PSW TO 7.
1313 003234 017737 176642 002124 MODCA: MOV @LNEPT,TSTLNE ;LOAD LINE TO UTILIZED.
1314 003242 017737 176636 002074      MOV @MNEPT,MODEM ;LOAD MODEM FLAG FOR LINE.
1315 003250 023727 002124 177777      CMP TSTLNE,#-1 ;THIS LINE REALLY A SEPARATOR?
1316 003256 001012      BNE 12$ ;NO-TEST IT.
1317 003260 062737 000002 002102      ADD #2,LNEPT ;YES-BUMP LINE POINTER,UPDATE VECTOR
1318 003266 062737 000002 002076      ADD #2,VECPT ; POINTER AND GET NEXT ADDRESS.
1319 003274 062737 000002 002104      ADD #2,MNEPT ;UPDATE MODEM FLAG POINTER
1320 003302 000713      BR MODCO
1321 003304 005046      12$: CLR -(SP) ;CLEAR THE PSW,LSI11 STYLE.
1322 003306 012746 003314      MOV #100$,-(SP)
1323 003312 000002      RTI
1324 003314 012737 033173 017126 100$: MOV #RCRLB+477,REBUF ;SET UP END OF BUFFER
1325 003322 012737 033673 017462      MOV #TCRLB+477,TEBUF
1326 003330 012737 032474 017124      MOV #RCRLB,RBBUF ;INITIALIZE REC.BUFFER.
1327 003336 012737 033174 017460      MOV #TCRLB,TBBUF ;INITIALIZE TRANSMIT BUFFER.
1328 003344 004037 020502      JSR R0,RESPTR ;RESET INTERRUPT POINTERS.
1329 003350 005037 002422      CLR BLKM ;CLEAR BLOCK MODE FLAG.
1330 003354 005037 023072      CLR XMZER ;CLEAR ZERO TRANSMIT FLAG
1331 003360 005037 002420      CLR VSTAT ;CLEAR ALL INTERRUPT FLAGS
1332 003364 004037 021730      JSR R0,CONVLN ;CONVERT BINARY LINE # TO OCTAL.
1333 003370 052777 000020 176514      BIS #MCLR,@VZCSR ;CLEAR SILO AND UARTS.
1334 003376 032777 000020 176506 110$: BIT #MCLR,@VZCSR ;WAIT FOR CLEAR OPERATION TO FINISH
1335 003404 001374      BNE 110$
1336 003406 013705 002124      MOV TSTLNE,R5 ;LOAD XMITTER CONTROL REG
1337 003412 005737 002074      TST MODEM ;WITH TRANSMIT ENABLE AND
1338 003416 100413      BMI 111$ ;DTR IF MODEM FLAG SET
1339 003420 000305      SWAB R5
1340 003422 063705 002124      ADD TSTLNE,R5
1341 003426 050577 176464      BIS R5,@VXTCR ;SET BOTH
1342 003432 042705 000377      BIC #377,R5
  
```

1343	003436	030577	176456		113\$:	BIT	R5,@VXBUF	;WAIT FOR CARRIER
1344	003442	001775				BEQ	113\$	
1345	003444	000402				BR	112\$	
1346	003446	050577	176444		111\$:	BIS	R5,@VXTCR	;SET TRANSMIT ENABLE ONLY
1347	003452	013737	002126	002072	112\$:	MOV	OCTLNE,SCRTCH	;GET THE LINE NUMBER.
1348	003460	063737	002134	002072		ADD	BDRATE,SCRTCH	;ADD IN SELECTED BAUD RATE
1349	003466	062737	010070	002072		ADD	#RXE!SCL,SCRTCH	;ADD IN REC ENB, STOP CODE AND CHARACTER LENGTH
1350	003474	013777	002072	176412		MOV	SCRTCH,@VRBUF	;SET LINE PARAM REGISTER.
1351	003502	052777	000040	176402		BIS	#SCAN,@VZCSR	;ALLOW REC. AND XMIT. SCANS.
1352	003510	005037	002072			CLR	SCRTCH	;SET RETRY COUNTER
1353	003514	004037	017672		82\$:	JSR	RO,ZFLAG	;ISSUE ESC Z TO VT61
1354	003520	012637	002320			MOV	(SP)+,IDENT	;POP STACK INTO IDENT
1355	003524	100017				BPL	11\$	;IF IDENT IS -1, CLEAR IT
1356	003526	005237	002072			INC	SCRTCH	;CAN WE TRY AGAIN?
1357	003532	023727	002072	000005		CMP	SCRTCH,#5	
1358	003540	001407				BEQ	81\$	;IF NO, BRANCH
1359	003542	013737	002136	002140		MOV	X,Y	;WAIT BEFORE RETRY
1360	003550	005337	002140		80\$:	DEC	Y	
1361	003554	001375				BNE	80\$	
1362	003556	000756				BR	82\$	;TRY AGAIN
1363	003560	005037	002320		8 \$:	CLR	IDENT	
1364	003564				11\$:			
(2)	003564	012637	002356			MOV	(SP)+,CHRD	;POP STACK INTO CHRD
1365	003570	001375				BNE	11\$	
1366	003572	105037	002321			CLRB	IDENT+1	;CLEAR ALL BUT IDENT BITS.
1367	003576	104401	001171			TYPE	,\$CRLF	
1368	003602	104401	027506			TYPE	,DVUNIT	;ISSUE UNIT UNDER TEST MESSAGE
1369	003606	013746	002112			MOV	VZCSR,-(SP)	;SAVE VZCSR FOR TYPEOUT
(1)								;TYPE THE ADDRESS
(1)	003612	104403				TYPOS		;GO TYPE--OCTAL ASCII
(1)	003614	006				.BYTE	6	;TYPE 6 DIGIT(S)
(1)	003615	001				.BYTE	1	;TYPE LEADING ZEROS
1370	003616	017746	176254			MOV	@VECPT,-(SP)	;SAVE @VECPT FOR TYPEOUT
(1)								;TYPE THE VECTOR
(1)	003622	104403				TYPOS		;GO TYPE--OCTAL ASCII
(1)	003624	006				.BYTE	6	;TYPE 6 DIGIT(S)
(1)	003625	000				.BYTE	0	;SUPPRESS LEADING ZEROS
1371	003626	013737	002126	002356		MOV	OCTLNE,CHRD	
1372	003634	013746	002356			MOV	CHRD,-(SP)	;SAVE CHRD FOR TYPEOUT
(1)								;TYPE THE LINE
(1)	003640	104403				TYPOS		;GO TYPE--OCTAL ASCII
(1)	003642	006				.BYTE	6	;TYPE 6 DIGIT(S)
(1)	003643	000				.BYTE	0	;SUPPRESS LEADING ZEROS
1373	003644	013746	002320			MOV	IDENT,-(SP)	;SAVE IDENT FOR TYPEOUT
(1)								;TYPE THE IDENT
(1)	003650	104403				TYPOS		;GO TYPE--OCTAL ASCII
(1)	003652	006				.BYTE	6	;TYPE 6 DIGITS
(1)	003653	000				.BYTE	0	;SUPPRESS LEADING ZEROS
1374	003654	104401	001171			TYPE	,\$CRLF	;CARRIAGE RETURN AND LINE FEED
1375	003660	032737	000001	002320		BIT	#BIT00,IDENT	;UNIT HAVE A COPIER?
1376	003666	001402				BEQ	20\$	;NO
1377	003670	104401	027633			TYPE	,\$DCOPYR	;YES-ISSUE COPIER MESSAGE
1378	003674	032737	000002	002320	20\$:	BIT	#BIT01,IDENT	;UNIT HAVE A PRINTER?
1379	003702	001402				BEQ	21\$	;NO
1380	003704	104401	027605			TYPE	,\$DPRTR	;YES-ISSUE PRINTER MESSAGE.
1381	003710	005037	002374		21\$:	CLR	FTLCNT	;CLEAR COUNT OF FATAL XMITs.

```

1382 003714 062737 000002 002104      ADD      #2,MNEPT      ;UPDATE MODEM FLAG POINTER
1383 003722 062737 000002 002102      ADD      #2,LNEPT      ;UPDATE LINE POINTER.
1384 003730 012737 033676 033674      MOV      #ABBUF,ABUFP    ;RESET THE REC. DATA POINTER
1385 003736 052777 000100 176146      BIS      #RENA,@VZCSR    ;SET THE REC. INT. ENABLE FOR TESTS
1386 003744 105737 002372              TSTB      MODE           ;CHECK TESTING MODE
1387 003750 001403              BEQ      ASTRT          ;AUTO MODE
1388 003752 012737 001572 002370      MOV      #INTAB,TSTPTR ;LOAD THE INITIAL TEST NUMBER
1389
1390      ;*****
1391      ;*****
1392      ;THIS TEST ISSUES ALL ESCAPE SEQUENCES AND
1393      ;INSURES THE VT61 HAS NOT FAILED DURING AN
1394      ;ESC SEQUENCE BY ISSUING A ESC Z TO FORCE A
1395      ;VT61 RESPONSE. THE PURPOSE OF THE TEST IS TO ATTEMPT TO
1396      ;INSURE THAT SUBSEQUENT TESTS WILL NOT RESULT IN
1397      ;A 'HUNG' UNIT. DATA IS NOT EVALUATED.
1398      ;*****
1399
1400 003760      ASTRT:
1401      ;*****
1402      (3)
1403      (2) 003760 000004      TST1:      SCOPE
1404      (1) 003762 012737 000001 001160      MOV      #1,$TIMES      ;;DO 1 ITERATION
1405      (1) 003770 012737 003776 001106      MOV      #ESTST,$LPADR  ;;SET SCOPE LOOP ADDRESS
1406
1407 1402 003776 113737 001102 002424      ESTST:  MOVB     $TSTNM,TSTNM    ;IN CASE OF ERROR - FOR TYPEOUT
1408 1403 004004 012701 002144      MOV      #BEL,R1        ;POINT TO FIRST COMMAND
1409 1404 004010 042777 000100 176074      BIC      #RENA,@VZCSR    ;CLEAR REC. INT. ENABLE
1410 1405 004016 005037 002410      CLR      PRESC          ;
1411 1406 004022 005004              CLR      R4
1412 1407 004024              ZERST:
1413      (2) 004024 013746 002364      MOV      ZERO,-(SP)      ;;PUSH ZERO ON STACK
1414 1408 004030 012702 002410      MOV      #PRESC,R2      ;SET UP SEQUENCE ADDRESS
1415 1409 004034 012103              GCMD:  MOV      (R1)+,R3      ;LOAD THE COMMAND
1416 1410 004036 001405      BEQ      1$           ;IF CHAR. ZERO,MUST BE XMIT TERMINATOR
1417 1411 004040 100537      BMI      ESTEX          ;TABLE EXPENDED - EXIT TEST.
1418 1412 004042 120327 000004      CMPEB     R3,#4          ;IS COMMAND ACTUALLY A DELIMITER?
1419 1413 004046 103444      BLO      DELIM          ;YES, GO UPDATE FUNCTIONS
1420 1414 004050 001473      BEQ      SPTN          ;NO, ITS A '10' - SPECIAL CASE.
1421 1415 004052 005704      1$:      TST      R4          ;SEE IF FLAG INDICATING SEQ.
1422 1416 004054 100474      BMI      SEQ4          ;4 IS SET. - YES EXIT
1423 1417 004056 010337 002412      2$:      MOV      R3,ESSEQ      ;PUSH THE SEQUENCE TO BE TESTED
1424 1418 004062              INXMT:
1425      (2) 004062 013746 002412      MOV      ESSEQ,-(SP)      ;;PUSH ESSEQ ON STACK
1426 1419 004066 005704      TST      R4          ;DOES THIS SEQUENCE REQUIRE
1427 1420 004070 001402      BEQ      3$           ;ADDITIONAL ESC?
1428 1421 004072 013746 002410      MOV      PRESC,-(SP)      ;;PUSH PRESC ON STACK
1429 1422
1430 1423 004076 004037 015464      3$:      JSR      R0,TESC      .GO TRANSMIT THIS SEQUENCE.
1431 1424
1432 1425 004102 005704      4$:      TST      R4          ;IN I/O SEQUENCES?
1433 1426 004104 100011      BPL      40$          ;NO
1434 1427 004106 012737 000054 021440      MOV      #44,$DCOUNT    ;YES,SET UP TO DELAY 1+ SEC.
1435 1428 004114 004037 021376      JSR      R0,DELAY
1436 1429 004120 032777 100000 175766      BIT      #RRDY,@VRBUF    ;CLEAR THE SILO
1437 1430 004126 001374      BNE      .-

```

PC	OP	OP2	OP3	OP4	OP5	OP6	OP7	OP8	OP9	OP10	OP11	OP12	OP13	OP14	OP15	OP16	OP17	OP18	OP19	OP20	OP21	OP22	OP23	OP24	OP25	OP26	OP27	OP28	OP29	OP30	OP31	OP32	OP33	OP34	OP35	OP36	OP37	OP38	OP39	OP40	OP41	OP42	OP43	OP44	OP45	OP46	OP47	OP48	OP49	OP50	OP51	OP52	OP53	OP54	OP55	OP56	OP57	OP58	OP59	OP60	OP61	OP62	OP63	OP64	OP65	OP66	OP67	OP68	OP69	OP70	OP71	OP72	OP73	OP74	OP75	OP76	OP77	OP78	OP79	OP80	OP81	OP82	OP83	OP84	OP85	OP86	OP87	OP88	OP89	OP90	OP91	OP92	OP93	OP94	OP95	OP96	OP97	OP98	OP99	OP100	OP101	OP102	OP103	OP104	OP105	OP106	OP107	OP108	OP109	OP110	OP111	OP112	OP113	OP114	OP115	OP116	OP117	OP118	OP119	OP120	OP121	OP122	OP123	OP124	OP125	OP126	OP127	OP128	OP129	OP130	OP131	OP132	OP133	OP134	OP135	OP136	OP137	OP138	OP139	OP140	OP141	OP142	OP143	OP144	OP145	OP146	OP147	OP148	OP149	OP150	OP151	OP152	OP153	OP154	OP155	OP156	OP157	OP158	OP159	OP160	OP161	OP162	OP163	OP164	OP165	OP166	OP167	OP168	OP169	OP170	OP171	OP172	OP173	OP174	OP175	OP176	OP177	OP178	OP179	OP180	OP181	OP182	OP183	OP184	OP185	OP186	OP187	OP188	OP189	OP190	OP191	OP192	OP193	OP194	OP195	OP196	OP197	OP198	OP199	OP200	OP201	OP202	OP203	OP204	OP205	OP206	OP207	OP208	OP209	OP210	OP211	OP212	OP213	OP214	OP215	OP216	OP217	OP218	OP219	OP220	OP221	OP222	OP223	OP224	OP225	OP226	OP227	OP228	OP229	OP230	OP231	OP232	OP233	OP234	OP235	OP236	OP237	OP238	OP239	OP240	OP241	OP242	OP243	OP244	OP245	OP246	OP247	OP248	OP249	OP250	OP251	OP252	OP253	OP254	OP255	OP256	OP257	OP258	OP259	OP260	OP261	OP262	OP263	OP264	OP265	OP266	OP267	OP268	OP269	OP270	OP271	OP272	OP273	OP274	OP275	OP276	OP277	OP278	OP279	OP280	OP281	OP282	OP283	OP284	OP285	OP286	OP287	OP288	OP289	OP290	OP291	OP292	OP293	OP294	OP295	OP296	OP297	OP298	OP299	OP300	OP301	OP302	OP303	OP304	OP305	OP306	OP307	OP308	OP309	OP310	OP311	OP312	OP313	OP314	OP315	OP316	OP317	OP318	OP319	OP320	OP321	OP322	OP323	OP324	OP325	OP326	OP327	OP328	OP329	OP330	OP331	OP332	OP333	OP334	OP335	OP336	OP337	OP338	OP339	OP340	OP341	OP342	OP343	OP344	OP345	OP346	OP347	OP348	OP349	OP350	OP351	OP352	OP353	OP354	OP355	OP356	OP357	OP358	OP359	OP360	OP361	OP362	OP363	OP364	OP365	OP366	OP367	OP368	OP369	OP370	OP371	OP372	OP373	OP374	OP375	OP376	OP377	OP378	OP379	OP380	OP381	OP382	OP383	OP384	OP385	OP386	OP387	OP388	OP389	OP390	OP391	OP392	OP393	OP394	OP395	OP396	OP397	OP398	OP399	OP400	OP401	OP402	OP403	OP404	OP405	OP406	OP407	OP408	OP409	OP410	OP411	OP412	OP413	OP414	OP415	OP416	OP417	OP418	OP419
----	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

```

1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
  
```

```

*****
;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 TRANSMISSION IS
;CHECKED OF SOM/EOM.
*****
*****
TST2:  SCOPE
      MOV  #5,$TIMES      ;;DO 5 ITERATIONS
      MOV  #CKMNT,$LPADR  ;;SET SCOPE LOOP ADDRESS
      CKMNT:  MOV  $TSTNM,TSTNM  ;IN CASE OF ERROR - FOR TYPEOUT
              JSR  R0,RESETV      ;RESET THE UNIT AND SETMAINT. MODE.
              MOV  #SOM,@TBUF     ;XMIT THE START OF MESSAGE.
              JSR  R0,XMIT1
              MOV  ESCZ,@TBUF
              JSR  R0,XMIT1      ;SEND AN IDENT REQUEST.
              MOV  ESCZ+1,@TBUF
              JSR  R0,XMIT1
              MOV  #EOM,@TBUF     ;XMIT END OF MESSAGE.
              JSR  R0,XMIT1
              MOV  #50,R1         ;SET UP FAILSAFE DELAY
              BIT  #RSOM,VSTAT    ;RECEIVED THE START OF MESSAGE?
              BNE  CKEOM          ;YES-GO LOOK FOR EOM.
              MOV  #1,DCOUNT      ;WAIT 20MS
              JSR  R0,DELAY
              DEC  R1             ;WAITED FULL COUNT?
              BNE  1$            ;BR IF NOT
              BNE  1$            ;AND KEEP LOOKING.
              CKEOM:  MOV  #50,R1 ;SET MAX DELAY FOR 500 M.S.
              BIT  #REOM,VSTAT    ;RECEIVED END OF MESSAGE?
              BNE  10$           ;YES-CHECK FOR BOTH RECEIVED.
              MOV  #1,DCOUNT      ;DELAY FOR 10 M.S.
              JSR  R0,DELAY
              DEC  R1             ;AND KEEP LOOKING.
              BNE  1$
              BIT  #RSOM,VSTAT    ;RECEIVED SOM?
              BEQ  2$            ;NO ISSUE ERROR
              BIT  #REOM,VSTAT    ;RECEIVED EOM?
              BNE  EXMNT         ;YES, NO ERRORS-EXIT.
              MOV  #6001,$GDDAT   ;LOAD ERROR WITH EXPECTED
              MOV  VSTAT,$BDDAT   ;AND ACTUAL STATUS.
              ERROR  22
      EXMNT:  NOP
*****
;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 POSITION ROW 23 COLUMN 80 AND THE POSITION IS AGAIN
;VERIFIED. ERRORS ARE REPORTED IF THE POSITIONS ARE INCORRECT.
*****
  
```

```

1538
1539
1540 *****
(2) 004600 000004
(1) 004602 012737 000005 001160 TST3: SCOPE
(1) 004610 012737 004616 001106 MOV #5,$TIMES ;;DO 5 ITERATIONS
MOV #CURS1,$LPADR ;;SET SCOPE LOOP ADDRESS
1541
1542 004616 113737 001102 002424 CURS1: MOVB $TSTNM,TSTNM ;IN CASE OF ERROR - FOR TYPEOUT
1543 004624 013701 017460 MOV TBBUF,R1 ;USE R1 AS XMIT BUFFER POINTER.
1544 004630 004037 017470 JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
1545 004634 013721 002250 MOV ESCOI,(R1)+ ;CLFT. RESET, READ CURSOR
1546 004640 113721 002244 MOVB RDCUR,(R1)+ ;POSITION, CURSOR LEFT.
1547 004644 012737 000003 017466 MOV #3,XMCNT ;XMIT 3 BYTES
1548
1549 004652 004037 020116 JSR R0,XMREC ;XMIT AND RECEIVE.
1550 004656 000402 BR 10$ ;NORMAL EXIT.
1551 004660 104011 ERROR 11 ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
1552 004662 000446 BR 2$ ;EXIT TEST.
1553 004664 013701 032474 10$: MOV RCRLB,R1 ;GET THE CURRENT CURSOR POSITION.
1554 004670 020137 002350 CMP R1,CUHME ;CURSOR REALLY HOME?
1555 004674 001405 BEQ 1$ ;YES EXIT
1556 004676 104012 ERROR 12 ;VT61 FAILURE MESSAGE
1557 004700 013746 002350 MOV CUHME,-(SP) ;PUSH CUHME ON STACK
1558 004704 004037 020562 JSR R0,CURER ;GO LOAD AND ISSUE CURSOR ERROR
1559
1560 004710 013701 017460 1$: MOV TBBUF,R1 ;LOAD XMIT BUFFER WITH
1561 004714 013721 002234 MOV DCRAD,(R1)+
1562 004720 013721 002236 MOV R23C79,(R1)+ ;CURSOR TO ROW 23,COL.79
1563 004724 013721 002250 MOV ESCOI,(R1)+ ;READ CURSOR POSITION
1564 004730 013721 002244 MOV RDCUR,(R1)+ ;IT AND CURSOR RIGHT
1565 004734 012737 000007 017466 MOV #7,XMCNT ;XMIT 7 BYTES.
1566 004742 004037 020116 JSR R0,XMREC ;XMIT AND RECEIVE
1567 004746 000402 BR 20$ ;NORMAL EXIT.
1568 004750 104011 ERROR 11 ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
1569 004752 000412 BR 2$ ;EXIT TEST.
1570 004754 012701 032474 20$: MOV #RCRLB,R1
1571
1572 004760 023711 002236 CMP R23C79,(R1) ;CHECK CURSOR POSITION TO LOWER RT.
1573 004764 001405 BEQ 2$ ;OK, EXIT
1574 004766 104012 ERROR 12 ;VT61 FAILURE MESSAGE
1575 004770 013746 002236 MOV R23C79,-(SP) ;PUSH R23C79 ON STACK
1576 004774 004037 020562 JSR R0,CURER ;LOAD AND ISSUE CURSOR ERROR .
1577 005000 000240 2$: NOP
1578 *****
1579 ;ROUTINE TO INSURE THE UNIT CAN ENTER LINEAR ADDRESSING
1580 ;MODE. 81 CHARACTERS ARE ISSUED TO THE UNIT UNDER TEST
1581 ;THEN THE CURSOR POSITION IS READ AND MUST BE ROW1,COL.0.
1582 *****
1583 *****
1584 *****
(2) 005002 000004
(1) 005004 012737 000005 001160 TST4: SCOPE
(1) 005012 012737 005020 001106 MOV #5,$TIMES ;;DO 5 ITERATIONS
MOV #CKLIN,$LPADR ;;SET SCOPE LOOP ADDRESS
1585
1586 005020 113737 001102 002424 CKLIN: MOVB $TSTNM,TSTNM ;IN CASE OF ERROR - FOR TYPEOUT
1587 005026 004037 017470 JSR R0,RESETV ;RESET THE UNIT-SET MAINT AND LINEAR MODES

```

```

1588 005032 013701 017460      MOV    TBBUF,R1
1589 005036 012703 000120      MOV    #80.,R3
1590 005042 004037 021442      JSR    R0,BLDINC      ;LOAD XMIT BUFFER WITH 80 CHAR AND
1591 005046 013721 002242      MOV    RCUR,(R1)+
1592 005052 013721 002244      MOV    RDCUR,(R1)+   ;READ (CURSOR POSINION.
1593 005056 012737 000123 017466  MOV    #83.,XMCNT
1594 005064 004037 020116      JSR    R0,XMREC      ;XMIT THE BUFFER.
1595 005070 000402
1596 005072 104011      BR     1$
1597 005074 000421      ERROR  11          ;LAST XMIT CAUSED UNIT TO HANG.
1598 005076 023777 002330 012020 1$:  BR     LINXT          ;EXIT TEST
1599 005104 001415      CMP    R01C00,@RBBUF ;CURSOR AT ROW1,COL. 0?
1600 005106 013737 002250 001124      BEQ    LINXT          ;YES-EXIT
1601 005114 000337 001124      MOV    ESCO,$GDDAT
1602 005120 013737 002204 001126      SWAB   $GDDAT
1603 005126 104001      MOV    DRECT,$BDDAT  ;ISSUE ESC SEQUENCE AND CURSOR
1604 005130 013746 002330      ERROR  1
1605 005134 004037 020562      MOV    R01C00,-(SP)  ;:PUSH R01C00 ON STACK
1606 005140 000240      JSR    R0,CURER
1607      LINXT:  NOP
1608
1609      ;:*****
1610      ;:TEST TO INSURE OPERATION OF XON/XOFF COMMANDS
1611      ;:FROM VT61. XOFF IS FORCED BY TRANSMITTING LINE 23 WHILE SIMUL-
1612      ;:TANEOUSLY FILLING THE SILO WITH DATA. AFTER SENSING
1613      ;:THE XOFF, THE TEST WAITS FOR THE TRANSMIT TO FINISH AND
1614      ;:INSURES XON OCCURS BEFORE THE MAX. TRANSFER TIME HAS ELAPSED.
1615      ;:(30 SECONDS)
1616      ;:*****
1617
1618      ;:*****
1619      ;:*****
1620      ;:*****
1621      ;:*****
1622      ;:*****
1623      ;:*****
1624      ;:*****
1625      ;:*****
1626      ;:*****
1627      ;:*****
1628      ;:*****
1629      ;:*****
1630      ;:*****
1631      ;:*****
1632      ;:*****
1633      ;:*****
1634      ;:*****
1635      ;:*****
1636      ;:*****
1637      ;:*****
1638      ;:*****
1639      ;:*****
1640      ;:*****
  
```

```

1641 005314 004037 017732          JSR      R0,CKSFT      ;GO RFPRT ERROR
1642 005320          3$:      MOV      #XMDNE,-(SP)      ;;PUSH #XMDNE ON STACK
      (2) 005320 012746 000001      MOV      #50,-(SP)      ;;PUSH #50. ON STACK
1643 005324 012746 000062          JSR      R0,WTBGND
1644 005330 004037 023074          BR       EXIT3      ;TIMEOUT-EXIT TEST.
1645 005334 000411          CMPS     @ABUFP,#XON      ;RECEIVED A XON?
1646 005336 127727 026332 000021      BEQ      EXIT3      ;YES-NO ERROR-EXIT
1647 005344 001405
1648
1649 005346 104012          ERROR     12      ;GENERAL VT61 FAILURE MESSAGE
1650 005350 012746 000001      MOV      #000001,-(SP)      ;;PUSH #000001 ON STACK
1651 005354 004037 017732          JSR      R0,CKSFT
1652 005360 004037 020502      EXIT3: JSR      R0,RESPTR      ;RESET INTERRUPT POINTERS.
1653
1654      ;*****
1655      ;ROUTINE TO VERIFY OPERATION OF XOFF AND XON TO THE VT61.
1656      ;A FULL SCREEN TRANSMIT IS INITIATED AND A SERIES OF XOFF AND
1657      ;XON ARE ISSUED TO THE TERMINAL SEQUENTIALLY.
1658      ;ERRORS ARE REPORTED IF XOFF DOES NOT STOP,OR XON RESTART
1659      ;THE TRANSMISSION. TEST IS ENDED WHEN EOM IS SENSED.
1660      ;*****
1661
1662      ;*****
      (2) 005364 000004      TST6:  SCOPE
      (1) 005366 012737 000001 001160      MOV      #1,$TIMES      ;;DO 1 ITERATION
      (1) 005374 012737 005402 001106      MOV      #ONOF61,$LPADR      ;;SET SCOPE LOOP ADDRESS
1663
1664 005402 113737 001102 002424      ONOF61: MOV     $TSTNM,TSTNM      ;IN CASE OF ERROR - FOR TYPEOUT
1665 005410 004037 017470          JSR      R0,RESETV      ;RESET THE UNIT AND WAIT FOR XON.
1666 005414 042737 077577 002420      BIC      #77577,VSTAT      ;CLEAR THE FLAGS
1667 005422 013746 002364          MOV      ZERO,-(SP)      ;;PUSH ZERO ON STACK
1668 005426 013746 002252          MOV      XMTAL,-(SP)      ;;PUSH XMTAL ON STACK
1669 005432 013746 002250          MOV      ESCO,-(SP)      ;;PUSH ESCO ON STACK
1670 005436 004037 015464          JSR      R0,TEC
1671 005442 012737 000010 021440      UNOFLP: MOV     #10,DCOUNT      ;ALLOW 100 M.S. FOR OPERATION
1672 005450 004037 021376          JSR      R0,DELAY      ;TO BEGIN.
1673 005454 112777 000023 012002      MOV     #XOFF,@TBUFP
1674 005462 004037 020372          JSR      R0,XMIT1      ;SEND A XOFF TO VT61.
1675 005466 012704 000036          MOV      #30,R4
1676 005472 013705 033674          OFFLP: MOV     ABUFP,R5      ;ALLOW 300M.S. FOR XMIT TO CEASE
1677 005476 012737 000001 021440      MOV      #1,DCOUNT
1678 005504 004037 021376          JSR      R0,DELAY
1679 005510 023705 033674          CMP      ABUFP,R5
1680 005514 001406          BEQ      ONOFA
1681 005516 005304          DEC      R4
1682 005520 001364          BNE     OFFLP
1683 005522 013737 002420 001120      MOV      VSTAT,$GDADR      ;COUNTER NO EQUAL 300 MS-LOOP
1684 005530 104015          ERROR     15      ;UNIT DID NOT RESPOND TO XOFF
1685
1686 005532 112777 000021 011724      ONOFA: MOV     #XON,@TBUFP      ;ISSUE ERROR
1687 005540 004037 020372          JSR      R0,XMIT1      ;SEND A XON TO THE VT61.
1688 005544 012704 000036          MOV      #30,R4      ;SET UP FOR 300MS DELAY.
1689 005550 032737 020000 002420      ONLP:  BIT      #REOM,VSTAT      ;EOM RECEIVED?
1690 005556 001020          BNE     ONOFT
1691 005560 013705 033674          MOV      ACJFP,R5      ;YES-EXIT
1692 005564 012737 000001 021440      MOV      #1,DCOUNT

```

```

1693 005572 004037 021376 JSR R0,DELAY ;ALLOW 300 MS FOR XMIT TO RESTART
1694 005576 023705 033674 CMP ABUF,R5
1695 005602 001317 BNE ONOFLP ;IT RESTARTED-GO STOP IT.
1696 005604 005304 DEC R4
1697 005606 001360 BNE ONLP ;NOT YET 300 MS LOOP.
1698 005610 013737 002420 001120 MOV VSTAT,$GDADR ;XMIT DIT NOT RESTART-ISSUE
1699 005616 104016 ERROR 16 ;ERROR AND EXIT
1700 005620 000240 ONOEXT: NOP
1701
1702 ;*****
1703 ;ROUTINE TO TEST VT61 RAM AND THE COMMUNICATION PATHS.
1704 ;THIS ROUTINE ISSUES A SERIES OF PATTERNS(77/100,100/77,
1705 ;52/125,INCREMENTING,AND REV. VIDEO INCREMENTING) TO THE VT61.
1706 ;THE SCREEN IS THEN TRANSMITTED TO THE HOST AND AFTER EACH
1707 ;ITERATION RECEIVED DATA IS CHECKED AND ALL ERRORS(INCLUDING
1708 ;TRANSMISSION) ARE REPORTED.
1709 ;*****
1710
1711 ;*****
1712 (2) 005622 000004 TST: SCOPE
1713 (1) 005624 012737 000001 001160 MOV #1,$TIMES ;DO 1 ITERATION
1714 (1) 005632 012737 005640 001106 MOV #MEM1,$LPADR ;SET SCOPE LOOP ADDRESS
1715 005640 113737 001102 002424 MEM1: MOV $TSTNM,TSTNM ;IN CASE OF ERROR - FOR TYPEOUT
1716 005646 004037 017470 JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
1717 005652 005005 CLR R5 ;CLEAR PATTERN OFFSET.
1718 005654 016504 006460 MEMA: MOV MPATT(R5),R4 ;LOAD PATTERN TO BE TRANSMITTED
1719 005660 005037 006500 CLR INCLCT ;RESET LINE COUNT FOR INC. PATTERN
1720 005664 012737 100040 006502 MOV #100040,INCSCH ;RESET SEED CHAR. FOR INC. PATTERN
1721 005672 004037 020502 JSR R0,RESPTR ;RESET POINTERS
1722 005676 042737 002420 BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1723 005704 112777 000002 011552 MOV $SOM,@TBUF ;XMIT THE START OF MESSAGE.
1724 005712 004037 020372 JSR R0,XMIT1
1725 005716 012702 003600 MOV #TOTCH,R2 ;LOAD A COUNT OF SCREEN
1726 005722 005302 MEMB: DEC R2 ;DECREMENT XMIT COUNT
1727 005724 001414 BEQ 10$ ;COUNT - ZERO?
1728 005726 004037 006360 12$: JSR R0,PATGN ;NO-GENERATE NEXT BYTE TO XMIT.
1729 005732 110477 011526 MOV $R4,@TBUF ;LOAD THE CHARACTER.
1730 005736 004037 020372 JSR R0,XMIT1 ;NO-XMIT ANOTHER BYTE.
1731 005742 023737 002374 002376 CMP FTLCNT,ALWCNT ;EXCEEDED FATAL ERROR COUNT?
1732 005750 103764 BLO MEMB ;NO-CHECK IF ANOTHER TRANSMISSION REQUIRED.
1733 005752 000137 006504 JMP MEMXT ;YES-GO ABORT TEST.
1734 005756 112777 000004 011500 10$: MOV $EOM,@TBUF ;XMIT END OF MESSAGE.
1735 005764 004037 020372 JSR R0,XMIT1
1736 005770 004037 020502 JSR R0,RESPTR ;RESET INTERRUPT POINTERS.
1737 005774 013701 017460 MOV TBUF,R1 ;LOAD XMIT BUFFER WITH
1738 006000 013721 002326 MOV ESCN,(R1)+
1739 006004 013721 002156 MOV CHOM,(R1)+ ;CURSOR HOME
1740 006010 013721 002322 MOV ESCZI,(R1)+ ;ESCAPE Z
1741 006014 013721 002250 MOV ESCO,(R1)+
1742 006020 013721 002252 MOV XMTAL,(R1)+ ;TRANSMIT ALL
1743 006024 013711 002150 MOV LNFD,(R1) ;LINE FEED.
1744 006030 012737 000010 017466 MOV #2,XMCNT ;SET UP TO XMIT 8 BYTES
1745 006036 004037 020116 JSR R0,XMREC ;XMIT,WAIT FOR REC. EOM
  
```

1746	006042	000403			BR	1\$	:NORMAL EXIT
1747	006044	104011			ERROR	11	:LAST TRANSMIT CAUSED VT61 TO HANG
1748	006046	000137	006504		JMP	MEMXT	:EXIT TEST
1749	006052	042737	077577	002420	1\$: BIC	#77577,VSTAT	:CLEAR ALL FLAGS BUT XOFF AND XMKIL
1750	006060	005037	006500		CLR	INCLCT	:RESET THE LINE COUNT FOR INC. PATTERN
1751	006064	012737	100040	006502	MOV	#100040,INCSCH	:RESET THE SEED CHAR. FOR INC. PATTERN
1752	006072	005002			CLR	R2	:CLEAR RECEIVE COUNTER.
1753	006074	016504	006460		MOV	MPATT(R5),R4	:LOAD PATTERN
1754	006100	012703	033474		MOV	#TCRLB+300,R3	:SET UT ERROR STORAGE
1755	006104	013701	017124		MOV	RBBUF,R1	:SET UP RECEIVE POINTER
1756	006110	005037	002414		CLR	DLAY	:SET UP TIME OUT DELAY
1757	006114	013737	017124	017130	MOV	RBBUF,RBUF	:RESET RECEIVE POINTER
1758	006122	023701	017130		1\$: CMP	RBUF,R1	:RECEIVED A CHAR?
1759	006126	001014			BNE	MEMD	:YES-GO CHECK IT.
1760	006130	032737	020000	002420	BIT	#REOM,VSTAT	:HAVE WE RECEIVED EOM?
1761	006136	001034			BNE	CKDAT	:YES, GO CHECK FOR DATA ERRORS
1762	006140	005337	002414		DEC	DLAY	:RUN TIME OUT DELAY.
1763	006144	001366			BNE	1\$	:NOT EXPIRED-KEEP LOOKING.
1764	006146	005237	002374		INC	FT' CNT	:TRANSMISSION FAILED-INCR. FATAL COUNT

1767	006152	104011		ERROR	11	
1768	006154	000137	006504	JMP	MEMXT	
1769	006160	005202		MEMD: INC	R2	:DATA IN. INCREMENT COUNTER
1770	006162	004037	006360	JSR	R0,PATGN	:GET GOOD CHARACTER,PUT IN R4 AND
1771	006166	122705	000010	CMPB	#10,R5	:CHECKING REV. VIDEO DATA?
1772	006172	001002		BNE	1\$	:NO-DO NOT MODIFY
1773	006174	052704	000200	BIS	#BIT07,R4	:YES-FORCE BIT 7.
1774	006200	121104		1\$: CMPB	(R1),R4	:COMPARE DATA
1775	006202	001742		BEQ	MEMC	
1776	006204	020227	003600	CMP	R2,#TOTCH	:COMPARING LAST CHAR?
1777	006210	001737		BEQ	MEMC	:YES-NEVER COUNT AS A ERROR.
1778						
1779	006212	020327	033544	CMP	R3,#TCRLB+350	:STORED 20 ERRORS?
1780	006216	103334		BHIS	MEMC	:YES-STORE NO MORE.
1781	006220	110423		MOVB	R4,(R3)+	:STORE THE GOOD DATA.
1782	006222	111123		MOVB	(R1),(R3)+	:STORE THE BAD DATA.
1783	006224	010223		MOV	R2,(R3)+	:STORE THE RECEIVE COUNT.
1784	006226	000730		BR	MEMC	
1785	006230	022703	033474	CKDAT: CMP	#TCRLB+300,R3	
1786	006234	001415		BEQ	CKMEM	
1787	006236	012701	033474	MOV	#TCRLB+300,R1	:LOAD FIRST ERROR ADDRESS.
1788	006242	004037	020074	1\$: JSR	R0,CLREG	:CLEAR ERROR REGISTERS
1789	006246	112137	001124	MOVB	(R1)+,\$GDDAT	:LOAD THE GOOD DATA.
1790	006252	112137	001126	MOVB	(R1)+,\$BDDAT	:LOAD THE ERROR BUFFER
1791	006256	012137	001120	MOV	(R1)+,\$GDADR	:LOAD RECEIVE COUNT
1792	006262	104004		ERROR	4	:ISSUE DATA ERROR MESSAGE.
1793	006264	020103		CMP	R1,R3	:ISSUED ALL ERRORS?
1794	006266	103765		BLO	1\$	:NO-CONTINUE
1795						
1796	006270	020227	003600	CKMEM: CMP	R2,#TOTCH	:DID WE XFER 1920 TIMES?
1797	006274	001406		BEQ	1\$	:YES - GO CHECK STATUS

1799	006276	012737	003600	001124	MOV	#TOTCH,\$GDDAT	;NO, PUT GOOD COUNT IN GDDAT
1800	006304	010237	001126		MOV	R2,\$BDDAT	;AND ACTUAL COUNT IN BDDAT.
1801	006310	104005			ERROR	5	;ISSUE COUNT ERROR.

```

1803
1804 006312 012746 060000 1$: MOV #60000,-(SP) ;;PUSH #60000 ON STACK
(2) 006312 004037 017732 JSR R0,CKSFT
1805 006316 062705 000002 ADD #2,R5 ;;INCREMENT PATTERN POINTER
1806 006322 005765 006460 TST MPATT(R5) ;;TEST NEXT PATTERN
1807 006332 001464 BEQ MEMXT ;;ZERO-END OF TEST EXIT.
1808 006334 100007 SPL 2$ ;;NOT INCRMENTING PATTERN.
1809 006336 122705 000010 CMPB #10,R5 ;;SET REVERSE VIDEO?
1810 006342 012703 006474 BNE 2$ ;;NO.
1811 006344 004037 020442 MOV #SETREV,R3 ;;YES-ENTER REVERSE VIDEO
1812 006350 000137 005654 JSR R0,LDXMIT ;;AND RE-ISSUE INCREMENTING PATTERN.
1813 006354 000137 005654 JMP MEMA ;;NOT ZERO, GO EXERCISE IT.
1814
1815
1816 006360 042704 000200 PATGN: BIC #200,R4 ;;CLEAR REV. VIDEO BIT IF SET.
1817 006364 005704 TST R4 ;;CHECK R4 FOR PATTERN
1818 006366 100402 BMI 1$ ;;IF MINUS, DO INCREMENTING.
1819 006370 000304 SWAB R4 ;;OTHERWISE SWAP BYTES AND
1820 006372 000200 RTS R0 ;;EXIT.
1821 006374 005237 006500 1$: INC INCLCT ;;INCREMENT THE LINE COUNT
1822 006400 022737 000121 006500 CMP #81.,INCLCT ;;REACHED FULL LINE? (80 CHAR.)
1823 006406 001015 BNE 12$ ;;BR IF NOT
1824 006410 005037 006500 CLR INCLCT ;;RESET COUNT
1825 006414 013704 006502 MOV INCSCH,R4 ;;START NEW LINE WITH SEED CHAR.
1826 006420 005237 006502 INC INCSCH ;;UPDATE SEED FOR NEXT LINE
1827 006424 122737 000177 006502 CMPB #177,INCSCH ;;EXCEEDED LIMIT?
1828 006432 103403 BLO 12$ ;;BR IF NOT.
1829 006434 016537 006460 006502 MOV MPATT(R5),INCSCH ;;RESET SEED
1830 006442 105204 12$: INCB R4 ;;ADD ONE TO INCREMENTING
1831 006444 120427 000177 CMPB R4,#177 ;;HAVE WE EXCEEDED LIMIT
1832 006450 103402 BLO 2$ ;;NO, EXIT
1833 006452 016504 006460 MOV MPATT(R5),R4 ;;YES, RESET PATTERN AND
1834 006456 000200 RTS R0 ;;EXIT.
1835
1836 006460 037500 MPATT =. ;;PATTERN 77,100
1837 006462 040077 .WORD 040077 ;;PATTERN 100,77
1838 006464 025125 .WORD 025125 ;;PATTERN 52,125
1839 006466 100040 .WORD 100040 ;;PATTERN INCREMENTING
1840 006470 100040 .WORD 100040 ;;PATTERN INCREMENTING-REV. VIDEO.
1841 006472 000000 .WORD 0 ;;PATTERN TABLE TERMINATOR
1842
1843
1844 006474 033 117 112 SETREV: .BYTE .ESC,.O,.EEMP,0 ;;SEQUENCE TO ENTER REVERSE VIDEO.
006477 000
1845 006500 000000 INCLCT: 0 ;;LINE COUNTER FOR INC. PATTERN
1846 006502 000000 INCSCH: 0 ;;SEED CHARACTER FOR INC. PATTERN
1847 006504 000240 MEMXT: NOP
1848
1849 ;;*****
1850
1851 ;;ROUTINE TO TEST THE ABILITY OF THE VT61 TO CALCULATE
1852 ;;AND TRANSMIT CHECKSUMS OF BOTH TRANSMITTED AND RECEIVED
1853 ;;DATA. SUBTEST A TRANSMITS A FULL BUFFER UPDATING A CALCULATED
1854 ;;CHECKSUM ON EACH CHARACTER TRANSMITTED. AN ESCAPE SEQUENCE
1855 ;;REQUESTING THE RECEIVER CHECKSUM IS EMBEDDED AT THE END OF
1856 ;;XMIT BUFFER AND THE RECEIVED CHECKSUM IS COMPARED TO THE

```

```

1857                                     ;CALCULATED. SUBTEST B PERFORMS THE SAME TYPE OF CHECK ON
1858                                     ;THE VT61 TRANSMIT CHECKSUM,UTILIZING THE DATA SENT TO THE VT61
1859                                     ;IN SUBTEST A,DURING A FULL SCREEN TRANSMIT.
1860
1861                                     ;:*****
1862                                     ;:*****
1863
(2) 006506 000004
(1) 006510 012737 000003 001160 TST10: SCOPE
(1) 006516 012737 006524 001106      MOV    #3,$TIMES      ;;DO 3 ITERATIONS
                                      MOV    #CKSUMA,$LPADR    ;;SET SCOPE LOOP ADDRESS
1864
1865 006524 113737 001102 002424 CKSUMA: MOVB    $TSTNM,TSTNM    ;IN CASE OF ERROR - FOR TYPEOUT
1866 006532 004037 017470      JSR    R0,RESETV      ;RESET THE UNIT AND WAIT FOR XON.
1867 006536 004037 020502      JSR    R0,RESPTR     ;RESET INTERRUPT POINTERS
1868 006542 012737 001001 002422      MOV    #1001,BLKM      ;SET XMIT TO SOM- DATA -EOM.
1869 006550 012703 007160      MOV    #ITSUMA,R3      ;DIS. RECT. MODE AND CLEAR CHECKSUM
1870 006554 004037 020442      JSR    R0,LDXMIT
1871 006560 042737 077577 002420      BIC    #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1872 006566 013701 017460      MOV    TBBUF,R1      ;LOAD XMIT BUFFER WITH
1873 006572 012703 000473      MOV    #315.,R3
1874 006576 004037 021442      JSR    R0,BLDINC      ;314 INCREMENTING CHAR.
1875 006602 113721 002326      MOVB    ESCN,(R1)+
1876 006606 113721 002156      MOVB    CHOM,(R1)+      ;CURSOR HOME
1877 006612 113721 002250      MOVB    ESCO,(R1)+
1878 006616 113721 002251      MOVB    ESCO+1,(R1)+
1879 006622 113711 002266      MOVB    TXRCK,(R1)      ;TRANSMIT RECEIVER CHECKSUM.
1880 006626 005004      CLR    R4      ;CLEAR CHECKSUM REGISTER
1881 006630 012705 000004      MOV    #EOM,R5      ;PRELOAD CHECKSUM REG. WITH
1882 006634 004037 022120      JSR    R0,CALCK      ;EOM FROM PRIOR XMIT.
1883 006640 052737 002000 002420      BIS    #CKSUM,VSTAT    ;REQUEST CHECKSUM CALCULATIONS.
1884 006646 012737 000500 017466      MOV    #320.,XMCNT      ;SETUP TO XMIT 320 BYTES
1885 006654 052777 040000 173230      BIS    #XENA,@VZCSR      ;ENABLE XMIT INTERRUPTS
1886 006662 012746 020000      MOV    #REOM,-(SP)      ;PUSH #REOM ON STACK
1887 006666 012746 000040      MOV    #40,-(SP)      ;PUSH #40 ON STACK
1888 006672 004037 023074      JSR    R0,WTBGND      ;LOOK FOR EOM.
1889 006676 000534      BR    CKEXT      ;ERROR EXIT IF NOT FOUND
1890 006700 127704 010220      CMPB    @RBBUF,R4      ;COMPARE CHECKSUMS
1891 006704 001414      BEQ    CKSUMB      ;GOOD GO TO SUBTEST B
1892 006706 004037 020074      JSR    R0,CLREG      ;BAD COMPARE
1893 006712 110437 001124      MOVB    R4,$GDDAT      ;LOAD CALCULATED CHECKSUM
1894 006716 117737 010202 001126      MOVB    @RBBUF,$BDDAT      ;AND VT61 RECEIVER CHECKSUM
1895 006724 104013      ERROR    13      ;ISSUE ERROR
1896 006726 012746 060001      MOV    #60001,-(SP)      ;PUSH #60001 ON STACK
1897 006732 004037 017732      JSR    R0,CKSFT      ;ERROR.
1898
1899 006736 042737 077577 002420 CKSUMB: BIC    #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1900 006744 005004      CLR    R4      ;CLEAR CHECKSUM REGISTER
1901 006746 052737 000100 002420      BIS    #TXSUM,VSTAT    ;SET UP FOR XMIT CHECKSUM GENERATION.
1902 006754 012737 001001 002422      MOV    #1001,BLKM      ;SET XMIT TO SOM- DATA -EOM.
1903 006762 013701 017460      MOV    TBBUF,R1
1904 006766 004037 022166      JSR    R0,LDBUF      ;LOAD THE BUFFER WITH:
1905 006772      033      117      134      .BYTE    .ESC,.O,.CLTCK,.ESC,.O,.XMTAL,.ESC,.O,.TXTCK,O
          006775      033      117      126
          007000      033      117      136
          007003      000
1906 007004 012737 000011 017466      MOV    #9.,XMCNT      ;SET UP TO XMIT 9 BYTES
  
```

1907	007012	052777	040000	173072		BIS	#XENA,@VZCSR	:ALLOW XMIT INTERRUPTS
1908	007020	012746	000001			MOV	#XMDNE,-(SP)	:PUSH #XMDNE ON STACK
1909	007024	012746	000002			MOV	#2,-(SP)	:PUSH #2 ON STACK
1910	007030	004037	023074			JSR	R0,WTBGND	:LOOK FOR XMIT DONE.
1911	007034	000455				BR	CKEXT	:TIME OUT - EXIT TEST.
1912	007036	005037	002414		CKSRC:	CLR	DLAY	:SET UP TIME OUT DELAY
1913	007042	013702	033674			MOV	ABUFP,R2	:RESET THE RECEIVER FLAG
1914	007046	023702	033674		1\$:	CMP	ABUFP,R2	:RECEIVED A CHAR?
1915	007052	001007				BNE	2\$	:YES-GO CHECK IT.
1916	007054	005337	002414			DEC	DLAY	:RUN TIME OUT DELAY.
1917	007060	001372				BNE	1\$	
1918	007062	005237	002374			INC	FTLCNT	:TIMED OUT-INCREMENT FATAL XMIT COUNT
1919	007066	104011				ERROR	11	:ISSUE HUNG MI-SSAGE AND EXIT.
1920	007070	000437				BR	CKEXT	
1921	007072	122777	000004	024574	2\$:	CMPB	#EOM,@ABUFP	:RECEIVED EOM CHAR?
1922	007100	001356				BNE	CKSRC	
1923	007102	042737	020000	002420		BIC	#REOM,VSTAT	:CLEAR THE EOM FLAG
1924	007110	032737	020000	002420		BIT	#REOM,VSTAT	:NOW WAIT FOR LAST EOM FLAG
1925	007116	001774				BEQ	.-6	:FROM XMIT TRANSMITTER CHECKSUM.
1926	007120	120477	010000			CMPB	R4,@RBBUF	:COMPARE 61 TO HOST CHECKSUM.
1927	007124	001421				BEQ	CKEXT	:EQUAL - EXIT TEST
1928	007126	004037	020074			JSR	R0,CLREG	
1929	007132	110437	001124			MOVB	R4,\$GDDAT	:LOAD THE HOST CALCULATED CHECKSUM
1930	007136	117737	007762	001126		MOVB	@RBBUF,\$BDDAT	:LOAD THE VT61 TRANSMITTED CHECKSUM
1931	007144	104014				ERROR	14	:ISSUE VT61 XMIT CHECKSUM ERROR
1932	007146	012746	060001			MOV	#60001,-(SP)	:PUSH #60001 ON STACK
1933	007152	004037	017732			JSR	R0,CKSFT	:CHECK FOR STATUS ERROR
1934	007156	000404				BR	CKEXT	
1935								
1936	007160	033	117	103	ITSUMA:	.BYTE	.ESC,.O,.DRECT,.ESC,.O,.CLRCK,0,0	
	007163	033	117	133				
	007166	000	000					
1937								
1938	007170	004037	020502		CKEXT:	JSR	R0,RESPTR	
1939								
1940								
1941								
1942								
1943								
1944								
1945								
1946								
1947								
1948								
1949								
1950								
(2)	007174	000004			TST1:	SCOPE		
(1)	007176	012737	000005	001160		MOV	#5,\$TIMES	:DO 5 ITERATIONS
(1)	007204	012737	007212	001106		MOV	#CURS1A,\$LPADR	:SET SCOPE LOOP ADDRESS
1951								
1952	007212	113737	001102	002424	CURS1A:	MOVB	\$TSTNM,TSTNM	:IN CASE OF ERROR - FOR TYPEOUT
1953	007220	013701	017460			MOV	TBBUF,R1	:LOAD XMIT BUFFER ADDRESS
1954	007224	004037	017470			JSR	R0,RESETV	:RESET THE UNIT AND WAIT FOR XON.
1955	007230	004037	022166			JSR	R0,LDBUF	:LOAD THE BUFFER WITH:
1956	007234	033	103	033		.BYTE	.ESC,.CRT,.ESC,.O,.RDCUR,.ESC,.CDWN,.ESC	
	007237	117	131	033				

1957	007242	102	033			
	007244	117	131	033	.BYTE	.O.,RDCUR,.ESC,.CLFT,.ESC,.O.,RDCUR
	007247	104	033	117		
	007252	131				
1958	007253	033	101	033	.BYTE	.ESC,.CUP,.ESC,.O.,RDCUR,.BEL,0
	007256	117	131	007		
	007261	000				
1959	007262	012737	000024	017466	MOV	#20,XMCNT ;SET TO XMIT 20 CHARACTERS
1960	007270	012737	000004	020364	MOV	#4,RECITT ;SET RECEIVE ITERATION TO 4
1961	007276	012737	033274	020366	MOV	#TCRLB+100,WDSTOR ;SET UP WORD STORAGE POINTER
1962	007304	004037	020116		JSR	R0,XMREC ;XMIT ,AND WAIT FOR REC.DONE
1963	007310	000402			BR	11\$;NORMAL EXIT
1964	007312	104011			ERROR	11 ;LAST XMIT CAUSED VT61 TO HANG.
1965	007314	000436			BR	CUR1XT ;EXIT TEST
1966	007316	012701	007402		MOV	#GDCURP,R1 ;R1=GOOD POSITION TABLE
1967	007322	012702	033274		MOV	#TCRLB+100,R2 ;R2=ACTUAL CURSOR POSITION
1968	007326	012703	002160		MOV	#CRT,R3 ;R3=CURSOR COMMAND TABLE
1969						
1970	007332	021112			12\$: CMP	(R1),(R2) ;COMPARE GOOD TO ACTUAL
1971	007334	001415			BEQ	2\$;OK-GO UPDATE POINTERS.
1972	007336	113737	002326	001125	MOVB	ESCN,\$GDDAT+1
1973	007344	111337	001124		MOVB	(R3),\$GDDAT ;LOAD COMMAND IN ESC ERROR
1974	007350	005037	001126		CLR	\$BDDAT
1975	007354	104001			ERROR	1 ;AND ISSUE IT
1976	007356	011237	032474		MOV	(R2),RCRLB ;LOAD BAD CURSOR POSITION
1977	007362	011146			MOV	(R1),-(SP) ;PUSH (R1) ON STACK
1978	007364	004037	020562		JSR	R0,CURER ;LOAD AND ISSUE CURSOR ERROR MESSAGE
1979	007370	022122			2\$: CMP	(R1)+,(R2)+ ;INCREMENT POSITION POINTERS.
1980	007372	022337	002166		CMP	(R3)+,CUP ;CHECK FOR COMMAND TERM.(CUP).
1981	007376	001355			BNE	12\$;NOT AT TERMINATOR-COMPARE AGAIN
1982	007400	000404			BR	CUR1XT ;EXIT TEST
1983						
1984						
1985	007402	020440			GDCURP: .WORD	20440 ;ROW 0, COL. 1
1986	007404	020441			.WORD	20441 ;ROW 1, COL. 1
1987	007406	020041			.WORD	20041 ;ROW 1, COL. 0
1988	007410	020040			.WORD	20040 ;ROW 0, COL. 0
1989	007412	000240			CUR1XT: NOP	
1990						
1991					;;*****	
1992						;ROUTINE TO INSURE THAT PEAD CHARACTER AT CURSOR
1993						;FUNCTIONS CORRECTLY. COMMAND SEQUENCE IS: RESET, A, CURSOR
1994						;LEFT, READ CHARACTER AT CURSOR.
1995						;AN ERROR IS REPORTED IF THE LAST READ IS NOT AN 'A'.
1996					;;*****	
1997						
1998					;;*****	
(2)	007414	000004			TST12: SCOPE	
(1)	007416	012737	000005	001160	MOV	#5,\$TIMES ;DO 5 ITERATIONS
(1)	007424	012737	007432	001106	MOV	#CURS1B,\$LPADR ;SET SCOPE LOOP ADDRESS
1999						
2000	007432	113737	001102	002424	CURS1B: MOVB	\$TSTNM,TSTNM ;IN CASE OF ERROR - FOR TYPEOUT
2001	007440	013701	017460		MOV	TBBUF,R1
2002	007444	004037	017470		JSR	R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
2003	007450	012721	000101		MOV	#1,(R1)+ ;A
2004	007454	113721	002326		MOVB	ESCN,(R1)+

```

2005 007460 113721 002164      MOV      CLFT,(R1)+      ;CURSOR LEFT
2006 007464 013721 002250      MOV      ESCOI,(R1)+
2007 007470 013711 002260      MOV      TCUCH,(R1)      ;TRANSMIT CH. AT CURSOR
2008 007474 012737 000006 017466  MOV      #6,XMCNT      ;SET UP TO XMIT 6 CHARACTERS
2009 007502 004037 020116      JSR      R0,XMREC      ;XMIT STRING AND WAIT FOR EOM.
2010 007506 000402          BR      10$      ;NORMAL EXIT
2011 007510 104011          ERROR 11      ;LAST XMIT CAUSED VT61 TO HANG/FAIL
2012 007512 000430          BR      2$      ;EXIT TEST
2013 007514 127727 007404 000101 10$: CMPB     ARBUF,#101      ;CHARACTER READ=A
2014 007522 001424          BFO      2$      ;YES-NEXT SUBTEST
2015 007524 013737 002250 001124      MOV      ESCOI,$GDDAT
2016 007532 000337 001124          SWAB     $GDDAT      ;REASSEMBLE ESC DATA
2017 007536 005037 001126          CLR      $BDDAT
2018 007542 113737 002260 001127      MOV      TCUCH,$BDDAT+1 ;LOAD FAILING ESC SEQUENCE
2019 007550 104001          ERROR 1      ;AND ISSUE IT
2020 007552 004037 020074          JSR      R0,CLREG
2021 007556 112737 000101 001124      MOV      #101,$GDDAT ;LOAD GOOD CH. AND CH.
2022 007564 117737 007334 001126      MOV      ARBUF,$BDDAT
2023 007572 104004          ERROR 4      ;READ AND ISSUE THEM.
2024
2025 007574 000240          2$: NOP      ;END OF TEST
2026
2027          ;:*****
2028          ;ROUTINE TO VERIFY OPERATION OF REPLACE AND INSERT MODE.
2029          ;INITIALLY ROW 0 IS WRITTEN TO 80 INCREMENTING CHAR.
2030          ;ON THE FIRST PASS(REPLACE MODE) A CHARACTER IS REPLACED
2031          ;AT HOME AND THE CHAR. AT ROW0,COL.0(172) AND ROW1,COL0(NULL)
2032          ;ARE VERIFIED. ON THE SECOND PASS, INSERT MODE IS ENTERED
2033          ;AND THE RESULTING INSERTION(AT HOME) IS VERIFIED.ROW0,COL0
2034          ;SHOULD BE 172 AND ROW1,COL0 SHOULD BE 161.
2035          ;:*****
2036          ;:*****
2037          ;:*****
2038          ;TST'3: SCOPE
2039          ;MOV      #5,$TIMES      ;;DO 5 ITERATIONS
2040          ;MOV      #INRPL,$LPADR   ;;SET SCOPE LOOP ADDRESS
2041
2042          INRPL: MOV      $TSTNM,TSTNM ;IN CASE OF ERROR - FOR PRINTOUT
2043          JSR      R0,RESETV      ;RESET THE UNIT
2044          MOV      TBBUF,R1
2045          INC      R1      ;LEAVE ROOM IN BUFFER FOR SOM.
2046          MOV      #80.,R3      ;CREATE A LINE OF 80 INCREMENTING
2047          JSR      R0,BLDINC      ;CHAR. ON THE SCREEN.
2048          CLRB     (R1)
2049          MOV      TBBUF,R3
2050          JSR      R0,LDXMIT
2051          CLR      R5      ;USE R5 AS TEST INDEXER.
2052          MOV      #2,RECITT      ;SET UP TO RECEIVE 2 CHAR.
2053          MOV      #TCRLB+200,BYSTOR ;SET UP STORAGE AREA.
2054          MOV      TBBUF,R1
2055          JSR      R0,LDBUF      ;LOAD THE BUFFER WITH:
2056          .BYTE     .ESC.,.CHOM,172,.ESC.,.CHOM,.ESC.,.O.,TCUCH
2057
2058          2053 007714 033 102 033      .BYTE     .ESC.,.CDWN,.ESC.,.O.,TCUCH,0
2059          007717 117 127 000
2060          2054 007722 012737 000015 017466      MOV      #13.,XMCNT      ;SET UP TO XMIT 13 CAHR.

```

```

2055 007730 004037 020116      JSR    RO,XMREC
2056 007734 000402      BR      1$          ;NORMAL EXIT
2057 007736 104011      ERROR   11          ;LAST XMIT CAUSED UNIT TO HANG.
2058 007740 000433      BR      INRXT      ;EXIT TEST.
2059 007742 026537 010020 033374 1$:    CMP    TDATA(R5),TCRLB+200 ;COMPARE GOOD TO REC.DATA.
2060 007750 001407      BEQ      2$          ;GOOD-LOOP OR EXIT.
2061 007752 016537 010012 001126      MOV    TFMCT(R5),SBDDAT
2062 007760 013737 002314 001124      MOV    ESCP,$GDDAT      ;LOAD ESCAPE SEQ. ERROR.
2063 007766 104001      ERROR   1
2064 007770 005725      2$:    TST     (R5)+      ;INCREMENT INDEXER.
2065 007772 020527 000004      CMP     R5,#4      ;THRU WITH TEST?
2066 007776 001414      BEQ      INRXT      ;YES-EXIT.
2067 010000 012703 010024      MOV     #ENSRT,R3      ;NO-SECOND PASS- ENTER
2068 010004 004037 020442      JSR     RO,LDXMIT      ;INSERT MODE AND DO AGAIN.
2069 010010 000723      BR      INAG
2070
2071 010012 000151 000111 177777  FUNCT: .WORD  .ERPL,.EINST,-1
2072 010020 172      000      172  TDATA: .BYTE  172,0,172,160
2073 010023 160
2074 010024 033      120      111  ENSRT: .BYTE  .ESC,.P,.EINST,0
2075 010027 000
2076 010030 000240      INRXT:  NOP
2077
2078 ;:*****
2079 ;ROUTINE TO INSURE VT61 WILL SCROLL IF A LINE FEED
2080 ;IS ISSUED FROM ROW 23 OR A DATA ENTRY FROM ROW23,COL. 79.
2081 ;IN SUBTEST A, ROW 0 IS INITIALLY WRITTEN TO A 0 AND ROW 1
2082 ;A 1. AFTER COMPLETION OF A LINE FEED(AND RESULTING SCROLL)
2083 ;ROW 00,COL.00 IS EXPECTED TO CONTAIN A 1.
2084 ;IN SUBTEST B, THE CURSOR IS PLACED AT ROW23,COL.79
2085 ;AND A DATA CHARACTER 'A' IS ENTERED. THE CURSOR
2086 ;POSITION IS THEN READ AND SHOULD BE ROW23,COL.00. THE
2087 ;CHARACTER AT HOME IS VERIFIED TO BE A NULL.
2088 ;:*****
2089 ;:*****
2090 ;TST14: SCOPE
2091 (2) 010032 000004
2092 (1) 010034 012737 000005 001160      MOV     #5,$TIMES      ;;DO 5 ITERATIONS
2093 (1) 010042 012737 010050 001106      MOV     #CKSCRA,$LPADR ;;;SET SCOPE LOOP ADDRESS
2094
2095 CKSCRA: MOVB    $TSTNM,TSTNM      ;IN CASE OF ERROR - FOR PRINTOUT
2096 010050 113737 001102 002424      JSR     RO,RSETV      ;RESET THE UNIT.
2097 010056 004037 017470      MOV     TBUF,R1
2098 010062 013701 017460      JSR     RO,LDBUF      ;LOAD THE XMIT BUFFER WITH:
2099 010066 004037 022166      .BYTE   60,.ESC,.CDWN,.ESC,.CLFT,61,.ESC,.Y,.R23,.COO
2100 010072 060      033      102
2101 010075 033      104      061
2102 010100 033      131      067
2103 010103 040
2104 010104 012      033      110
2105 010107 033      117      127
2106 010112 007      000
2107 010114 012737 000020 017466      MOV     #16,.XMCNT      ;SET UP TO XMIT 16 BYTES.
2108 010122 004037 020116      JSR     RO,XMREC
2109 010126 000402      BR      1$          ;NORMAL EXIT
2110 010130 104011      ERROR   11          ;LAST XMIT CAUSED UNIT TO HANG.
2111 010132 000452      BR      GDSCRL      ;EXIT TEST.

```

```

2101 010134 127727 006764 000061 1$: CMPB @RBBUF,#61 ;CHARACTER AT HOME A 1?
2102 010142 001401 BEQ CKSCRB ;YES-NEXT TEST
2103 010144 104023 ERROR 23 ;NO-ISSUE NO SCROLL ERROR.
2104 010146 012737 000002 020364 CKSCRB: MOV #2,RECITT ;SET UP FOR TWO REC. LOOPS.
2105 010154 012737 033374 020366 MOV #TCRLB+200,WDSIOR ;SET UP CURSOR POSITION STROAGE.
2106 010162 013701 017460 MOV TBBUF,R1
2107 010166 004037 022166 JSR R0,LDBUF ;LOAD XMIT BUFFER WITH:
2108 010172 033 131 067 .BYTE .ESC,,Y,,R23,,C79,101,.ESC,,O,,RDCUR
      010175 157 101 033
      010200 117 131
2109 010202 033 110 033 .BYTE .ESC,,CHOM,.ESC,,O,,TCUCH,0
      010205 117 127 000
2110 010210 012737 000015 017466 MOV #13,,XMCNT ;SET UP TO XMIT 13 BYTES.
2111 010216 004037 020116 JSR R0,XMREC ;XMIT AND WAIT FOR RECEIVED DONE.
2112 010222 000402 BR 1$
2113 010224 104011 ERROR 11 ;LAST XMIT CAUSED VT61 TO HANG.
2114 010226 000414 BR GDSCRL ;ERROR EXIT
2115 010230 127737 006670 002364 1$: CMPB @RBBUF,ZERO ;NULL RECEIVED?
2116 010236 001410 BEQ GDSCRL ;YES-EXIT TEST
2117 010240 104023 ERROR 23 ;NO-ISSUE NO SCROLL ERROR.
2118 010242 013777 033374 006654 MOV TCRLB+200,@RBBUF ;LOAD RECEIVED CURSOR POSITION.
2119 010250 013746 002340 MOV R23C00,-(SP) ;PUSH R23C00 ON STACK
2120 010254 004037 020562 JSR R0,CURER ;GO ISSUE CURSOR ERROR.
2121 010260 000240 GDSCRL: NOP
2122
2123
2124 ;:*****
2125 ;THIS TEST INSURES THAT THE VT61 CURSOR CAN BE
2126 ;POSITIONED TO VERY POSSIBLE ROW/COLUMN POSITON
2127 ;ON THE SCREEN. THIS IS TESTED BY FILLING THE
2128 ;COMPLETE SCREEN WITH A CHARACTER(A) AND THEN
2129 ;POSITIONING THE CURSOR (VIA DCA) TO EVERY POSITION
2130 ;AND THE 'A' AT THAT POSITION IS REPLACED WITH A SPACE.
2131 ;THE SCREEN IS THEN READ TO VERIFY THAT ONLY SPACES
2132 ;EXIST ON THE SCREEN. ALL POSITIONS CONTAINING
2133 ;NON-SPACES ARE REPORTED.
2134
2135 ;:*****
2136
2137 ;:*****
2138 (2) 010262 000004 TST15: SCOPE
2139 (1) 010264 012737 000001 001160 MOV #1,$TIMES ;DO 1 ITERATION
2140 (1) 010272 012737 010300 001106 MOV #CURS2,$LPADR ;SET SCOPE LOOP ADDRESS
2141 010300 113737 001102 002424 CURS2: MOVB $TSTNM,TSTNM ;IN CASE OF ERROR - FOR TYPEOUT
2142 010306 042737 077577 002420 BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2143 010314 004037 017470 JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
2144 010320 012702 003600 MOV #TOTCH,R2 ;LOAD A COUNT OF SCREEN(1920).
2145 010324 112777 000002 007132 MOVB #SOM,@TBUFP ;XMIT THE START OF MESSAGE.
2146 010332 004037 020372 JSR R0,XMIT1
2147 010336 005302 1$: DEC R2 ;DECREMENT XMIT COUNT
2148 010340 001413 BEQ 10$ ;COUNT ZERO?
2149 010342 112777 000101 007114 MOVB #101,@TBUFP ;LOAD THE CHARACTER(A).
2150 010350 004037 020372 JSR R0,XMIT1 ;NO-XMIT ANOTHER BYTE.
      010354 023737 002374 002376 CMP FTLCNT,ALWCNT ;EXCEEDED FATAL ERROR COUNT?

```

2151	010362	103765			BLO	1\$		:NO-CHECK IF XMIT COMPLETE.
2152	010364	000137	010754		JMP	C2XT		:YES-GO ABORT TEST.
2153	010370	004037	020502		JSR	R0,RESPTR		:RESET INTERRUPT POINTERS.
2154	010374	013737	002354	020764	MOV	R23C78,LNRW		:SET UP 1ST ADDRESS
2155	010402	013701	017460		MOV	TBBUF,R1		:LOAD XMIT BUFFER WITH
2156	010406	013721	002234		MOV	DCRAD,(R1)+		
2157	010412	010102			MOV	R1,R2		:R2 POINTS TO CURSOR ADD. IN BUFFER
2158	010414	013721	002354		MOV	R23C78,(R1)+		:CURSOR TO LOWER RIGHT -1.
2159	010420	112721	000040		MOVB	#40,(R1)+		:SPACE
2160	010424	012737	000005	017466	MOV	#5,XMCNT		:SET UP TO XMIT 5 CHARACTERS
2161	010432	042737	077577	002420	BIC	#77577,VSTAT		:CLEAR ALL FLAGS BUT XOFF AND XMKIL
2162	010440	052777	040000	171444	BIS	#XENA,@VZCSR		:XMIT INTERRUPTS.
2163	010446	012746	000001		MOV	#XMDNE,-(SP)		:PUSH #XMDNE ON STACK
2164	010452	012746	000002		MOV	#2,-(SP)		:PUSH #2 ON STACK
2165	010456	004037	023074		JSR	R0,WTBGND		:LOOK FOR XMIT DONE
2166	010462	000534			BR	C2XT		:NOT FOUND-ERROR EXIT
2167	010464	021237	002350		CMP	(R2),CUHME		:DELETED TO HOME?
2168	010470	001405			BEQ	3\$		:YES
2169	010472	004037	020660		JSR	R0,CMPOS		:NO-GET NEXT POSITION TO BE DELETED
2170	010476	013712	020764		MOV	LNRW,(R2)		:LOAD IT IN XMIT BUFFER
2171	010502	000750			BR	2\$		:AND DELETE IT.
2172	010504	004037	020502		JSR	R0,RESPTR		:RESET INTERRUPT POINTERS
2173	010510	013737	002350	020764	MOV	CUHME,LNRW		:LOAD INITIAL CHECK POSITION(HOME)
2174	010516	013701	017460		MOV	TBBUF,R1		:LOAD XMIT BUFFER WITH
2175	010522	010102			MOV	R1,R2		:STORE ERRORS IN XMIT BUFFER
2176	010524	042737	077577	002420	BIC	#77577,VSTAT		:CLEAR ALL FLAGS BUT XOFF AND XMKIL
2177	010532	012737	001001	002422	MOV	#1001,BLKM		:SET XMIT TO SOM- DATA -EOM.
2178	010540	013721	002326		MOV	ESCN,(R1)+		
2179	010544	013721	002156		MOV	CHOM,(R1)+		:CURSOR HOME
2180	010550	013721	002250		MOV	ESCO,(R1)+		
2181	010554	013721	002252		MOV	XMTAL,(R1)+		:TRANSMIT ALL
2182	010560	012737	000005	017466	MOV	#5,XMCNT		
2183	010566	052777	040000	171316	BIS	#XENA,@VZCSR		:SET XMIT ENABLE
2184	010574	012746	000001		MOV	#XMDNE,-(SP)		:PUSH #XMDNE ON STACK
2185	010600	012746	000003		MOV	#3,-(SP)		:PUSH #3 ON STACK
2186	010604	004037	023074		JSR	R0,WTBGND		:LOOK FOR SOM OR XMIT DONE
2187	010610	000461			BR	C2XT		:NOT FOUND-ERROR EXIT
2188	010612	013701	017130		MOV	RBUF,R1		:SET UP RECEIVE FLAG
2189	010616	005037	002414		CLR	DLAY		:SET UP TIME OUT DELAY
2190	010622	020137	017130		CMP	R1,RBUF		:CHARACTER RECEIVED?
2191	010626	103411			BLO	41\$		:YES-GO CHECK IT.
2192	010630	032737	020000	002420	BIT	#REOM,VSTAT		:LOOK FOR END OF MESSAGE
2193	010636	001025			BNE	C2CK		:FOUND IT, EXIT TEST
2194	010640	005337	002414		DEC	DLAY		:RUN TIME OUT DELAY.
2195	010644	001366			BNE	40\$		:AND LOOK FOR RECEIVED CH.
2196	010646	104011			ERROR	11		:LAST XMIT CAUSED VT61 TO HANG.
2197	010650	000420			BR	C2CK		:GO SEE IF ANY ERRORS STORED.
2198	010652	013737	017124	017130	MOV	RBUF,RBUF		:RESET RECEIVE POINTER
2199	010660	127727	006240	000040	CMPS	@RBUF,#40		:CHAR EQUAL A SPACE?
2200	010666	001003			BNE	6\$		:NOT A SPACE-MUST BE ERROR-STORE IT
2201	010670	004037	020722		JSR	R0,CPPOS		:UPDATE CURSOR POSITION
2202	010674	000746			BR	4\$		
2203	010676	022702	033220		CMP	#TCRLB+20.,R2		:STORED 10 ERRORS?
2204	010702	101772			BLOS	5\$		:YES-IGNORE ANY FURTHER ERRORS.
2205	010704	013722	020764		MOV	L1,W,(R2)+		:STORE FAILING CURSOR POSITION
2206	010710	000767			BR	5\$		

```

2207
2208 010712 020237 017460 C2CK: CMP R2,TBBUF ;ANY ERRORS STORED?
2209 010716 001416 BEQ C2XT ;NO EXIT TEST
2210 010720 013701 017460 MOV TBBUF,R1 ;USE R1 AS ERROR POINTER
2211 010724 021137 002236 1$: CMP (R1),R23C79 ;CURSOR TO LOWER RIGHT?
2212 010730 001411 BEQ C2XT ;YES-NOT AN ERROR.
2213 010732 104012 ERROR 12 ;NO-ISSUE ERROR MESSAGES
2214 010734 012746 020040 MOV #20040,-(SP) ;PUSH #20040 ON STACK
2215 010740 012177 006160 MOV (R1)+,@RBBUF ;LOAD FAILING POS.
2216 010744 004037 020562 JSR R0,CURER ;ISSUE CURSOR ERROR
2217 010750 020102 CMP R1,R2 ;DONE WITH ERRORS?
2218 010752 103764 BLO 1$ ;NO, DUMP ANOTHER.
2219 010754 000240 C2XT: NOP ;EXIT TEST
2220
2221 ;*****
2222 ;ROUTINE TO INSURE PROPER OPERATION OF CARRIAGE RETURN
2223 ;AND LINE FEED DURING NORMAL MODE. INITIALLY THE CURSOR IS
2224 ;SET(VIA D.C.A.) TO ROW0,COL 20 AND A LINE FEEL IS ISSUED
2225 ;THE CURSOR POSITION IS THEN READ AND MUST BE ROW1,COL20.
2226 ;A CARRIAGE RETURN IS THEN ISSUED AND CURSOR POSITION VERIFIED
2227 ;TO BE ROW1,COL0.
2228
2229 ;*****
2230
2231 ;*****
2232
2233 (2) 010756 000004 TST16: SCOPE
2234 (1) 010760 012737 000005 001160 MOV #5,$TIMES ;DO 5 ITERATIONS
2235 (1) 010766 012737 010774 001106 MOV #NWLN,$LPADR ;SET SCOPE LOOP ADDRESS
2236
2237 010774 113737 001102 002424 NWLN: MOVR $TSTNM,TSTNM ;IN CASE OF ERROR - FOR PRINTOUT
2238 011002 004037 017470 JSR R0,RESETV ;RESET THE UNIT AND ENTER MAINT.MODE
2239 011006 013701 017460 MOV TBBUF,R1
2240 011012 004037 022166 JSR R0,LDBUF ;LOAD XMIT BUFFER WITH-
2241 011016 033 131 040 .BYTE .ESC,.Y,.R00,.C20
2242 011021 064 .BYTE .LNFED,.ESC,.O,.RDCUR,.BEL,0
2243 011022 012 033 117
2244 011025 131 007 000
2245 011030 012737 000011 017466 MOV #9,XMCNT ;SETUP TO XMIT 9 CHARACTERS
2246 011036 004037 020116 JSR R0,XMREC ;GO DO IT
2247 011042 000402 BR 30$ ;NORMAL EXIT.
2248 011044 104011 ERROR 11 ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
2249 011046 000454 BR 4$ ;EXIT TEST
2250 011050 023777 002332 006046 30$: CMP R01C20,@RBBUF ;CHECK CURSOR POS. S/B ROW 1, COL 20.
2251 011056 001412 BEQ 3$
2252 011060 005037 001124 CLR $GDDAT
2253 011064 013737 002150 001126 MOV LNFED,$BDDAT
2254 011072 104001 ERROR 1 ;ISSUE IT
2255 011074 013746 002332 MOV R01C20,-(SP) ;PUSH R01C20 ON STACK
2256 011100 004037 020562 JSR R0,CURER ;SFTUP AND ISSUE CURSOR ERROR
2257 011104 013701 017460 3$: MOV TBBUF,R1
2258 011110 013721 002146 MOV CARRT,(R1)+ ;LOAD XMIT BUFFER WITH
2259 011114 013721 002250 MOV ESCOI,(R1)+ ;CARRIAGE RETURN, READ CURSOR
2260 011120 013721 002244 MOV RDCUR,(R1)+ ;POSITION
2261 011124 012737 000004 017466 MOV #4,XMCNT ;SET UP TO TRANSMIT 4 CHARACTERS
2262 011132 004037 020116 JSR R0,XMREC ;GO DO IT
2263 011136 000402 BR 40$ ;NORMAL EXIT.

```

```

2258 011140 104011 ERROR 11 ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
2259 011142 000416 BR 4$ ;EXIT TEST
2260 011144 023777 002330 005752 40$: CMP R01C00,@RBBUF ;CHECK CURSOR POS. S/B ROW1, COL 0.
2261 011152 001412 BEQ 4$ ;EXIT TEST IF GOOD.
2262 011154 005037 001124 CLR $GDDAT
2263 011160 013737 002146 001126 MOV CARRT,$BDDAT
2264 011166 104001 ERROR 1 ;ISSUE IT
2265 011170 013746 002330 MOV R01C00,-(SP) ;PUSH R01C00 ON STACK
2266 011174 004037 020562 JSR R0,CURER ;SET UP AND ISSUE CURSOR ERROR
2267 011200 000240 4$: NOP
2268
2269
2270 ;*****
2271 ;ROUTINE TO VERIFY PROPER OPERATION OF ERASE TO END-OF-
2272 ;SCREEN. SCREEN IS WRITTEN TO 1920 INCREMENTING CHAR.
2273 ;ERASE TO END OF SCREEN IS THEN ISSUED AND THE
2274 ;ENTIRE SCREEN IS READ VERIFYING THAT IT IS ALL NULLS.
2275 ;*****
2276
2277 ;*****
2278 ;*****
2279 (2) 011202 000004 TST17: SCOPE
2280 (1) 011204 012737 000003 001160 MOV #3,$TIMES ;DO 3 ITERATIONS
2281 (1) 011212 012737 011220 001106 MOV #ERSE,$LPADR ;SET SCOPE LOOP ADDRESS
2282
2283 011220 113737 001102 002424 ERSE: MOVB $TSTNM,TSTNM ;IN CASE OF ERROR - FOR PRINTOUT
2284 011226 004037 017470 JSR R0,RESETV ;RESET THE UNIT -SET MAINT. MODE.
2285 011232 005077 005666 CLR @RBBUF ;CLEAR THE CHECK LOCATION.
2286 011236 004037 021470 JSR R0,DATSC ;FILL THE SCREEN.
2287 011242 013701 017460 MOV TBBUF,R1
2288 011246 004037 022166 JSR R0,LDBUF ;LOAD XMIT BUFFER WITH:
2289 011252 033 110 033 .BYTE .ESC,.CHOM,.ESC,.EOS,.ESC,.O,.XMTAL,0
2290 011255 112 033 117
2291 011260 126 000
2292 011262 113737 002326 001125 MOVB ESCN,$GDDAT+1
2293 011270 113737 002170 001124 MOVB EOS,$GDDAT ;LOAD ERROR WITH ERASE TO EOS
2294 011276 005037 001126 CLR $BDDAT
2295 011302 005077 005616 CLR @RBBUF
2296 011306 012737 000007 017466 MOV #7,XMCNT ;SET UP TO XMIT 7 BYTES
2297 011314 004037 020116 JSR R0,XMREC ;XMIT AND WAIT FOR REC. DONE
2298 011320 000402 BR 5$
2299 011322 104011 ERROR 11 ;ESC ERROR
2300 011324 000413 BR ERSXT ;EXIT TEST
2301 011326 127737 005572 002364 5$: CMPB @RBBUF,ZERO ;VT61 XMITTED SOM/EOM ONLY?
2302 011334 001407 BEQ ERSXT ;YES-EXIT TEST.
2303 011336 104001 ERROR 1 ;NO-ERASE TO END OF SCREEN
2304 011340 004037 020074 JSR R0,CLREG ;GO CLEAR ERROR STORAGE
2305 011344 117737 005554 001126 MOVB @RBBUF,$BDDAT
2306 011352 104004 ERROR 4 ;ISSUE DATA ERROR
2307 011354 000240 ERSXT: NOP
2308 ;*****
2309 ;ROUTINE TO SET UP END OF PASS INDICATION.

```

```

2309      ;SELF TEST(ESC O T) IS ISSUED TO THE UNIT UNDER TEST
2310      ;AND AN ERROR IS ISSUED IF THE UNIT CANNOT RESPOND AFTER
2311      ;SELF TEST IS COMPLETE. IF SELF TEST IS SUCCESSFUL THE
2312      ;SCREEN IS WRITTEN TO 23 LINES OF INCREMENTING CHARACTERS
2313      ;AND 23 LINES OF INCREMENTING CHAR. IN REVERSE VIDEO.
2314      ;THE IDENT IS THEN CHECKED AND IF A COPIER IS PRESENT A
2315      ;COPY SCREEN COMMAND IS ISSUED(NOTE: THIS COMMAND WILL CAUSE
2316      ;THE UNIT TO BE 'BUSY' AND NOT RESPOND TO ANY FURTHER COMMANDS
2317      ;UNTIL THE SCREEN HAS BEEN COMPLETELY COPIED.)
2318      ;:*****
2319      ;:*****
2320      ;:*****
2321      (2) 011356 000004
2322      (1) 011360 012737 000001 001160 TST20: SCOPE
2323      (1) 011366 012737 011374 001106 MOV #1,$TIMES ;:DO 1 ITERATION
2324      MOV #LSTST,$LPADR ;:SET SCOPE LOOP ADDRESS
2325      LSTST: MOV $TSTNM,$TSTNM ;:IN CASE OF ERROR - FOR PRINTOUT
2326      MOV ZERO,-(SP) ;:PUSH ZERO ON STACK
2327      MOV TSTER,-(SP) ;:PUSH TSTER ON STACK
2328      MOV ESCO,-(SP) ;:PUSH ESCO ON STACK
2329      JSR RO,TESC ;:TRANSMIT IT.
2330      JSR RO,GETON ;:GO LOOK FOR A XON.
2331      BR 1$ ;:VT61 RESPONDED-NOT HUNG
2332      MOV VZCSR,$GDDAT ;:LOAD THE ADDRESS
2333      MOV VECT,$BDDAT ;:LOAD THE VECTOR
2334      ERROR 10 ;:REPORT SELF TEST FAILURE
2335      JSR RO,RESETV ;:RESET AND SET MAINT. MODE.
2336      CLR BUBCT ;:SET UP HALF-SCREEN FLAG.
2337      BIC #77577,VSTAT ;:CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2338      MOV TBBUF,R1 ;:SET UP BEG. OF XMIT BUFFER
2339      MOV #320,R3 ;:FILL BUFFER WITH INCREMENTING CHAR.
2340      JSR RO,BLDINC
2341      MOV #1001,BLKM ;:SET XMIT TO SOM- DATA -EOM.
2342      MOV #960,XMCNT ;:SEND 12 LINE TO VT61
2343      BIS #XENA,@VZCSR ;:ENABLE XMIT INTERRUPTS
2344      MOV #XMDNE,-(SP) ;:PUSH #XMDNE ON STACK
2345      MOV #240,-(SP) ;:PUSH #240 ON STACK
2346      JSR RO,WTBGND ;:LOOK FOR XMDNE.
2347      BR ENDSEL ;:NOT FOUND-EXIT.
2348      TST BUBCT ;:DONE WITH SCREEN?
2349      BNE 3$ ;:YES-EXIT
2350      MOV #SETREV,R3 ;:NO-ISSUE ENTER REVERSE VIDEO
2351      JSR RO,LDXMIT ;:ESCAPE SEQUENCE.
2352      INC BUBCT ;:INCREMENT SCREEN HALF FLAG.
2353      BR 2$ ;:AND ISSUE SECOND HALF IN REV. VIDEO.
2354      BIT #BIT00,IDENT ;:IDENT = COPIER?
2355      BEQ ENDSEL ;:NO
2356      MOV ZERO,-(SP) ;:PUSH ZERO ON STACK
2357      MOV #CPYSC,-(SP) ;:PUSH #CPYSC ON STACK
2358      MOV ESCN,-(SP) ;:PUSH ESCN ON STACK
2359      JSR RO,TESC
2360      JSR RO,CKABRT ;:CHECK FOR A PERIPHERAL ABORT.
2361      TSTB MODE ;:IF IN MAN MODE DO NOT ENTER EOP.
2362      BEQ 1$
2363      JMP ACTRT
2364      1$: JMP MODCA ;:CHECK FOR MORE LINES BEFORE EOP
  
```

```

2362 011636
2363
(1)
(2)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1) 011636
(1) 011636 000004
(1) 011640 005037 001102
(1) 011644 005037 001160
(1) 011650 005237 001100
(1) 011654 042737 100000 001100
(1) 011662 005327
(1) 011664 000001
(1) 011666 003031
(1) 011670 012737
(1) 011672 000001
(1) 011674 011664
(1) 011676 104401 012021
(2) 011702 013746 001100
(2) 011706 104405
(1) 011710 104401 012016
(1) 011714 013700 000042
(1) 011720 001414
(1) 011722 005046
(1) 011724 012746 011732
(1) 011730 000426
(1)
(1) 011732
(1) 011732 013700 000042
(1) 011736 001405
(1) 011740 000005
(1) 011742 004710
(1) 011744 000240
(1) 011746 000240
(1) 011750 000240
(1) 011752
(2) 011752 104400
(1) 011754 042716 000020
(1) 011760 032777 010000 167152
(1) 011766 001005
(1) 011770 005137 012014
(1) 011774 100402
(1) 011776 052716 000020
(1) 012002 012746 012010
(1) 012006 000002
(1)
(1)
(1) 012010
(1) 012010 000137
(1) 012012 003102
  
```

ENDPS:
 .SBTTL END OF PASS ROUTINE

 *INCREMENT THE PASS NUMBER (\$PASS)
 *INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
 *TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
 *IF SW12=1 INHIBIT TRACE TRAP
 *IF THERES A MONITOR GO TO IT
 *IF THERE ISN'T JUMP TO MODCK

\$EOP:

SCOPE
 CLR \$STNM ;;ZERO THE TEST NUMBER
 CLR \$TIMES ;;ZERO THE NUMBER OF ITERATIONS
 INC \$PASS ;;INCREMENT THE PASS NUMBER
 BIC #100000,\$PASS ;;DON'T ALLOW A NEG. NUMBER
 DEC (PC)+ ;;LOOP?

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

\$ENDCT: .WORD 1
 \$EOPCT
 TYPE \$ENDMG ;;TYPE 'END PASS #'
 MOV \$PASS,-(SP) ;;SAVE \$PASS FOR TYPEOUT
 TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
 TYPE \$ENULL ;;TYPE A NULL CHARACTER
 \$GET42: MOV a#42,R0 ;;GET MONITOR ADDRESS
 BEQ \$DOAGN ;;BRANCH IF NO MONITOR
 CLR -(SP) ;;INSURE THE 'T' BIT IS CLEAR
 MOV #CLR.T,-(SP) ;;SETUP FOR AN RTI OR RTT
 BR \$RTN ;;GO DO AN RTI OR RTT TO LOAD THE PSW
 ;;WITH A CLEARED 'T' BIT

\$CLR.T:
 MOV a#42,R0
 BEQ \$DOAGN
 RESET

\$ENDAD: JSR PC,(R0) ;;INSURE R0 CONTAINS THE MONITORS
 ;;RETURN ADDRESS
 ;;CLEAR THE WORLD
 ;;GO TO MONITOR
 ;;SAVE ROOM
 ;;FOR
 ;;ACT11

\$DOAGN:
 TRAP
 BIC #20,(SP) ;;PUSH OLD PSW AND PC ON STACK
 BIT #B112,aSWR ;;CLEAR THE 'T' BIT
 BNE 1\$;;RUN WITH TRACE TRAP?
 COM \$TBIT
 BMI 1\$;;BR IF NO
 BIS #20,(SP) ;;IS IT TIME FOR TRACE TRAP
 1\$: MOV #1,\$LOOP
 \$RTN: RTI ;;BR IF NO
 ;;SET TRACE TRAP
 ;;JUMP TO START OF TEST
 ;;RETURN--THIS IS CHANGED TO
 ;;AN 'RTT' IF 'RTT' IS A LEGAL
 ;;INSTRUCTION

\$LOOP:
 JMP a(PC)+
 \$RTNAD: .WORD MODCK ;;RETURN

```

(1) 012014 000000 $TBIT: .WORD 0 ;:'T' BIT STATE INDICATOR
(1) 012016 377 377 000 $ENULL: .BYTE -1,-1,0 ;:NULL CHARACTER STRING
(1) 012021 015 042412 042116 $ENDMG: .ASCIZ <15><12>/END PASS #/
(1) 012026 050040 051501 020123
(1) 012034 000043

2364 ;:*****
2365 ;:ROUTINE TO ECHO THE KEYBOARD. KEYS FOR TAB,BELL,CARRIAGE
2366 ;:AND LINE FEED ECHO A MNEMONIC, NON-DISPLAY CHAR. ECHO OCTAL
2367 ;:EQUIVALENTS AND DISPLAY CHAR. ECHO THEMSELVES.
2368 ;:(EXAMPLES-CHAR.,SPACE,ESC,SPACE OR 037,SPACE.) A
2369 ;:CONTROL C (003) WILL CAUSE A TEST EXIT.
2370
2371 ;:*****
2372
2373 ;:*****
(2) 012036 000004 TST21: SCOPE
(1) 012040 012737 000001 001160 MOV #1,$TIMES ;:DO 1 ITERATION
(1) 012046 012737 012054 001106 MOV #KYBD,$LPADR ;:SET SCOPE LOOP ADDRESS

2374
2375 012054 113737 001102 002424 KYBD: MOVB $TSTNM,TSTNM ;:IN CASE OF ERROR - FOR PRINTOUT
2376 012062 004037 020502 JSR R0,RESPTR
2377 012066 012702 030476 MOV #DKYBD,R2 ;:LOAD MESSAGE ADDRESS INR2
2378 012072 004037 021536 JSR R0,DSMES ;:DISPLAY KEYBOARD MESSAGE
2379 012076 012703 031064 MOV #DCNTZ,R3 ;:ISSUE CONTROL C EXIT MESSAGE
2380 012102 004037 020442 JSR R0,LDXMIT
2381 012106 012703 012344 MOV #EXMAIN,R3
2382 012112 004037 020442 JSR R0,LDXMIT ;:ISSUE EXIT MAINTENANCE MODE.
2383 012116 042737 077577 002420 KYSTRT: BIC #77577,VSTAT ;:CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2384 012124 105777 021544 TSTB @ABUFF ;:SEE IF A CHAR. RECEIVED
2385 012130 001001 BNE 11$ ;:YES-GO PROCESS IT
2386 012132 000001 WAIT ;:WAIT FOR A CH.
2387 012134 117701 021534 11$: MOVB @ABUFF,R1 ;:GET RAW RECEIVED DATA
2388 012140 004037 023020 JSR R0,EXTST ;:CHECK FOR EXIT CONDITIONS
2389 012144 000402 BR 10$ ;:NO EXIT -CONTINUE.
2390 012146 000137 003760 JMP ASTRT ;:EXIT TEST 4
2391 012152 105077 021516 10$: CLRB @ABUFF ;:CLEAR CHAR FROM BUFFER
2392 012156 032737 000400 002420 BIT #ESC,VSTAT ;:CHAR.=ESC(033)?
2393 012164 001405 BEQ 12$ ;:NO
2394 012166 005037 017132 CLR ESAMB ;:YES - RESET ESC ASSEMBLY FLAG
2395 012172 012703 030471 MOV #DESC,R3 ;:LOAD ESC MESSAGE ADDRESS
2396 012176 000454 BR KYBXMT
2397 012200 120127 000041 12$: CMPB R1,#41 ;:CHAR. LESS THAN 41 OR
2398 012204 103415 BLO 2$ ;:HIGHER THAN 176, GO ECHO
2399 012206 120127 000176 CMPB R1,#176 ;:OCTAL EQUIVALENT
2400 012212 101012 BHI 2$
2401 012214 110177 005244 MOVB R1,@TBUF ;:LOAD CHAR. IN XMIT BUFF.
2402 012220 004037 020372 JSR R0,XMIT1 ;:GO XMIT IT
2403 012224 112777 000040 005232 MOVB #40,@TBUF ;:LOAD A SPACE
2404 012232 004037 020372 JSR R0,XMIT1 ;:AND XMIT IT.
2405 012236 000727 BR KYSTRT
2406 012240 120137 002144 2$: CMPB R1,BEL ;:CHAR.=BELL?
2407 012244 001003 BNE 3$
2408 012246 012703 030745 MOV #DBELL,R3 ;:LOAD BELL MESSAGE ADDRESS
2409 012252 000426 BR KYBXMT
2410 012254 120137 002152 3$: CMPB R1,TAB ;:CHAR. -TAB?
2411 012260 001003 BNE 4$
  
```

```

2412 012262 012703 030726      MOV    #DTAB,R3      ;YES-ECHO 'TAB'
2413 012266 000420      BR      KYBXMT
2414 012270 123701 002146      4$:    CMPB   CARRT,R1      ;CHAR.=CARRIAGE RETURN?
2415 012274 001003      BNE     5$
2416 012276 012703 030733      MOV    #DCR,R3      ;YES - ECHO 'C/R'.
2417 012302 000412      BR      KYBXMT
2418 012304 120137 002150      5$:    CMPB   R1,LNFED      ;CHAR.=LINE FEED?
2419 012310 001003      BNE     6$
2420 012312 012703 030740      MOV    #DLF,R3      ;NO CHECK FOR CONTROL Z
2421 012316 000404      BR      KYBXMT      ;YES - ECHO 'L/F'.
2422 012320 004037 021632      6$:    JSR     R0,BINOC      ;CONVERT BINARY TO OCTAL
2423 012324 012703 002360      MOV    #SVER1,R3
2424 012330 042737 077577 002420 KYBXMT: BIC    #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2425 012336 004037 020442      JSR     R0,LDXMIT      ;GO XMIT BUFFER
2426 012342 000665      BR      KYSTRT      ;WAIT FOR NEXT CHAR.
2427
2428      ;SEQUENCE TO EXIT MAINTENANCE MODE.
2429 012344 033 117 141 EXMAIN: .BYTE .ESC,.O,.DMAIN,0
2430 012347 000
2431
2432      ;*****
2433      ;ROUTINE TO UTILIZE THE VT61 AS A PRINTER CONTROLLER.
2434      ;AFTER TEST MESSAGE IS DISPLAYED, THE TEST WAITS
2435      ;FOR A C/R BEFORE ACTUALLY ENTERING TEST. A PATTERN
2436      ;OF INCREMENTING, ROLLING CHAR. WILL BE OUTPUTTED UNTIL A
2437      ;CONTROL C(003) IS RECEIVED.
2438      ;*****
2439
2440      ;*****
2441      ;*****
2442      ;*****
2443      ;*****
2444      ;*****
2445      ;*****
2446      ;*****
2447      ;*****
2448      ;*****
2449      ;*****
2450      ;*****
2451      ;*****
2452      ;*****
2453      ;*****
2454      ;*****
2455      ;*****
2456      ;*****
2457      ;*****
2458      ;*****
2459      ;*****
2460      ;*****
2461      ;*****
2462      ;*****
  
```

```

2463 012524 001774      BEQ      .-6
2464 012526 004037 023020 JSR      R0,EXTST      ;CHECK FOR EXIT REQUEST.
2465 012532 000402      BR       6$      ;NO-CONTINUE
2466 012534 000137 003760 JMP      ASTRT      ;YES-EXIT TEST!!
2467 012540 004037 022174 6$: JSR      R0,CKABRT    ;CHECK FOR A PERIPHERAL ABORT.
2468 012544 122705 000177 CMBB     #177,R5      ;EXCEEDED PATT. LIMIT?
2469 012550 001337      BNE      5$      ;NO
2470 012552 000734      BR       4$      ;YES RESET IT
2471
2472 ;:*****
2473
2474 ;ROUTINE TO LOOP DATA/COMMANDS FROM THE VT61 BACK TO
2475 ;THE VT61. DATA TRANSMISSIONS RESULTING FROM A ESC
2476 ;SEQUENCE WILL ALSO BE LOOPED AND WILL ENTER THE SCREEN
2477 ;AT THE CURSOR POSITION.THIS TEST CAN BE USED TO SIMULATE,
2478 ;OR CREATE, SPECIFIC SCREEN PATTERNS AND OPERATIONS.
2479 ;:*****
2480
2481 ;:*****
2482
2483 (2) 012554 000004      TST23: SCOPE
2484 (1) 012556 012737 000001 001160 MOV      #1,$TIMES      ;;DO 1 ITERATION
2485 (1) 012564 012737 012572 001106 MOV      #LPTST,$LPADR  ;;SET SCOPE LOOP ADDRESS
2486
2487 012572 113737 001102 002424 LPTST: MOVB     $TSTNM,TSTNM  ;IN CASE OF ERROR - FOR PRINTOUT
2488 012600 004037 020502 JSR      R0,RESPTR     ;RESET POINTERS
2489 012604 012702 030753 MOV      #DLOOP,R2     ;LOAD LOOP MESSAGE ADDRESS
2490 012610 004037 021536 JSR      R0,DSMES      ;DISPLAY IT
2491 012614 012703 012344 MOV      #EXMAIN,R3
2492 012620 004037 020442 JSR      R0,LDXMIT     ;ISSUE EXIT MAINTENANCE MODE.
2493 012624 004037 022570 JSR      R0,LOOP      ;GO LOOP VT61
2494 012630 000137 003760 JMP      ASTRT      ;ENTER MAN MODE VIA SCOPE ROUTINE.
2495
2496 ;:*****
2497
2498 ;PRODUCTION KEYBOARD TEST. ALL KEYS MUST BE DEPRESSED
2499 ;IN THE SEQUENCE INDICATED ON THE SCREEN. ALL ERRORS
2500 ;OR MISTAKES ARE DISPLAYED IN OCTAL POSITIONAL FORMAT AND THE
2501 ;CORRECT KEY POSITION IN THE ROW IS DISPLAYED IN DECIMAL.
2502 ;THIS TEST IS RUN IN MAINTENANCE MODE, THEREFORE THE KEYS
2503 ;WILL ECHO THEIR POSITION,NOT THEIR INDICATED MNEMONIC. 10
2504 ;ERRORS WILL CAUSE AN AUTOMATIC EXIT FROM TEST.
2505 ;:*****
2506
2507 (2) 012634 000004      TST24: SCOPE
2508 (1) 012636 012737 000001 001160 MOV      #1,$TIMES      ;;DO 1 ITERATION
2509 (1) 012644 012737 012652 001106 MOV      #PDKBD,$LPADR  ;;SET SCOPE LOOP ADDRESS
2510
2511 012652 113737 001102 002424 PDKBD: MOVB     $TSTNM,TSTNM  ;IN CASE OF ERROR - FOR PRINTOUT
2512 012660 012702 031346 MOV      #DKBD ,R2
2513 012664 004037 021536 JSR      R0 ,DSMES      ;DISPLAY KEYBOARD TEST MESSAGE.
2514 012670 005037 002404 CLR      BUBCT      ;CLEAR ERROR COUNT LOCATION.
2515 012674 005005      CLR      R5
2516
2517 012676 016504 013022      DOAROW: MOV      DTTBL(R5),R4  ;SET UP 'GOOD' CHAR. POINTER

```

```

2513 012702 016503 012774      MOV      MSTBL(R5),R3
2514 012706 001414      BEQ      FEXIT
2515 012710 100421      BMI      CLMAIN      ;MESSAGE WAS ZERO-EXIT.
2516 012712 004037 020442      JSR      R0,LDXMIT      ;IF MESSAGE IS -1,CLEAR MAINT. MODE.
2517 012716 004037 022322      JSR      R0,CKKBD      ;ISSUE 'ROW OR FUNCTION' MESSAGE.
2518 012722 123727 002404 000012  CMPB     BUBCT,#10.      ;GO CHECK IT.
2519 012730 103401      BLO      1$      ;TEN ERROR EXIT?
2520 012732 000402      BR       FEXIT      ;NO-CONTINUE.
2521 012734 005725      1$:      TST      (R5)+      ;YES-EXIT TEST.
2522 012736 000757      BR       DOAROW      ;INCREMENT OFFSET.
2523 012740 012702 031115      FEXIT:  MOV      #DEXT,R2      ;NO-DO NEXT ROW/FUNCTION.
2524 012744 004037 021536      JSR      R0,DSMES      ;ISSUE EXIT MESSAGE
2525 012750 000137 003760      JMP      ASTRT
2526 012754 012703 012770      CLMAIN: MOV      #RSMAIN,R3      ;SET UP TO EXIT MAINT. MODE.
2527 012760 004037 020442      JSR      R0,LDXMIT
2528 012764 005725      TST      (R5)+      ;INCREMENT OFFSET.
2529 012766 000743      BR       DOAROW      ;NOW TEST CONTROL AND SHIFT FUNCTIONS.
2530 012770 033      117      141  RSMAIN: .BYTE      .ESC,.0,.DMAIN,0
2531 012773 000

```

;TABLE OF MESSAGE ADDRESSES.

```

2535 012774 031551 031656 031713  MSTBL:  .WORD      DTOP,DSEC,DTHRD,DBOT
2536 013002 032042
2536 013004 032120 032144 177777      .WORD      DSPCE,DKPD,-1,DCONT,DL SHFT,DRSHFT,0
2536 013012 031772 031477 031603
2536 013020 000000
2537
2538 013022 032345 032366 032406  DTTBL:  .WORD      ROW1,ROW2,ROW3,ROW4,SPCB
2538 013030 032424 032446
2539 013034 032450 000000 032442      .WORD      KYPD,0,CNTRA,SHFTA,SHFTA
2539 013042 032444 032444

```

;;*****
 ;SUBROUTINE TO ALLOW SETUP FROM MULTIPLE ENTRIES
 ;;*****

```

2540
2541
2542
2543
2544
2545 013046      SETA:
(1)      .SBTTL  INITIALIZE THE COMMON TAGS
(1)      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
(1) 013046 012706 001100      MOV      #SCMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
(1) 013052 005026      CLR      (R6)+      ;;CLEAR MEMORY LOCATION
(1) 013054 022706 001140      CMP      #SWR,R6 ;;DONE?
(1) 013060 001374      BNE      -6      ;;LOOP BACK IF NO
(1) 013062 012706 001100      MOV      #STACK,SP      ;;SETUP THE STACK POINTER
(1)      ;;INITIALIZE A FEW VECTORS
(1) 013066 012737 023214 000020      MOV      #SCOPE,@IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
(1) 013074 012737 000340 000022      MOV      #340,@IOTVEC+2 ;;LEVEL 7
(1) 013102 012737 023470 000030      MOV      #ERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
(1) 013110 012737 000340 000032      MOV      #340,@EMTVEC+2 ;;LEVEL 7
(1) 013116 012737 025074 000034      MOV      #STRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(1) 013124 012737 000340 000036      MOV      #340,@TRAPVEC+2 ;;LEVEL 7
(1) 013132 012737 024704 000024      MOV      #SPWRDN,@PWRVEC ;;POWER FAILURE VECTOR
(1) 013140 012737 000340 000026      MOV      #340,@PWRVEC+2 ;;LEVEL 7
(1) 013146 013737 011672 011664      MOV      $ENDCT,$EOPCT      ;;SETUP END-OF-PROGRAM COUNTER

```

```

(1) 013154 005037 001160 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
(1) 013160 005037 001162 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
(1) 013164 112737 000001 001115 MOVB #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
(2) ;;INITIALIZE THE 'T-BIT' TRAP VECTOR. THEN LOAD LOCATION '$RTRN', IN
(2) ;;THE 'END-OF-PASS' ($EOP) ROUTINE, WITH A 'RTI' OR 'RTT'.
(2) 013172 012737 012006 000014 MOV #$RTRN,@#TBITVEC ;;SET 'T' BIT VECTOR TO $RTRN
(2) 013200 012737 000340 000016 MOV #340,@#TBITVEC+2 ;;LEVEL 7
(2) 013206 012737 000002 012006 MOV #RTI,$RTRN ;;SET $RTRN TO A RTI
(2) 013214 012737 013242 000010 MOV #65$,@#RESVEC ;;TRY TO DO A RTT
(2) 013222 005046 CLR -(SP) ;;DUMMY PS
(2) 013224 012746 013232 MOV #64$,-(SP) ;;AND PC
(2) 013230 000006 RTT ;;TRY THE RTT
(2) 013232 012737 000006 012006 64$: MOV #RTT,$RTRN ;;RTT IS LEGAL--SET $RTRN TO A RTT
(2) 013240 000402 BR 66$
(2) 013242 062706 000010 65$: ADD #10,SP ;;RTT ILLEGAL--CLEAN OFF THE STACK
(2) 013246 012737 000012 000010 66$: MOV #RESVEC+2,@#RESVEC ;;RESTORE TRAP CATCHER
(2) 013254 005037 012014 CLR $TBIT ;;CLEAR 'T' BIT SWITCH
(1) 013260 012737 013260 001106 MOV #,$$LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
(1) 013266 012737 013266 001110 MOV #,$$LPERR ;;SETUP THE ERROR LOOP ADDRESS
(2) ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
(2) ;;EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
(2) 013274 013746 000004 MOV @#ERRVEC,-(SP) ;;SAVE ERROR VECTOR
(2) 013300 012737 013334 000004 MOV #67$,@#ERRVEC ;;SET UP ERROR VECTOR
(2) 013306 012737 177570 001140 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
(2) 013314 012737 177570 001142 MOV #DISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
(2) 013322 022777 177777 165610 CMP #-1,@SWR ;;TRY TO REFERENCE HARDWARE SWR
(2) 013330 001012 BNE 69$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
(2) ;;AND THE HARDWARE SWR IS NOT -1
(2) 013332 000403 BR 68$ ;;BRANCH IF NO TIMEOUT
(2) 013334 012716 013342 67$: MOV #68$,(SP) ;;SET UP FOR TRAP RETURN
(2) 013340 000002 RTI
(2) 013342 012737 000176 001140 68$: MOV #SWREG,SWR ;;POINT TO SOFTWARE SWR
(2) 013350 012737 000174 001142 MOV #DISPREG,DISPLAY
(2) 013356 012637 000004 69$: MOV (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
(1)
2546 .SBTTL TYPE PROGRAM NAME
(1) ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
(1) 013362 005227 177777 INC #-1 ;;FIRST TIME?
(1) 013366 001017 BNE 70$ ;;BRANCH IF NO
(1) 013370 022737 011742 000042 CMP #SENDAD,@#42 ;;ACT-11?
(1) 013376 001413 BEQ 70$ ;;BRANCH IF YES
(1) 013400 104401 013406 TYPE 71$ ;;TYPE ASCIZ STRING
(1) 013404 000410 BR 70$ ;;GET OVER THE ASCIZ
(1) ;;71$: .ASCIZ <CRLF>#MD-11-DZVTM-AA<CRLF>
(1) 70$:
2547 013426 104401 025214 TYPE $STUPM ;;ISSUE SET-UP MESSAGE.
2548 013432 012737 013452 000010 MOV #TRPA,@#10 ;;AND VECTOR
2549 013440 000230 SPL 0 ;;PROCESSOR IS 11/45?
2550 013442 012737 000004 021436 MOV #4,$PMULT ;;YES-DELAY MULTIPLIER - 4
2551 013450 000416 BR RTRP
2552
2553 013452 022626 TRPA: POP2SP ;;NO
2554 013454 012737 013476 000010 MOV #TRPB,@#10 ;;RELOAD TRAP ADDRESS
2555 013462 000737 002356 SXT CHR0 ;;PROCESSOR IS 11/40 OR 35?
2556 013466 012737 000002 021436 MOV #2,$PMULT ;;YES-DELAY MULTIPLIER=2
2557 013474 000404 BR RTRP

```

```

2558
2559 013476 022626 TRPB: POP2SP
2560 013500 012737 000001 021436 MOV #1,PMULT ;PROCESSOR MUST BE 11/05
2561 013506 012737 000012 000010 RTRP: MOV #12,@#10 ;RESTORE TRAP CATCHER
2562 013514 105737 002372 TSTB MODE ;CHECK MODE FOR CORRECT EXIT.
2563 013520 001402 BEQ 70$
2564 013522 000137 002516 JMP MANSA ;EXIT TO MANUAL SELECT
2565 013526 000137 002450 70$: JMP AUTOA ;EXIT TO AUTO MODE.
2566 ;:*****
2567 ;THIS SUBROUTINE WILL VERIFY EACH ADDRESS AS IT IS
2568 ;ENTERED VIA THE KEYBOARD WHEN IN MANUAL MODE.
2569 ;:*****
2570
2571 013532 020227 160000 TMNAD: CMP R2,#160000 ;INSURE ADDRESS IS IN RANGE.
2572 013536 103407 BLO BDEXT ;ITS NOT-TYPE A ? AND EXIT.
2573 013540 012737 013554 000004 MOV #BDEXTA,@#4 ;SET UP TRAP EXIT.
2574 013546 005712 TST @R2 ;CHECK THE ADDRESS.
2575 013550 000240 NOP
2576 013552 000405 BR ALEXT ;IF WE GOT THIS FAR ITS OK.
2577
2578 013554 022626 BDEXTA: POP2SP ;ADDRESS TRAPPED-PURGE IT, TYPE A
2579 013556 104401 031344 BDEXT: TYPE ,QMRK ;QUESTION MARK, RESTORE TRAP
2580 013562 012700 002536 MOV #BLDADD,R0 ;LOCATION, AND EXIT TO BUILD THE
2581 013566 012737 000006 000004 ALEXT: MOV #6,@#4 ;NEXT ENTRY.
2582 013574 000200 RTS R0
2583
2584 ;:*****
2585 ;THIS ROUTINE MAPS ALL POSSIBLE DZ11 ADDRESSES AND STORES
2586 ;THEM IN A TABLE (INTAB). ALL ADDRESSES WHICH DO NOT
2587 ;RESULT IN TIMEOUTS ARE STORED.
2588 ;:*****
2589
2590 013576 012701 000300 TRPVEC: MOV #300,R1 ;START AT BEG. OF FLOATING VECTORS
2591 013602 012702 000302 MOV #302,R2
2592 013606 012703 000004 MOV #4,R3
2593 013612 010221 1$: MOV R2,(R1)+ ;R3 CONTAINS IOT TRAP INST.
2594 013614 010321 MOV R3,(R1)+ ;START LOADING ADDRESSES
2595 013616 062702 000004 ADD #4,R2 ;LOAD THE TRAP
2596 013622 020127 001000 CMP R1,#1000 ;ASSUME 4 REGISTERS PER INTERFACE
2597 013626 002771 BLT 1$ ;DONE?
2598 013630 005037 000006 CLR @#6 ;NO CONTINUE LOADING TRAPS
2599 013634 012737 013674 000004 MOV #TPENT,@#4 ;SET UP TIME-OUT TRAP ADDRESS
2600 013642 005001 CLR R1 ;CLEAR THE TABLE POINTER
2601 013644 012705 001572 MOV #INTAB,R5 ;R5=DESTINATION TABLE
2602 013650 013702 002106 FADD: MOV STRTAB,R2 ;PUT THE ADDRESS TO BE TESTED IN R2
2603 013654 023702 002110 TRPE: CMP ENDTAB,R2 ;HAVE WE EXCEEDED END OF TABLE ADDRESS?
2604 013660 103407 BLO TBLCK ;YES GET NEXT BASE ADDRESS.
2605 013662 005712 TST @R2 ;ADDRESS THE DEVICE IF POSSIBLE
2606 013664 010225 MOV R2,(R5)+ ;IF WE GOT THIS FAR THERE IS A DEVICE THERE-SAVE IT
2607 013666 062702 000010 FADD1: ADD #10,R2 ;INCREMENT TO THE NEXT POSSIBLE ADDRESS
2608 013672 000770 BR TRPE ;GO TEST THE NEXT ADDRESS
2609 013674 022626 TPENT: POP2SP ;RESTORE THE STACK AND TEST
2610 013676 000773 BR FADD1 ;NEXT ADDRESS
2611 013700 005015 TBLCK: CLR (R5) ;SET UP TABLE TERMINATOR OF ZEROS.
2612 013702 000200 RTS R0
2613 ;:*****

```

```

2614                                     ;THIS ROUTINE WILL INSURE THAT THE DEVICE(DZ11)
2615                                     ;WILL INTERRUPT WHEN XMIT INT. ENABLE BIT IS SET.
2616                                     ;:*****
2617
2618 013704 005046 CDEV: CLR -(SP) ;CLEAR THE PSW,LSI11 STYLE.
2619 013706 012746 013714 MOV #100$,-(SP)
2620 013712 000002 RTI
2621 013714 012737 000004 000004 100$: MOV #4,@#4 ;INSTALL IOT TRAP INST. AT LOCATION 4.
2622 013722 012737 014052 000020 MOV #TDEV,@#IOTVEC ;SET UP IOT TRAP EXIT ADDRESS
2623 013730 012737 000340 000022 MOV #340,@#IOTVEC+2 ;SET PSW TO 7-ALLOW NO OTHER INTERRUPTS
2624 013736 000005 RESET ;INSURE ALL XMIT FLAGS HIGH.
2625 013740 012703 001552 MOV #VVECT,R3 ;VECTOR STORAGE ADDRESS SET
2626 013744 012702 001572 MOV #INTAB,R2 ;PRIMARY DEVICE TABLE ADDRESS SET
2627 013750 005004 CLR R4 ;DZ LINE TABLE/MODEM FLAG TABLE OFFSET
2628 013752 012705 001562 MOV #DZTBL,R5 ;DEVICE TABLE ADDRESS SET.
2629 013756 005037 002074 CDEVA: CLR MODEM ;CLEAR GENERAL MODEM FLAG
2630 013762 012701 002112 MOV #VZCSR,R1 ;VT51 DEVICE ADDRESS SET.
2631 013766 005712 TST (R2) ;CHECKED ALL DEVICES?
2632 013770 001002 BNE 11$ ;IF ZERO,...
2633 013772 000137 014324 JMP AOUT ;EXIT.
2634 013776 100404 1$: BMI 1$ ;INSURE ADDRESS IS IN PROPER RANGE(17XXXX)
2635
2636 014000 062702 000002 ADD #2,R2 ;ADDRESS IS DEFINITELY NOT GOOD -PURGE
2637 014004 000137 013756 JMP CDEVA ;AND LOOK FOR ANOTHER.
2638 014010 004037 015172 1$: JSR R0,LDADD ;LOAD NEXT ADDRESSES TO BE CHECKED
2639 014014 012701 001200 MOV #1200,R1 ;NOW USE R1 AS FAILSAFE COUNTER
2640 014020 052777 000377 166070 BIS #377,@VXTCR ;ENABLE ALL LINES TO BE SCANNED.
2641 014026 052777 000040 166056 BIS #XSCN,@VZCSR ;ENABLE THE XMITTER SCANNER.
2642 014034 052777 040000 166050 BIS #XENA,@VZCSR ;SET XMIT ENABLE
2643 014042 005301 DEC R1 ;IF DEVICE DOES NOT INTERRUPT WITHIN
2644 014044 001376 BNE -2 ;APPROX. 200US IT IS NOT A DZ11.
2645 014046 000137 013756 JMP CDEVA ;THEREFORE, GO TRY ANOTHER DEVICE.
2646 014052 042777 040000 166032 TDEV: BIC #XENA,@VZCSR ;CLEAR XMIT ENABLE.
2647 014060 042777 000040 166024 BIC #XSCN,@VZCSR ;DISABLE THE XMIT SCANS.
2648 014066 105077 166024 CLRB @VXTCR ;CLEAR XMIT LINE SCAN REG.
2649 014072 162716 000010 SUB #10,(R6) ;RESET TO RECEIVER VECTOR ADDRESS
2650 014076 012613 MOV (R6)+,(R3) ;STORE IT IN VECTOR TABLE(VVECT).
2651 014100 005726 TST (R6)+ ;POP THE OLD PSW AND DISCARD
2652 014102 022626 POP2SP ;POP THE ADD. AND PSW PRIOR TO INTERRUPT.
2653 ;:*****
2654 ;THIS ROUTINE IS A QUICK TEST OF ANY DZ11 ENCOUNTERED
2655 ;A DATA PATTERN WILL BE RUN ON ALL ENTRIES IN INTAB
2656 ;:*****
2657 014104 005046 CLR -(SP) ;CLEAR THE PSW,LSI11 STYLE.
2658 014106 012746 014114 MOV #100$,-(SP)
2659 014112 000002 RTI
2660 014114 012301 100$: MOV (R3)+,R1 ;GET THE RECEIVE VECTOR ADDRESS
2661 014116 012721 016064 MOV #RECAD,(R1)+ ;AND STORE SAME.
2662 014122 012721 000340 MOV #340,(R1)+ ;SET RECEIVE PSW TO 7.
2663 014126 012721 016160 MOV #TSMAD,(R1)+ ;STORE THE XMIT VECTOR ADDRESS
2664 014132 012711 000340 MOV #340,(R1) ;SET XMIT PSW TO 7.
2665 014136 052777 000010 165746 BIS #10,@VZCSR ;SET MAINT. BIT
2666 014144 052777 177400 165744 BIS #177400,@VXTCR ;SET ALL DTR BITS
2667 014152 032777 177400 165736 BIT #177400,@VXTCR ;CARRIER STAYS UP IF EIA CONTROL
2668 014160 001403 BEQ 101$ ;BR IF NOT
2669 014162 005237 002074 INC MODEM ;SET MODEM FLAG

```

```

2670 014166 000402          BR      102$
2671 014170 005337 002074    101$: DEC      MODEM          ;SET FOR NO MODEM
2672 014174 042777 177400    102$: BIC      #177400,@VXTCR    ;CLEAR ALL DTR BITS
2673 014202 042777 000010    BIC      #10,@VZCSR          ;CLEAR MAINT BIT
2674 014210 012737 000001 002124    MOV      #1,TSTLNE        ;SET UP TO TEST LINE 0 FIRST.
2675 014216 013764 002124 001632    LNECK: MOV      TSTLNE,DZLNE(R4) ;STORE THE GOOD LINE
2676 014224 013764 002074 001752    MOV      MODEM,MZLNE(R4)
2677 014232 062704 000002      ADD      #2,R4
2678 014236 106337 002124      ASLB      TSTLNE          ;SET UP TO CHECK NEXT LINE.
2679 014242 001365          BNE      LNECK          ;IF NOT ZERO-GO CHECK IT.
2680 014244 012764 177777 001632    MOV      #-1,DZLNE(R4) ;STORE DZ11 LINE SEPARATOR.
2681 014252 012764 177777 001752    MOV      #-1,MZLNE(R4)
2682 014260 062704 000002      ADD      #2,R4
2683 014264 013725 002112      MOV      VZCSR,(R5)+      ;STORE THE DZ11 ADDRESS.
2684 014270 042777 000040 165614    BIC      #SCAN,@VZCSR    ;DISABLE REC. AND XMIT SCANNERS.
2685 014276 052777 000020 165606    BIS      #MCLR,@VZCSR    ;CLEAR SILO AND UARTS
2686 014304 032777 000020 165600    $:      BIT      #MCLR,@VZCSR ;WAIT FOR OPERATION COMPLETION.
2687 014312 001374          BNE      1$
2688 014314 104401 001171      TYPE      $,CRLF
2689 014320 000137 013756      JMP      CDEVA          ;:CHECK ANOTHER DZ11
2690 014324 005015          AUJT:  CLR      (R5)          ;SET A ZERO TABLE TERMINATOR.
2691 014326 005064 001632      CLR      DZLNE(R4)      ;SET THE LINE TABLE TERMINATOR
2692 014332 005064 001752      CLR      MZLNE(R4)
2693 014336 012737 000006 000004    MOV      #6,@#4          ;RESTORE LOCATION 4 TO HALT CONDITION
2694 014344 005037 000006      CLR      @#6          ;TO CATCH ERRORS AND ILLEGAL INTERRUPTS.
2695 014350 012737 023214 000020    MOV      #SCOPE,@#IOTVEC ;RELOAD IOT VECTOR FOR SCOPE
2696 014356 012737 000340 000022    MOV      #340,@#IOTVEC+2 ;LOOP.
2697 014364 012701 000300      MOV      #300,R1
2698 014370 012702 000302      MOV      #302,R2
2699 014374 010221 1$:      MOV      R2,(R1)+
2700 014376 005021          CLR      (R1)+      ;RESTORE HALTS TO ALL LOCATIONS CONTAINING IOTS
2701 014400 062702 000004      ADD      #4,R2
2702 014404 020127 001000      CMP      R1,#1000      ;TO LOCATION 1000
2703 014410 103771          BLO      1$
2704 014412 000005          RESET          ;CLEAR ALL FLAGS
2705 014414 000200          RTS      RO
2706
2707          ;:*****
2708          ;INITIALIZATION ROUTINE FOR AUTO SELECTION. THIS ROUTINE
2709          ;WILL INSURE THAT ALL DZ11S IN DZTBL HAVE A VT61 CONNECTED
2710          ;ALL UNITS WHICH EITHER DO NOT OR DO NOT RESPOND WILL BE PURGED.
2711          ;:*****
2712 014416 012702 001562    INITA: MOV      #DZTBL,R2      ;R2 POINTS TO DZ11 ADDRESS TABLE
2713 014422 012704 001752      MOV      #MZLNE,R4      ;POINTER TO MODEM FLAGS
2714 014426 012703 001632      MOV      #DZLNE,R3      ;POINTER TO DZ11 LINES.
2715 014432 012701 002112 11$:      MOV      #VZCSR,R1      ;POINTER TO VT61 DZ11
2716 014436 052777 000020 165446    BIS      #MCLR,@VZCSR    ;CLEAR SILO AND UARTS.
2717 014444 032777 000020 165440 110$: BIT      #MCLR,@VZCSR ;WAIT FOR CLEAR OPERATION TO COMPLETE.
2718 014452 001374          BNE      110$
2719
2720 014454 005712          TST      (R2)          ;SEE IF ALL CHECKED
2721 014456 001556          BEQ      INTXT          ;YES-EXIT
2722 014460 004037 015172      JSR      RO,LDADD        ;NO-GO LOAD THE ADDRESSES
2723 014464 052777 000020 165420 12$: BIS      #MCLR,@VZCSR    ;MASTER CLEAR
2724 014472 022713 177777      CMP      #-1,(R3)      ;AT A LINE TABLE TERMINATOR?
2725 014476 001006          BNF      13$          ;NO-CONTINUE TESTING THIS ADDRESS.

```

2726	014500	005723			TST	(R3)+	;IT IS-BUMP THE POINTER AND GET NEXT
2727	014502	005724			TST	(R4)+	
2728	014504	052777	000020	165400	BIS	#MCLR,@VZCSR	;SHUT DOWN DZ11 AFTER SAMPLING COMPLETE
2729	014512	000747			BR	11\$	;DZ11 ADDRESS(IF ANY).
2730	014514	011337	002124		13\$: MOV	(R3),TSTLNE	;LOAD LINE TO BE TESTED.
2731	014520	004037	021730		JSR	R0,CONVLN	;CONVERT BINARY LINE TO OCTAL LINE.
2732	014524	012337	002072		MOV	(R3)+,SCRCH	;ENABLE LINE AND SET DTR
2733	014530	013705	002072		MOV	SCRCH,R5	
2734	014534	005724			TST	(R4)+	;MODEM PRESENT?
2735	014536	100420			BMI	21\$	;BR IF NOT
2736	014540	000305			SWAB	R5	;SET MASK FOR BOTH TRANSMIT ENABLE
2737	014542	063705	002072		ADD	SCRCH,R5	;AND DATA TERMINAL READY
2738	014546	050577	165344		BIS	R5,@VXTCR	;SET
2739	014552	042705	000377		BIC	#377,R5	
2740	014556	005037	002072		CLR	SCRCH	;SET UP FAILSAFE TIMER
2741	014562	030577	165332		22\$: BIT	R5,@VXBUF	;WAIT FOR CARRIER
2742	014566	001006			BNE	23\$	;BR IF DONE
2743	014570	005337	002072		DEC	SCRCH	
2744	014574	001470			BEQ	52\$	;DROP LINE IF FAILSAFE TIME EXCEEDED
2745	014576	000771			BR	22\$	
2746	014600	050577	165312		2\$: BIS	R5,@VXTCR	;SET TRANSMIT ENABLE ONLY
2747	014604	013737	002126	002072	23\$: MOV	OCTLNE,SCRCH	;GET THE LINE NUMBER
2748	014612	063737	002134	002072	ADD	BDRATE,SCRCH	;ADD IN BAUD RATE
2749	014620	062737	010070	002072	ADD	#RXE!SCL,SCRCH	;ADD IN LINE PARAMETERS
2750	014626	013777	002072	165260	MOV	SCRCH,@VRBUF	;LOAD LINE PARAMETER REGISTER
2751	014634	052777	000040	165250	BIS	#SCAN,@VZCSR	;ENABLE XMIT AND REC SCANNERS.
2752	014642	005037	002072		CLR	SCRCH	;SET UP FOR RETRY
2753	014646	004037	017672		49\$: JSR	R0,ZFLAG	;ISSUE ESCZ AND LOOK FOR RESPONSE.
2754	014652				2\$:		
(2)	014652	012637	002356		MOV	(SP)+,CHRD	;POP STACK INTO CHRD
2755	014656	100414			BMI	5\$	;TIMEOUT OCCURRED NO CHARACTER
2756	014660	123727	002356	000140	CMPB	CHRD,#140	;CHECK IDENT FOR VT61 IDENTIFIERS
2757	014666	103430			BLO	51\$	;NOT A VT61-SET UP TO PURGE ADDRESS
2758	014670	123727	002356	000172	CMPB	CHRD,#172	;IDENTS ARE SMALL A THRU Z
2759	014676	101024			BHI	51\$	;NOT A VT61-PURGE
2760	014700				4\$:		
(2)	014700	012637	002356		MOV	(SP)+,CHRD	;POP STACK INTO CHRD
2761	014704	001375			BNE	4\$	
2762	014706	000666			BR	12\$	;TEST ANOTHER ADDRESS
2763	014710				5\$:		
(2)	014710	012637	002356		MOV	(SP)+,CHRD	;POP STACK INTO CHRD
2764	014714	001375			BNE	5\$	
2765	014716	005237	002072		INC	SCRCH	;RETRY?
2766	014722	023727	002072	000005	CMP	SCRCH,#5	
2767	014730	001412			BEQ	52\$	;IF NO, BRANCH
2768	014732	013737	002136	002140	MOV	X,Y	;WAIT BEFORE RETRY
2769	014740	005337	002140		80\$: DEC	Y	
2770	014744	001375			BNE	80\$	
2771	014746	000737			BR	49\$	;RETRY
2772	014750				51\$:		
(2)	014750	012637	002356		MOV	(SP)+,CHRD	;POP STACK INTO CHRD
2773	014754	001375			BNE	51\$	
2774	014756	162703	000002		52\$: SUB	#2,R3	;RESET LINE POINTER.
2775	014762	162704	000002		SUB	#2,R4	
2776	014766	010346			MOV	R3,-(SP)	;PUSH R3 ON STACK
2777	014770	012746	000001		MOV	#1,-(SP)	;PUSH #1 ON STACK

```

2778 014774 004037 015412      JSR    R0,BBLUP
2779 015000 010446      MOV    R4,-(SP)
2780 015002 012746 000001      MOV    #1,-(SP)
2781 015006 004037 015412      JSR    R0,BBLUP
2782 015012 000624      BR     12$
2783 015014 022737 177777 001632 INTXT: CMP    #-1,DZLNE      ;TRY ANOTHER DZ11 LINE.
2784 015022 001021      BNE    EXINT      ;GOOD LINE FROM 1ST ADDRESS?
2785 015024 022737 177777 001634      CMP    #-1,DZLNE+2    ;YES-BEGIN TESTING.
2786 015032 001403      BEQ    NOUNIT     ;SEE IF SECOND DJ HAS GOOD LINES.
2787 015034 005737 001634      TST    DZLNE+2      ;NO GOOD LINES ON 2ND DZ11.
2788 015040 001012      BNE    EXINT      ;ZERO TERMINATOR FOUND?
2789 015042 104401 025401      NOUNIT: TYPE   EXINT     ;NO-FIRST DZ11 HAS GOOD LINES.
2790 015046 012737 002734 021440      MOV    #1500.,DCOUNT ;NO-ISSUE NO VT61 MESSAGE.
2791 015054 004037 021376      JSR    R0,DELAY      ;SET DELAY TO 30 SEC.
2792 015060 062700 000004      ADD    #4,R0         ;AND DO IT.
2793 015064 000200      RTS    R0          ;SET UP 'NO VT61 FOUND' EXIT
2794 015066 012702 001562      EXINT: MOV    #DZTBL,R2    ;LOAD AND ISSUE GOOD ADDRESSES
2795 015072 005712      TST    (R2)          ;INSURE A GOOD ADDRESS.
2796 015074 001762      BEQ    NOUNIT     ;NONE FOUND-EXIT
2797 015076 012703 001632      MOV    #DZLNE,R3    ;LOAD TABLE ADDRESS OF GOOD LINES.
2798 015102 012704 001752      MOV    #MZLNE,R4
2799 015106 104401 025330      TYPE    ,DUNTST     ;OF RESPONSIVE VT61S.
2800 015112      1$:
(1) 015112 012246      MOV    (R2)+,-(SP)    ;;SAVE (R2)+ FOR TYPEOUT
(1)      ;;TYPE AN ADDRESS
(1) 015114 104403      TYP0S      ;;GO TYPE--OCTAL ASCII
(1) 015116 006      .BYTE    6      ;;TYPE 6 DIGIT(S)
(1) 015117 001      .BYTE    1      ;;TYPE LEADING ZEROS
2801 015120 104401 001171      TYPE    ,SCLRF
2802 015124 012337 002124      2$: MOV    (R3)+,TSTLNE    ;LOAD A LINE TO PRINT
2803 015130 005724      TST    (R4)+
2804 015132 022737 177777 002124      CMP    #-1,TSTLNE    ;IF LINE IS A TERMINATOR, DO NOT
2805 015140 001407      BEQ    3$          ;DISPLAY - GO SET UP NEXT DZ11.
2806 015142 004037 021730      JSR    R0,CONVLN     ;CONVERT IT TO A OCTAL #.
2807 015146 013746 002126      MOV    OCTLNE,-(SP)  ;;SAVE OCTLNE FOR TYPEOUT
(1)      ;;TYPE A GOOD LINE
(1) 015152 104403      TYP0S      ;;GO TYPE--OCTAL ASCII
(1) 015154 003      .BYTE    3      ;;TYPE 3 DIGIT(S)
(1) 015155 000      .BYTE    0      ;;SUPPRESS LEADING ZEROS
2808 015156 000762      BR     2$          ;NO-TYPE ANOTHER LINE.
2809 015160 104401 001171      3$: TYPE    ,SCLRF
2810 015164 005712      TST    (R2)          ;AT END OF GOOD UNITS?
2811 015166 001351      BNE    1$          ;NO PRINT ANOTHER ADDRESS.
2812 015170 000200      RTS    R0
2813
2814      ;;*****
2815      ;SUBROUTINE TO LOAD 4 ADDRESSES FROM THE LOCATION AT (R2).
2816      ;TO A LOCATION POINTED TO BY R1.EXIT WITH R2 INC. BY 2.
2817      ;;*****
2818
2819
2820
2821 015172 012211      LDADD: MOV    (R2)+,(R1)    ;LOAD THE ADDRESS
2822 015174 012111      1$: MOV    (R1)+,(R1)    ;STORE AN ADDRESS
2823 015176 062711 000002      ADD    #2,(R1)      ;INCREMENT THE ADDRESS
2824 015202 020127 002120      CMP    R1,#VXBUF    ;LOADED 4?

```

```

2825 015206 002772      BLT      1$      ;NO LOAD ANOTHER
2826 015210 000200      RTS      RO      ;YES-EXIT
2827
2828 ;*****
2829 ;ROUTINE TO RECEIVE CHARACTER(S). ENTERED WITH
2830 ;NUMBER OF CHARACTERS TO RECEIVE ON THE STACK.
2831 ;ROUTINE EXITS WITH CHARACTER(S) ON STACK. IF A
2832 ;PROGRAM TIME-OUT (100 M.S.) OCCURS BEFORE A CHARACTER
2833 ;IS RECEIVED ROUTINE EXITS WITH -1 ON STACK. FORMAT
2834 ;FOR DATA IS (BYTE2, BYTE1) ETC. A WORD OF ZEROS TERMINATES
2835 ;DATA STRING ON THE STACK. SOM/EOM, IF SENT, ARE RECEIVED
2836 ;BUT NOT STORED.
2837
2838 ;*****
2839
2840 015212      RECTM:
2841 (2) 015212 012637 002416      MOV      (SP)+,ROSVE      ;;POP STACK INTO ROSVE
2842 015216 012637 002404      MOV      (SP)+,@BUBCT      ;;POP STACK INTO @BUBCT
2843 015222 013746 002364      MOV      ZERO,-(SP)      ;;PUSH ZERO ON STACK
2844 015226 005037 002356      1$:      CLR      CHRD      ;CLEAR CHARACTER STORAGE LOCATION.
2845 015232 005037 002414      CLR      DLAY      ;SET UP FAILSAFE DELAY
2846 015236 032777 000200 164646 3$:      BIT      #RECDN,@VZCSR      ;SEE IF DONE FLAG SET
2847 015244 001007      BNE      4$
2848 015246 005337 002414      DEC      DLAY      ;DECREMENT FAILSAFE CNTR.
2849 015252 001371      BNE      3$      ;NOT AT ZERO-CONTINUE WAITING.
2850 015254 012737 177777 002356 31$:      MOV      #-1,CHRD      ;SET UP FOR FAILSAFE EXIT.
2851 015262 000446      BR       RECEX      ;EXIT ROUTINE.
2852 015264 017737 164624 002356 4$:      MOV      @VRBUF,CHRD      ;STORE THIS CHARACTER.
2853 015272 042737 170200 002356      BIC      #170200,CHRD      ;STRIP ALL BUT CHAR. AND LINE #.
2854 015300 123737 002126 002357      (MPB      OCTLINE,CHRD+1      ;RECEIVED FROM CORRECT LINE?
2855 015306 001347      BNE      1$      ;NO-IGNORE THIS CHAR.
2856 015310 122737 000057 002356      (MPB      #SLSH,CHRD      ;RECEIVED A IDENT SLASH(57)?
2857 015316 001007      BNE      41$      ;NO-STORE A CHARACTER.
2858 015320 105337 002405      DECB      BUBCT+1      ;DECREMENT ALLOWABLE SLASH COUNT.
2859 015324 001753      BEQ      31$      ;COUNT EQUAL ZERO-SET UP ERROR EXIT.
2860 015326 123727 002405 000213      (MPB      BUBCT+1,#139.      ;RECEIVED FIRST SLASH?
2861 015334 103734      BLO      1$      ;YES-IGNORE THIS ONE.
2862 015336 122737 000002 002356 41$:      (MPB      #SOM,CHRD      ;IS CHAR. ACTUALLY SOM?
2863 015344 001003      BNE      5$      ;NO
2864 015346 105237 002404      INCB      BUBCT      ;YES -SET UP TO RECEIVE EOM ALSO
2865 015352 000725      BR       1$      ;AND RECEIVE NEXT CHAR.
2866 015354 122737 000004 002356 5$:      (MPB      #EOM,CHRD      ;CHAR. - EOM?
2867 015362 001410      BEQ      RECEXA      ;YES- DO NOT PUSH IT ON STACK
2868 015364 105337 002404      DECB      BUBCT      ;DECREMENT CHARACTER COUNT.
2869 015370 001403      BEQ      RECEX      ;COUNT-0. EXIT WERE DONE.
2870 015372 013746 002356      MOV      CHRD,-(SP)      ;PUSH CHRD ON STACK
2871 015376 000713      BR       1$      ;GO READ AGAIN.
2872
2873 (2) 015400 013746 002356      RECEX:      MOV      (CHRD,-(SP)      ;;PUSH CHRD ON STACK
2874 (2) 015404 013746 002416      RECEXA:      MOV      ROSVE,-(SP)      ;;PUSH ROSVE ON STACK
2875 015410 000200      RTS      RO
2876
2877 ;*****

```

2878
2879
2880
2881
2882

:THIS ROUTINE WILL 'BUBBLE UP' XX WORDS TO
:ELIMINATE NON-RESPONSIVE ADDRESSES. ENTERED
:WITH ADDRESS TO BE 'BUBBLED' TO ON THE STACK. LOCATIONS
:ELIMINATED WILL BE FILLED WITH ZEROS. THE STACK MUST ALSO
:BE LOADED WITH THE NUMBER OF POSITIONS TO BUBBLE.

```

2885
2886
2887 015412
(2) 015412 012637 002416
2888 015416 012637 002404
2889 015422 012637 002402
2890 015426 010446
2891 015430 013704 002402
2892 015434 012437 002356
2893 015440 012464 177774
2894 015444 001375
2895 015446 005337 002404
2896 015452 001366
2897 015454
(2) 015454 012604
2898 015456 013746 002416
2899 015462 000200
2900
2901
2902
2903
2904
2905
2906
2907 015464
(2) 015464 012637 002416
2908 015470 010437 002404
2909 015474 112777 000000 164416
2910 015502 032777 100000 164402
2911 015510 001774
2912 015512 112777 000002 164400
2913 015520 012705 177777
2914 015524 012604
2915 015526 001415
2916 015530 110405
2917 015532 105704
2918 015534 001406
2919 015536 032777 100000 164346
2920 015544 001774
2921 015546 110477 164346
2922 015552 000304
2923 015554 120405
2924 015556 001760
2925 015560 000764
2926 015562 032777 100000 164322
2927 015570 001774
2928 015572 012777 000004 164320

```

```

;;*****
BBLUP:
      MOV      (SP)+,ROSVE      ;;POP STACK INTO ROSVE
      MOV      (SP)+,BUBCT      ;;POP STACK INTO BUBCT
      MOV      (SP)+,TOADD      ;;POP STACK INTO TOADD
      MOV      R4,-(SP)         ;;PUSH R4 ON STACK
2$:   MOV      @TOADD,R4        ;;PUT LAST GOOD DZ11 ADDRESS IN R4
      MOV      (R4)+,CHRD       ;;MOVE NEXT WORD TO CHRD FOR STORAGE
1$:   MOV      (R4)+,-4(R4)     ;;BUBBLE UP DATA.
      BNE      1$              ;;BUBBLE UNTIL ZERO BYTE MOVED.
      DEC      BUBCT           ;;SUBTRACT ONE FROM BUBBLE COUNT.
      BNE      2$              ;;IF BUBBLE COUNT NOT ZERO - DO AGAIN.
3$:   MOV      (SP)+,R4         ;;POP STACK INTO R4
      MOV      ROSVE,-(SP)      ;;PUSH ROSVE ON STACK
      RTS      R0              ;;YES-EXIT
;;*****
      ;;THIS ROUTINE OUTPUTS THE ESC SEQUENCE FOUND ON
      ;;THE STACK. A WORD OF ZEROS MUST TERMINATE THE SEQUENCE.
      ;;FORMAT FOR STACK WORD IS SEQ-ESC, IE-XXX033.
;;*****
TESC:
      MOV      (SP)+,ROSVE      ;;POP STACK INTO ROSVE
      MOV      R4,BUBCT        ;;SAVE R4.
      MOVB     #0,@VXBUF        ;;SEND NULL
11$:  BIT      #TRDY,@VZCSR     ;;WAIT
      BEQ      11$
      MOVB     #SOM,@VXBUF      ;;SEND A START OF MESSAGE.
1$:   MOV      #-1,R5           ;;ALL ONES TO THE CHECK LOCATION.
      MOV      (R6)+,R4        ;;GET COMMAND FROM STACK.
      BEQ      3$              ;;IF ZERO TERMINATOR FOUND-EXIT.
      MOVB     R4,R5           ;;LOAD CHECK BYTE.
2$:   TSTB     R4              ;;CHECK BYTE FOR A ZERO.
      BEQ      20$            ;;IF ZERO_DO NOT XMIT IT.
      BIT      #TRDY,@VZCSR
      BEQ      -6             ;;WAIT FOR XMIT READY BIT
      MOVB     R4,@VXBUF       ;;XMIT A BYTE.
20$:  SWAB     R4              ;;GET THE OTHER BYTE
      CMPB     R4,R5           ;;IF GOOD COMPARE WE HAVE CHECKED BOTH
      BEQ      1$              ;;BYTES SO POP ANOTHER WORD.
      BR       2$              ;;GO XMIT ANOTHER BYTE
3$:   BIT      #TRDY,@VZCSR     ;;SEE IF READY SET
      BEQ      -6
      MOV      #EOM,@VXBUF     ;;SEND A EOM.

```

```

2931 015600 032777 100000 164304      BIT    #TRDY,@VZCSR      ;SEE IF READY SET
2932 015606 001774          BEQ    .-6                      ;
2933 015610 112777 000000 164302      MOV    #0,@VXBUF      ;SEND A NULL
2934 015616 032777 100000 164266 31$:  BIT    #TRDY,@VZCSR      ;WAIT
2935 015624 001774          BEQ    31$                      ;
2936 015626 013704 002404          MOV    BUJCT,R4          ;RESTORE R4.
2937 015632 013746 002416          MOV    ROSVE,-(SP)        ;PUSH ROSVE ON STACK
2938 015636 000200          RTS    R0
2939
2940      ;;*****
2941      ;ROUTINE TO READ A CHARACTER FROM THE CONSOLE.
2942      ;EXITS WITH CHARACTER ON THE STACK.
2943      ;;*****
2944
2945 015640      CONRD:
2946 (2) 015640 012637 002416          MOV    (SP)+,ROSVE      ;;POP STACK INTO ROSVE
2947 015644 032777 000200 163272      BIT    #RECDN,@$TKS    ;LOOK FOR DONE BIT
2948 015652 001774          BEQ    .-6                      ;WAIT FOR IT
2949 015654 117746 163266          MOV    @$TKB,-(R6)        ;PUSH CHARACTER TO STACK
2950 015660 042716 000200          BIC    #200,(R6)          ;STRIP ANY PARITY BIT.
2951 015664 013746 002416          MOV    ROSVE,-(SP)        ;PUSH ROSVE ON STACK
2952 015670 000200          RTS    R0
2953
2954      ;;*****
2955      ;MANUAL TEST SELECT MONITOR
2956      ;SELECTS TESTS TO BE EXECUTED FROM THOSE ENTERED IN
2957      ;INITIAL DIALOGUE. IF TEST 377 WAS REQUESTED THE TESTS WILL
2958      ;REPEAT INFINITELY.
2959      ;;*****
2960
2960 015672 105737 002372      MONIT: TSTB    MODE          ;TEST MODE SWITCH
2961 015676 001012          BNE     1$                      ;MANUAL MODE
2962 015700 023737 002374 002376      CMP     FTLCNT,ALWCNT    ;COMPARE FATAL XMTS WITH ALLOWED.
2963 015706 103405          BLO     100$                    ;FATALS LESS THAN ALLOWED-CONTINUE.
2964 015710 104401 027315          TYPE    ,DABRT          ;ISSUE ABORT MESSAGE.
2965 015714 000005 200$:  RESET          ;CLEAR ALL INTERFACE FLAGS.
2966 015716 000137 013046          JMP     SETA          ;SET UP TO RESTART TEST.
2967 015722 000200 100$:  RTS     R0          ;AUTO MODE
2968 015724 005726 1$:  TST     (R6)+          ;POP THE STACK
2969 015726 022626          POP2SP          ;POP SCOPE RETURN AND VECTOR
2970 015730 005037 002374          CLR     FTLCNT          ;DO NOT INC. FATAL COUNT IN MANUAL MODE.
2971 015734 032777 000200 163202 10$:  BIT    #RECDN,@$TKS    ;CONSOLE ACTIVE?
2972 015742 001407          BEQ     11$                      ;
2973 015744 117701 163176          MOV    @$TKB,R1          ;STORE INPUT BUFFER
2974 015750 042701 000200          BIC    #200,R1          ;CLEAR THE PARITY BIT
2975 015754 122701 000003          CMPB   #3,R1          ;CHAR. EQUAL ESC. C?
2976 015760 001755          BEQ     200$                    ;YES-EXIT
2977 015762 117701 164402 11$:  MOV    @TSTPTR,R1          ;GET THE NEXT TEST #
2978 015766 010137 001102          MOV    R1,$TSTNM
2979 015772 001005          BNE     2$                      ;NOT AT END OF LIST
2980 015774 042777 000100 164110 12$:  BIC    #RENA,@VZCSR    ;CLEAR REC. INTERRUPTS BEFORE NEXT UNIT SELECT.
2981 016002 000137 003234          JMP     MODCA          ;END OF LIST-GO SET UP NEXT 61
2982 016006 100004 2$:  BPL     3$                      ;WAS TEST REPEAT REQUESTED?
2983 016010 012737 001572 002370          MOV    #INTAB,TSTPTR    ;YES-RESET TEST POINTER
2984 016016 000746          BR     10$                    ;AND GET FIRST TEST SELECTED
2985 016020 005301 3$:  DEC     R1          ;ADJUST OFFSET

```

```

2986 016022 006301          ASL      R1          ;USE TEST # TO FORM ADDRESS OFFSET
2987 016024 016137 025144 016062  MOV    TSTADD(R1),JMPADD+2 ;LOAD NEW ADDRESS
2988 016032 062737 000002 016062  ADD     #2,JMPADD+2 ;BYPASS INITIAL SCOPE LOOP
2989 016040 005237 002370          INC     TSTPTR    ;INCREMENT TEST OPINTER
2990 016044 005037 177776          CLR     PSW      ;SET NON-INT. PRIORITY TO ZERO
2991 016050 005046          CLR     -(SP)    ;CLEAR THE PSW,LSI11 STYLE.
2992 016052 012746 016060          MOV     #JMPADD,-(SP)
2993 016056 000002          RTI
2994 016060 000137 016060  JMPADD: JMP    JMPADD    ;EXIT TO NEXT SELECTED TEST
2995  (2)
2996  (2)
2997  (2)
2998  (2)
2999 016064 117737 164024 002356  RECAD: MOVB   @VRBUF,CHRD ;GET THE RECEIVED CHAR.
3000 016072 042737 000200 002356    BIC     #200,CHRD ;CLEAR ANY PARITY.
3001 016100 123737 002406 002356    CMPB   TPREG,CHRD ;COMPARE RECEIVED TO XMITTED
3002 016106 001407          BEQ     UPD4      ;BR IF OKAY
3003 016110 042777 040010 163774  TOFF:  BIC     #TCOMB,@VZCSR ;DATA ERROR OCCURED OR WE ARE DONE
3004 016116 042777 000100 163766    BIC     #RENA,@VZCSR ;EITHER WAY-EXIT.
3005 016124 000002          REEX:  RTI
3006 016126 106337 002406          UPD4:  ASLB   TPREG    ;UPDATE DATA PATTERN
3007 016132 032737 000100 002406    BIT     #BIT06,TPREG ;DONE?
3008 016140 001004          BNE     1$      ;BR IF YES
3009 016142 052777 040000 163742    BIS     #XENA,@VZCSR ;ENABLE TRANSMIT INTERRUPTS
3010 016150 000765          BR      REEX
3011 016152 005037 002406          1$:   CLR     TPREG    ;SET DONE
3012 016156 000754          BR      TOFF    ;AND EXIT.
3013 016160 113777 002406 163732  TSMAD: MOVB   TPREG,@VXBUF ;XMIT DATA
3014 016166 042777 040000 163716    BIC     #XENA,@VZCSR ;CLEAR XMIT INT. UNTIL LAST BIT REC.
3015 016174 000002          RTI
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031 016176          INTRC:
3032  (2) 016176 010146          MOV     R1,-(SP) ;:PUSH R1 ON STACK
3033  (2) 016200 010246          MOV     R2,-(SP) ;:PUSH R2 ON STACK
3034 016202 017701 163706          MOV     @VRBUF,R1 ;USE R1 FOR STORAGE OF STATUS AND CH.
3035 016206 010102          MOV     R1,R2 ;SET UP LINE CHECK LOCATION.
3036 016210 042702 170377          BIC     #170377,R2 ;CLEAR ALL BUT LINE BITS.
3037 016214 013737 002126 002072  MOV     OCTLINE,SCRCH ;GET A COPY OF EXPECTED LINE NUMBER.
3038 016222 000337 002072          SWAB   SCRCH ;PUT IT INTO THE RIGHT POSITION.
3039 016226 023702 002072          CMP     SCRCH,R2 ;COMPARE CHAR LINE TO LINE UNDER TEST.

```

3038	016232	001404			BEQ	13\$	:YES-EXAMINE IT.
3039	016234	052737	000010	002420	BIS	#ILLNE,VSTAT	:NO-SET ILLEGAL LINE FLAG AND EXIT.
3040	016242	000567			BR	ERLNE	
3041	016244	042701	000200		BIC	#200,R1	:STRIP PARITY BIT.
3042	016250	032737	000100	002420	BIT	#TXSUM,VSTAT	:CHECKSUM CALCULATION REQUESTED?
3043	016256	001403			BEQ	11\$	:NO
3044	016260	010105			MOV	R1,R5	:YES-STORE CHAR. AND
3045	016262	004037	022120		JSR	R0,CALCK	:CALCULATE THE CHECKSUM.
3046	016266	005237	033674		INC	ABUFP	:INCREMENT THE RAW DATA POINTER
3047	016272	023727	033674	033760	CMP	ABUFP,#ABBUF+50.	:AT THE END OF BUFFER?
3048	016300	001003			BNE	12\$	:NO
3049	016302	012737	033676	033674	MOV	#ABBUF,ABUFP	:YES-RESET IT
3050	016310	110177	015360		MOVB	R1,@ABUFP	:STORE THE RAW DATA
3051	016314	001505			BEQ	6\$	:IF CHAR. IS NULL-GO STORE IT
3052	016316	032737	000013	017132	BIT	#BIT00+BIT01+BIT03,ESAMB	:ESC OR ESC O?
3053	016324	001152			BNE	ASESC	:YES-KEEP ASSEMBLING
3054	016326	120137	002326		CMPB	R1,ESCN	:BYTE = ESCN?
3055	016332	101076			BHI	6\$	:NO-PROBABLY A DISPLAY CH.-STORE IT.
3056	016334	001007			BNE	1\$	:NO-DECODE FOR XON,XOFF,SOM,EOM
3057	016336	012737	000001	017132	MOV	#1,ESAMB	:YES SET ESC ASSEMBLY FLAG.
3058	016344	052737	000400	002420	BIS	#ESC,VSTAT	:SET ESC RECEIVED FLAG
3059	016352	000515			BR	RSTER	:AND EXIT
3060	016354	120127	000023		CMPB	R1,#XOFF	:SEE IF RECEIVED BYTE WAS XOFF
3061	016360	001004			BNE	2\$	:NO
3062	016362	052737	100000	002420	BIS	#RXOFF,VSTAT	:YES, SET XOFF IN STATUS REG.
3063	016370	000506			BR	RSTER	:EXIT
3064	016372	120127	000021		CMPB	R1,#XON	:SEE IF BYTE WAS XON
3065	016376	001016			BNE	3\$	:NO
3066	016400	042737	100000	002420	BIC	#RXOFF,VSTAT	:YES, CLEAR XOFF IN VSTAT.
3067	016406	032737	000200	002420	BIT	#XMKIL,VSTAT	:CHECK XMIT KILL BIT.
3068	016414	001474			BEQ	RSTER	:NOT SET, EXIT
3069	016416	052777	040000	163466	BIS	#XENA,@VZCSR	:SET XMIT INT. ENABLE.
3070	016424	042737	000200	002420	BIC	#XMKIL,VSTAT	:CLEAR THE XMIT KILLED FLAG
3071	016432	000465			BR	RSTER	:EXIT
3072	016434	120127	000002		CMPB	R1,#SOM	:SEE IF BYTE WAS SOM
3073	016440	001004			BNE	4\$	:NO
3074	016442	052737	040000	002420	BIS	#RSOM,VSTAT	:YES, SET SOM IN VSTAT.
3075	016450	000456			BR	RSTER	:EXIT
3076							
3077	016452	120127	000004		CMPB	R1,#EOM	:WAS BYTE EOM?
3078	016456	001012			BNE	5\$	:NO
3079	016460	052737	020000	002420	BIS	#REOM,VSTAT	:NOW SET EOM IN VSTAT.
3080	016466	013737	017124	017130	MOV	RBBUF,RBUFP	:RESET THE BUFFER POINTER.
3081	016474	042737	000100	002420	BIC	#TXSUM,VSTAT	:CLEAR CHECKSUM REQUEST BIT.
3082	016502	000441			BR	RSTER	:AND EXIT
3083	016504	123701	002146		CMPB	CARRT,R1	:CHAR. =CARRIAGE RETURN?
3084	016510	001403			BEQ	51\$	:YES-GO SET END OF LINE FLAG
3085	016512	123701	002150		CMPB	LNFEED,R1	:CHAR.= LINEFEED?
3086	016516	001004			BNE	6\$	:NO- GO STORE IT
3087	016520	052737	001000	002420	BIS	#EPL,VSTAT	:SET END OF LINE INDICATOR
3088	016526	000427			BR	RSTER	
3089							
3090	016530	023737	017130	017126	CMP	RBUFP,REBUF	:IS CIRCULAR BUFFER FILLED?
3091	016536	001003			BNE	61\$	:NO
3092	016540	013737	017124	017130	MOV	R2BUF,RBUFP	:YES, RESET POINTER TO BEGINNING
3093	016546	032737	000020	002420	BIT	#COMGP,VSTAT	:RECEIVING GRAPHICS CHAR.?

3094	016554	001402				BEQ	7\$	:NO
3095	016556	162701	000137			SUB	#137,R1	:YES-SUBTRACT 137 FROM RECEIVED CHAR.
3096								
3097	016562	032737	000040	002420	7\$:	BIT	#REVID,VSTAT	:REVERSE VIDEO MODE?
3098	016570	001402				BEQ	70\$	:NO STORE RECEIVED BYTE.
3099	016572	052701	000200			BIS	#200,R1	:YES-FORCE BIT7 AS REV. VIDEO IND.
3100	016576	110177	000326		70\$:	MOVB	R1,@RBUF	:STORE BYTE AND
3101	016602	005237	017130			INC	RBUF	:INCREMENT POINTER.
3102								
3103	016606	032701	070000		RSTER:	BIT	#70000,R1	:CHECK FOR STATUS ERROR
3104	016612	001414				BEQ	RECT	:NO, EXIT ROUTINE
3105								
3106	016614	052737	004000	002420		BIS	#PSTT,VSTAT	:SET STATUS ERROR FLAG IN VSTAT
3107	016622	027727	000330	177777	ERLNE:	CMP	@STTTP,#-1	:IS ERROR TABLE FULL?
3108	016630	001405				BEQ	RECT	:YES, EXIT ROUTINE
3109	016632	010177	000320			MOV	R1,@STTTP	:NO, STORE STATUS ERR. AND CHECK
3110	016636	062737	000002	017156		ADD	#2,STTTP	:INCREMENT STATUS ERR. POINTER
3111								
3112	016644				RECT:			
(2)	016644	012602				MOV	(SP)+,R2	:POP STACK INTO R2
(2)	016646	012601				MOV	(SP)+,R1	:POP STACK INTO R1
3113	016650	000002				RTI		:EXIT
3114	016652	032737	000002	017132	ADESC:	BIT	#2,ESAMB	:ASSEMBLING ESC P?
3115	016660	001063				BNE	AESCP	:YES-GO GET LAST CH.
3116	016662	032737	000010	017132		BIT	#BIT03,ESAMB	:ASSEMBLING ESC O?
3117	016670	001062				BNE	AESCO	:YES
3118	016672	122701	000120			CMPB	#120,R1	:CH. = A P?
3119	016676	001004				BNE	10\$	:NO KEEP CHECKING
3120	016700	052737	000002	017132		BIS	#BIT01,ESAMB	:YES-SET ESCP ASSEMBLY FLAG
3121	016706	000737				BR	RSTER	:AND EXIT
3122	016710	122701	000077		10\$:	CMPB	#77,R1	:CHAR. IS AN ESC ? ?
3123	016714	001403				BEQ	110\$	:YES-FAKE AN ESC O.
3124	016716	122701	000117			CMPB	#117,R1	:CHAR = O?
3125	016722	001004				BNE	11\$	:NO
3126	016724	052737	000010	017132	110\$:	BIS	#BIT03,ESAMB	:YES SET ESC O ASSEMBLY FLAG
3127	016732	000725				BR	RSTER	:AND EXIT
3128	016734	123701	002244		11\$:	CMPB	RDCUR,R1	:BYTE= CURSOR POSITION?
3129	016740	001004				BNE	1\$	:NO-
3130	016742	052737	000004	002420		BIS	#CURPOS,VSTAT	:YES-SET RECEIVED CURSOR POSITION.
3131	016750	000424				BR	CESAM	
3132	016752	122701	000057		1\$:	CMPB	#SLSH,R1	:BYTE=TERMINAL ID ESC?
3133	016756	001004				BNE	2\$	:NO-CHECK FOR GRAPHICS SEQUENCE.
3134	016760	052737	000002	002420		BIS	#TRMID,VSTAT	:YES-SET TERM. IDENT FLAG IN VSTAT
3135	016766	000415				BR	CESAM	
3136	016770	122701	000106		2\$:	CMPB	#CKGP,R1	:RECEIVED GRAPHICS CHAR. SEQUENCE?
3137	016774	001004				BNE	3\$	:NO
3138	016776	052737	000020	002420		BIS	#COMGP,VSTAT	:YES-SET GRAPHICS DATA FLAG.
3139	017004	000406				BR	CESAM	
3140	017006	122701	000107		3\$:	CMPB	#NCKGP,R1	:RECEIVED RESET GRAPHICS SEQ.?
3141	017012	001003				BNE	CESAM	:NO
3142	017014	042737	000020	002420		BIC	#COMGP,VSTAT	:YES-SET NORMAL CHAR. RECEIVE.
3143	017022	005037	017132		CESAM:	CLR	ESAMB	:CLEAR ASSEMBLY FLAG.
3144	017026	000667				BR	RSTER	:AND EXIT.
3145								
3146	017030	110137	017162		AESCP:	MOVB	R',STRP	:STORE ANY UNCHECKED FOR ESC. P
3147	017034	000772				BR	CESAM	

```

3148
3149 017036 123701 002212 AESCO: CMPB EEMP,R1 ;BYTE=ESC 0 -REV. VIDEO- ?
3150 017042 001004 BNE 1$ ;NO
3151 017044 052737 000040 002420 BIS #REVID,VSTAT ;YES-SET REVERSE VIDEO MODE IN VSTAT.
3152 017052 000763 BR CESAM
3153
3154 017054 123701 002214 1$: CMPB DEMP,R1 ;BYTE=ESC 0 DISABLE REV. VIDEO MODE?
3155 017060 001004 BNE 2$ ;NO
3156 017062 042737 000040 002420 BIC #REVID,VSTAT ;YES-CLEAR REVERSE VIDEO MODE IN VSTAT.
3157 017070 000754 BR CESAM
3158 017072 122701 000171 2$: CMPB #CPABRT,R1 ;COPIER ABORT?
3159 017076 001403 BEQ 3$ ;YES-SET ABORT FLAG IN VSTAT
3160 017100 122701 000172 CMPB #PRABRT,R1 ;PRINTER ABORT?
3161 017104 001004 BNE 4$ ;NO
3162 017106 052737 010000 002420 3$: BIS #PABRT,VSTAT ;YES-SET THE ABORT FLAG.
3163 017114 000742 BR CESAM ;AND EXIT.
3164 017116 110137 017160 4$: MOVB R1,STRO ;STORE ESCAPE 0 COMMAND
3165 017122 000737 BR CESAM
3166
3167 017124 000000 RBUF: .WORD ;ADDRESS OF STAT OF BUFFER
3168 017126 000000 REUF: .WORD ;ADDRESS OF END OF BUFFER.
3169 017130 000000 RBUF: .WORD ;READ BUFFER POINTER.
3170 017132 000000 ESAMB: .WORD 0 ;ESCAPE SEQ.ASSEMBLY AREA
3171
3172 017134 000010 STTER:
3174 017134 000000 0
(1) 017136 000000 0
(1) 017140 000000 0
(1) 017142 000000 0
(1) 017144 000000 0
(1) 017146 000000 0
(1) 017150 000000 0
(1) 017152 000000 0
3175 017154 177777 .WORD -1 ;STATUS REGISTER DELIMITER.
3176 017156 000000 STEP: .WORD ;STATUS ERROR POINTER.
3177 017160 000000 STRO: .WORD 0 ;ESCAPE 0 STORAGE
3178 017162 000000 STRP: .WORD ;ESCAPE P STORAGE
3179
3180 ;*****
3181 ;TRANSMIT INTERRUPT ROUTINE. IF XOFF BIT IS SET
3182 ;IN VSTAT TRANSMISSION WILL NOT OCCUR AND THIS ROUTINE
3183 ;WILL RESET XMIT INT. ENABLE. IF AFTER TRANSMISSION
3184 ;OF THE CHARACTER DURING THIS INTERRUPT CYCLE, THE
3185 ;XMIT COUNT (XMCNT) IS EQUAL TO ZERO,
3186 ;THE XMIT DONE BIT WILL BE SET IN VSTAT AND XMIT
3187 ;INT ENABLE BIT WILL CLEARED. TRANSMIT COUNT(XMCNT) MUST BE
3188 ;SET TO THE NUMBER OF BYTE/CHARACTER TO TRANSMIT.
3189 ;IF LOCATION BLKM IS SET TO 1001, A SOM WILL PRECEED THE
3190 ;DATA AND A EOM WILL FOLLOW IT. IF XMZER IS SET TO NON-
3191 ;ZERO, ALL DATA(INCLUDING ZEROS) WILL BE XMITTED.
3192 ;*****
3193 017164 005737 002420 INTXM: TST VSTAT ;HAS 61 TRANSMITTED XOFF?
3194 017170 100004 BPL NOKIL ;NO XMIT ANOTHER
3195 017172 052737 000200 002420 BIS #XMKIL,VSTAT ;SET XMIT KILLED BIT IN VSTAT
3196 017200 000523 BR KIFNA ;GO KILL XMIT ENABLE
3197

```

```

3198 017202 105737 002423      NOKIL: TSTB   BLKM+1      ;SOM/EOM TRANSMIT?
3199 017206 001410              BEQ     NOSOM      ;NO
3200 017210 112777 000002 162702      MOVB   #SOM,@VXBUF ;YES-ISSUE START OF MESSAGE.
3201 017216 005037 002142              CLR     NULL      ;CLEAR FLAG
3202 017222 105037 002423              CLRB   BLKM+1      ;AND CLEAR SOM FLAG.
3203 017226 000002              RTI
3204 017230 005737 017466      NOSOM: TST     XMCNT      ;XMITTED THE BUFFER?
3205 017234 001017              BNE     100$      ;NO-XMIT A NORMAL CHAR.
3206 017236 005737 002142              TST     NULL      ;SEND TRAILING NULL?
3207 017242 001006              BNE     101$      ;BR IF YES
3208 017244 112777 000004 162646      MOVB   #EOM,@VXBUF ;YES SEND EOM AND EXIT
3209 017252 005237 002142              INC     NULL      ;SET UP TRANSMIT OF TRAILING NULL
3210 017256 000460              BR      2$
3211 017260 112777 000000 162632 101$: MOVB   #0,@VXBUF ;LEAVE
3212 017266 105037 002422              CLRB   BLKM      ;SEND TRAILING NULL
3213 017272 000452              BR      2$
3214 017274 105777 000164              00$: TSTB   @TBUF P      ;CHECK FOR CH.= ZERO. IF SO DO NOT XMIT
3215 017300 001016              BNE     1$
3216 017302 005737 023072              TST     XMZER      ;OR COUNT BYTE. OR ARE WE
3217 017306 001023              BNE     22$      ;XMITTING ZEROS?
3218 017310 023737 017464 017462      CMP     TBUF P,TEB JF ;YES-XMIT NEXT BYTE
3219 017316 001004              BNE     10$      ;AT END OF BUFFER?
3220 017320 013737 017460 017464      MOV     TBBUF,TBUF P ;NO
3221 017326 000725              BR      NOKIL      ;YES-RESET BUFFER POINTER
3222 017330 005237 017464              10$: INC     TBUF P
3223 017334 000722              BR      NOKIL      ;LOOK FOR NON-ZERO BYTE TO TRANSMIT.
3224
3225 017336 032737 002000 002420 1$: BIT     #CKSUM,VSTAT ;CHECKSUM REQUESTED?
3226 017344 001404              BEQ     22$
3227 017346 117705 000112              MOVB   @TBUF P,R5 ;YES,LOAD THE BYTE
3228 017352 004037 022120              JSR     R0,CALCK    ;AND CALCULATE THE NEW CHECKSUM.
3229 017356 117777 000102 162534 22$: MOVB   @TBUF P,@VXBUF ;TRANSMIT A CHARACTER
3230 017364 023737 017464 017462      CMP     TBUF P,TEB JF ;AT END OF CIRCULAR BUFFER?
3231 017372 001004              BNE     11$      ;NO
3232 017374 013737 017460 017464      MOV     TBBUF,TBUF P ;YES, RESET IT TO START.
3233 017402 000402              BR      12$      ;BY-PASS INCREMENT BUFF. POINTER
3234 017404 005237 017464              11$: INC     TBUF P ;INCREMENT BUFFER POINTER.
3235
3236 017410 005337 017466              12$: DEC     XMCNT    ;DECREMENT THE TRANSMIT COUNT
3237 017414 001401              BEQ     2$
3238 017416 000002              RTI          ;YES,CLEANUP,REQUEST ERRORS AND EXIT.
3239 017420 105737 002422              2$: TSTB   BLKM      ;NO, CONTINUE
3240 017424 001014              BNE     TXEX      ;SOM/EOM XMIT?
3241 017426 052737 000001 002420      BIS     #XMDNE,VSTAT ;YES-DO NOT SET XMDNE UNTIL EOM SENT.
3242 017434 042737 002000 002420      BIC     #CKSUM,VSTAT ;SET THE DONE BIT IN VSTAT.
3243 017442 013737 017460 017464      MOV     TBBUF,TBUF P ;CLEAR THE CHECKSUM FLAG WHEN DONE.
3244 017450 042777 040000 162434 KIENA: BIC     #XENA,@VZCSR ;RESET BUFFER POINTER.
3245 017456 000002              TXEX: RTI        ;CLEAR XMIT. INT. ENABLE
3246
3247
3248 017460 000000              TBBUF: .WORD      ;CONTAINS INITIAL ADDRESS
3249 017462 000000              TEBUF: .WORD      ;CONTAIN LAST ADDRESS
3250 017464 000000              TBUF P: .WORD     ;CONTAINS CURRENT LOCATION
3251
3252 017466 000000              XMCNT: .WORD 0      ;LOADED WITH NUMBER OF XMITTS.
3253
;:*****

```

```

3254
3255
3256           ;SUBROUTINE TO ISSUE RESET TO THE VT61, ENTERS MAINTENANCE MODE
3257           ;AND FORCES LINEAR ADDRESSING.
3258           ;;*****
3259
3260 017470      RESETV:
3261 (2) 017470 013746 002364      MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
3262 017474 023727 002424 000022  CMP     TSTNM,#22      ;;TEST 22?
3263 017502 001004      BNE     10$      ;;BR IF NOT
3264 017504 113746 002312      MOV     DPNT,-(SP)      ;;MUST ALSO ISSUE ESC X
3265 017510 112746 000033      MOV     #33,-(SP)
3266 (2) 017514 013746 002324      MOV     RESET,-(SP)      ;;PUSH RESET ON STACK
3267 017520 013746 002250      MOV     ESCO,-(SP)      ;;PUSH ESCO ON STACK
3268 017524 004037 015464      JSR     R0,TESC      ;;GO XMIT IT
3269 017530 004037 017622      JSR     R0,GETON      ;;GO LOOK FOR XON.
3270 017534 000411      BR      1$      ;;FOUND IT.
3271 017544 003005      CMP     TSTNM,#20
3272 017546 005237 002374      BGT     1$
3273 017552 010037 001120      INC     FTLCNT      ;;ADD 1 TO FATAL XMIT COUNT.
3274 017556 104017      MOV     R0,$GDADR      ;;NO XON ISSUE XON ERROR
3275 017560      ERROR  17
3276 (2) 017560 013746 002364      MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
3277 017564 013746 002174      MOV     EMAN,-(SP)      ;;PUSH EMAN ON STACK
3278 017570 013746 002250      MOV     ESCO,-(SP)      ;;PUSH ESCO ON STACK
3279 017574 013746 002204      MOV     DRECT,-(SP)      ;;PUSH DRECT ON STACK
3280 017600 013746 002250      MOV     ESCO,-(SP)      ;;PUSH ESCO ON STACK
3281 017604 004037 015464      JSR     R0,TESC
3282 017610 005037 002420      CLR     VSTAT      ;;CLEAR INT. FLAGS AFTER TERMINAL RESET
3283 017614 005037 021312      CLR     HDFLG      ;;CLEAR PRINT HEADER FLAG.
3284 017620 000200      RTS     R0
3285
3286           ;;*****
3287           ;SUBROUTINE TO WAIT FOR AN XON. NO XON EXIT IS PC +2.
3288           ;;*****
3289 017622 012737 000454 002404  GETON:  MOV     #300.,BUBCT      ;;SET UP TO LOOK FOR 3 SEC.
3290 017630 105077 014040      CLRB    @ABUFP
3291 017634 127727 014034 000021  1$:    CMPB    @ABUFP,#XON      ;;RECEIVED A XON?
3292 017642 001412      BEQ     GOTON      ;;YES-EXIT.
3293 017644 012737 000001 021440      MOV     #1,DCOUNT      ;;NO-DELAY 10 M.S.
3294 017652 004037 021376      JSR     R0,DELAY
3295 017656 005337 002404      DEC     BUBCT      ;;AT END OF DELAY?
3296 017662 001364      BNE     1$      ;;NO
3297 017664 062700 000002      ADD     #2,R0      ;;YES-SET UP ERROR EXIT.
3298 017670 000200      GOTON:  RTS     R0
3299
3300           ;;*****
3301           ;SUBROUTINE TO ISSUE ESCZ AND LOOK FOR A RESPONSE-EITHER
3302           ;A -1 OR THE RETURNED IDENT. THE -1 INDICATES NO
3303           ;RESPONSE FROM THE UNIT UNDER TEST.
3304           ;;*****
3305
3306
  
```

```

3307 017672          ZFLAG:
(2) 017672 012637 017730      MOV      (SP)+,ROSV1      ;;POP STACK INTO ROSV1
3308 017676 013746 002364      MOV      @#ZERO,-(SP)      ;;PUSH @#ZERO ON STACK
(2) 017702 013746 002322      MOV      @#ESZ,-(SP)      ;;PUSH @#ESZ ON STACK
3309 017706 004037 015464      JSR      R0,TESE      ;;GO ISSUE ESZ SEQUENCE
3310 017712 012746 106003      MOV      #106003,-(SP)      ;;PUSH #106003 ON STACK
3311 017716 004037 015212      JSR      R0,RECTM      ;;GO READ THE CHARACTER
3312 017722 013746 017730      MOV      ROSV1,-(SP)      ;;PUSH ROSV1 ON STACK
3313 017726 000200      RTS      R0
3314 017730 000000      ROSV1: .WORD 0
3315
3316      ;;*****
3317      ;ROUTINE TO CHECK SOFTWARE STATUS REGISTER (VSTAT)
3318      ;RECEIVE FLAGS ONLY. ENTERED WITH ANTICIPATED
3319      ;STATUS WORD ON THE STACK.
3320      ;;*****
3321
3322 017732          CKSFT:
(2) 017732 012637 002416      MOV      (SP)+,ROSVE      ;;POP STACK INTO ROSVE
3323 017736 010137 002360      MOV      R1,SVER1      ;;SAVE R1
3324 017742 010237 002362      MOV      R2,SVER2      ;;SAVE R2
3325
3326 017746 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
3327 017750 013702 002420      MOV      VSTAT,R2      ;;SET R2 EQUAL TO VSTAT
3328
3329 017754 042702 003566      BIC      #003566,R2      ;;CLEAR NON-ERROR BITS
3330 017760 020102      CMP      R1,R2      ;;COMPARE ANTICIPATED TO ACTUAL.
3331 017762 001432      BEQ      NOER      ;;NO UNUSAL BITS EXIT
3332
3333 017764 010137 001124      MOV      R1,$GDDAT      ;;MOVE GOOD STATUS TO MESSAGE
3334 017770 013737 002420 001126      MOV      VSTAT,$BDDAT      ;;MOVE BAD STATUS TO MESSAGE
3335 017776 104003      ERROR      3      ;;ISSUE ERROR MESSAGE.
3336
3337
3338      ;;*****
3339      ;ROUTINE TO PRINT THE STATUS REGISTER IN THE FOLLOWING
3340      ;FORMAT: STATUS BITS (XXX 000), CHARACTER TRANSFERRED (000 X X)
3341      ;;*****
3342
3343 020000 012701 017134      MOV      #STTER,R1      ;;SET R1 EQUAL TO FIRST ENTRY
3344 020004 013702 017156      MOV      STTER,R2      ;;SET R2 EQUAL LAST ENTRY
3345 020010 020102      CMP      R1,R2      ;;ARE THEY EQUAL
3346 020012 001416      BEQ      NOER      ;;YES-RESET POINTERS AND EXIT.
3347 020014 004037 020074      JSR      R0,CLREG      ;;CLEAR ERROR PRINT LOC.
3348 020020 013737 002112 001120      MOV      VZCSR,$GDADR      ;;LOAD ADDRESS
3349 020026 017737 162060 001124      MOV      @VZCSR,$GDDAT      ;;LOAD CSR
3350 020034 112137 001126      2$: MOV      (R1)+,$BDDAT      ;;MOVE CHARACTER AND
3351 020040 112137 001123      MOV      (R1)+,$BDADR+1      ;;STATUS BITS TO ERROR REGISTERS.
3352 020044 104002      ERROR      2      ;;ISSUE ERROR MESSAGE
3353 020046 000760      BR      1$      ;;DO AGAIN
3354 020050 013701 002360      NOER: MOV      SVER1,R1      ;;RESTORE R1 AND
3355 020054 013702 002362      MOV      SVER2,R2      ;;R2.
3356 020060 012737 017134 017156      MOV      #STTER,STTER      ;;RESET STATUS ERROR POINTER.
3357 020066 013746 002416      MOV      ROSVE,-(SP)      ;;PUSH ROSVE ON STACK
3358 020072 000200      RTS      R0      ;;EXIT
3359

```

```

3360 ;:*****
3361 ;SUBROUTINE TO CLEAR ERROR/DATA OUTPUT LOCATIONS. NEEDED
3362 ;ONLY WHEN DISPLAYING BYTES IN WORD LOCATIONS.
3363 ;:*****
3364
3365 CLREG: CLR $GDADR
3366 CLR $BDADR
3367 CLR $GDDAT
3368 CLR $BDDAT
3369 RTS R0
3370 ;:*****
3371 ;SUBROUTINE TO TRANSMIT THE BUFFER AND WAIT FOR 'MIT DONE
3372 ;AND END OF RECEIVE MESSAGE. SUBROUTINE WILL LOC IF LOCATION
3373 ;RECITT IS PRE-LOADED WITH A NUMBER HIGHER THAN 1 (IE. MULTIPLE
3374 ;RECEIVES CAN BE ACCOMPLISHED WITH ONLY ONE ENTRY TO SUB-
3375 ;ROUTINE). WDSTOR AND BYSTOR ARE THE WORD (CURSOR POS.) AND BYTE
3376 ;STORAGE LOCATIONS, RESPECTIVELY. DEFAULT STORAGE IS THE REC. BUFFER.
3377
3378 ;:*****
3379
3380 XMREC:
3381 (2) 020116 010546 MOV R5, -(SP) ;: PUSH R5 ON STACK
3382 020120 012737 001001 002422 MOV #1001, BLKM ;: SET UP FOR A SOM/EOM TRANSMIT.
3383 020126 042737 077577 002420 BIC #77577, VSTAT ;: CLEAR ALL FLAGS BUT XOFF AND XMKIL.
3384 020134 013701 020370 MOV BYSTOR, R1 ;: LOAD THE STORAGE POINTERS
3385 020140 013702 020366 MOV WDSTOR, R2
3386 020144 052777 040000 161740 BIS #XENA, @VZCSR ;: SET INTERRUPT ENABLES
3387 020152 042737 061466 002420 XMITT: BIC #61466, VSTAT ;: CLEAR SOM, EOM, EPL, ESC, REV, VID., PARA, DELIM., IDENT, CUR.
3388 020160 005037 002414 1$: CLR DLAY ;: SET UP TIME OUT DELAY.
3389 020164 032737 000001 002420 BIT #XMDNE, VSTAT ;: IS XMIT DONE?
3390 020172 001015 BNE 3$ ;: YES-LOOK FOR RECEIVE DONE.
3391 020174 032737 020000 002420 2$: BIT #REOM, VSTAT ;: RECEIVED AN EOM?
3392 020202 001401 BFO 20$ ;: NO
3393 020204 000435 BR CKSTR ;: YES-GO HANDLE DATA
3394 020206 032737 100000 002420 20$: BIT #RXOFF, VSTAT ;: NO- IS XOFF SET?
3395 020214 001761 BEQ 1$ ;: NO-STILL TRANSMITTING.
3396 020216 005337 002414 DEC DLAY ;: YES- RUN DELAY
3397 020222 001364 BNE 2$ ;: WAITING FOR XON
3398 020224 000416 BR XMAD2 ;: NO XON-REPORT VT61 FAILURE.
3399 020226 013705 033674 3$: MOV ABUFP, R5 ;: LOAD CH. RECEIVED FLAG.
3400 020232 005037 002414 CLR DLAY ;: SET UP RECEIVE DELAY.
3401 020236 032737 020000 002420 4$: BIT #REOM, VSTAT ;: RECEIVE END OF MESSAGE?
3402 020244 001015 BNE CKSTR ;: YES-CHECK DATA STORAGE POINTERS
3403 020246 020537 033674 CMP R5, ABUFP ;: RECEIVED ANOTHER CHARACTER?
3404 020252 001365 BNE 3$ ;: YES-RESET CH. FLAG AND DELAY
3405 020254 005337 002414 5$: DEC DLAY ;: RUN DELAY
3406 020260 001366 BNE 4$ ;: AND KEEP LOOKING FOR EOM.
3407 020262 062700 000002 XMAD2: ADD #2, R0 ;: TIME OUT OCCURRED-SET UP ERROR EXIT.
3408 020266 005237 002374 INC FTLCNT ;: INCREMENT FATAL XMIT COUNT.
3409 020272 004037 020502 JSR R0, RESPTR ;: AND REST ALL INTERRUPT POINTERS.
3410 020276 000422 BR CKVST
3411 020300 020102 CKSTR: CMP R1, R2 ;: STORAGE POINTERS CLEARED?
3412 020302 001413 BEQ CHKITT ;: YES--LEAVE DATA IN REC. BUFFER.
3413 020304 032737 000004 002420 BIT #CJRPOS, VSTAT ;: RECEIVED A CURSOR POSITION?
3414 020312 001403 BEQ STRBYT ;: NO-GO STORE A BYTE.

```

```

3415 020314 017722 176604      MOV      @RBBUF,(R2)+ ;YES,STORE IT.
3416 020320 000404      BR      CHKITT ;AND CHECK ITERATION COUNT.
3417 020322 005701      STRBYT: TST      R1 ;STORING A CHAR?
3418 020324 001402      BEQ      CHKITT ;NO
3419 020326 117721 176572      MOVB     @RBBUF,(R1)+ ;STORE A RECEIVED BYTE
3420 020332 005337 020364      CHKITT: DEC      RECITT ;DONE RECEIVING?
3421 020336 001305      BNE      XMITT ;NO-LOOP SUBROUTINE
3422 020340 004037 023170      JSR      RO,CKOFF ;SEE IS XOFF IS UP.
3423 020344      CKVST:
3424 020344 012746 060001      MOV      #60001,-(SP) ;:PUSH #60001 ON STACK
3425 020350 004037 017732      JSR      RO,CKSFT
3426 020354 004037 020502      JSR      RO,RESPTR ;RESET INTERRUPT POINTERS.
3427 020360 012605      MOVB     (SP)+,R5 ;:POP STACK INTO R5
3428 020362 000200      XMITT: RTS      RO ;EXIT SUBROUTINE.
3429 020364 000000      RECITT: .WORD 0 ;RECEIVE ITERATION COUNT.
3430 020366 000000      WDSTOR: .WORD 0 ;WORD STORAGE POINTER
3431 020370 000000      BYSTOR: .WORD 0 ;BYTE STORAGE POINTER
3432      ;:*****
3433      ;SUBROUTINE TO XMIT THE BYTE AT TBUF.
3434      ;:*****
3435
3436 020372 042737 000001 002420      XMIT1: BIC      #1,VSTAT ;CLEAR XMIT DONE FLAG
3437 020400 012737 000001 017466      MOV      #1,XMCNT ;SET UP TO XMIT 1 BYTE
3438 020406 052777 040000 161476      BIS      #XENA,@VZCSR
3439 020414      1$:
3440 020414 012746 000001      MOV      #XMDNE,-(SP) ;:PUSH #XMDNE ON STACK
3441 020420 012746 000001      MOV      #1,-(SP) ;:PUSH #1 ON STACK
3442 020424 004037 023074      JSR      RO,WTBGND ;LOOK FOR XMIT DONE
3443 020430 000401      BR      FTLEXT ;HUNG TRANSMIT-CLEAR FLAGS AND EXIT
3444 020432 000402      BR      NORXT ;NORMAL EXIT.
3445 020434 005037 002420      FTLEXT: CLR      VSTAT ;CLEAR ANY FLAGS
3446 020440 000200      NORXT: RTS      RO ;AND EXIT
3447      ;:*****
3448      ;SUBROUTINE TO ISSUE A BYTE AT A TIME UNTIL A ZERO
3449      ;BYTE IS ENCOUNTERED.
3450      ;:*****
3451
3452 020442 112777 000002 177014      LDXTMIT: MOVB     #SOM,@TBUF ;SEND THE START OF MESSAGE.
3453 020450 000403      BR      2$
3454 020452 112377 177006      1$: MOVB     (R3)+,@TBUF ;MOVE A BYTE TO XMIT BUFFER
3455 020456 001403      BEQ      LDOUT ;IF A ZERO BYTE-EXIT
3456 020460 004037 020372      2$: JSR      RO,XMIT1 ;GO XMIT A BYTE
3457 020464 000772      BR      1$ ;XMIT AGAIN.
3458 020466 112777 000004 176770      LDOUT: MOVB     #EOM,@TBUF ;SEND THE END OF MESSAGE.
3459 020474 004037 020372      JSR      RO,XMIT1
3460 020500 000200      RTS      RO
3461
3462      ;:*****
3463      ;ROUTINE TO RESET ALL INTERRUPT POINTERS.
3464      ;:*****
3465
3466 020502 042777 040000 161402      RESPTR: BIC     #XENA,@VZCSR ;CLEAR INTERRUPT ENABLES
3467 020510 013737 017124 017130      MOV      R2BUF,RBUF ;RESET RECEIVE BUF POINTER
3468 020516 013737 017460 017464      MOV      TBUF,TBUF ;RESET XMIT BUF POINTER

```

```

3469 020524 012737 017134 017156      MOV      #STTER,STTEP      ;RESET RECEIVE STATUS ERR POINTER
3470 020532 005037 017466              CLR      XMCNT          ;CLEAR TRANSMIT COUNT
3471 020536 005037 017132              CLR      ESAMB          ;CLEAR ESC ASSEMBLY FLAGS
3472 020542 012737 000001 020364      MOV      #1,RECITT       ;RESET REC. ITERATION COUNT
3473 020550 005037 020366              CLR      WDSTOR         ;CLEAR STORAGE POINTERS
3474 020554 005037 020370              CLR      BYSTOR
3475 020560 000200                      RTS      R0
3476
3477
3478
3479      ;*****
3480      ;SUBROUTINE TO ISSUE CURSOR POSITION ERROR. GOOD
3481      ;LINE/COLUMN MUST BE A WORD ON STACK. ERROR
3482      ;POSITION IS EXPECTED TO BE @ RBBUF.
3483      ;*****
3484 020562 012637 002416      CURER:  MOV      (SP)+,ROSVE      ;POP STACK INTO ROSVE
3485 020566 012637 002356      MOV      (SP)+,CHRD      ;POP STACK INTO CHRD
3486 020572 162737 020040 002356      SUB      #20040,CHRD      ;EXTRACT MOD 40 FROM GOOD POSITION
3487 020600 004037 020074      JSR      R0,CLREG
3488 020604 113737 002357 001124      MOV      CHRD+1,$GDDAT    ;LOAD MESSAGE WITH GOOD
3489 020612 113737 002356 001120      MOV      CHRD,$GDADR      ;LINE AND COLUMN
3490 020620 017737 176300 002356      MOV      @RBBUF,CHRD      ;LINE AND COLUMN.
3491 020626 162737 020040 002356      SUB      #20040,CHRD      ;EXTRACT MOD 40 FROM BAD POSITION.
3492 020634 113737 002357 001126      MOV      CHRD+1,$BDDAT    ;LOAD MESSAGE WITH BAD
3493 020642 113737 002356 001122      MOV      CHRD,$BDADR      ;LINE AND COLUMN.
3494 020650 104006              ERROR      6          ;ISSUE ERROR
3495 020652 013746 002416      MOV      ROSVE,-(SP)      ;PUSH ROSVE ON STACK
3496 020656 000200                      RTS      R0
3497
3498
3499
3500
3501      ;*****
3502      ;SUBROUTINE TO DECREMENT CURSOR POSITION IN A
3503      ;LINEAR SEQUENCE. (IE. ROW 20, COL 1 ;ROW 20 COL 0 ;ROW 17, COL 157).
3504      ;*****
3505 020660 123727 020765 000040  CMPOS:  CMP      LNRW+1,#40      ;AT LEFT EDGE OF ROW?
3506 020666 001403              BEQ      1$          ;YES, GO ADJUST COL., ROW.
3507 020670 105337 020765              DECB     LNRW+1          ;NO, DECREMENT COL. AND EXIT
3508 020674 000200                      RTS      R0
3509 020676 123727 020764 000040  1$:    CMP      LNRW,#40      ;AT ROW 0?
3510 020704 001405              BEQ      2$          ;YES, NO DECREMENT POSSIBLE-EXIT.
3511 020706 105337 020764              DECB     LNRW          ;NO, DECREMENT ROW AND
3512 020712 112737 000157 020765      MOV      #157,LNRW+1    ;SET COL. TO RIGHT EDGE.
3513 020720 000200              RTS      R0
3514
3515      ;*****
3516      ;SUBROUTINE TO INCREMENT CURSOR POSITION IN A LINEAR
3517      ;SEQUENCE (IE. ROW 10, COL 78, ROW 10. COL 79, ROW 11 , COL 0).
3518      ;*****
3519
3520 020722 123727 020765 000157  CPPOS:  CMP      LNRW+1,#157     ;AT RIGHT EDGE OF ROW
3521 020730 001403              BEQ      1$          ;YES, ADJUST ROW AND COLUMN.
3522 020732 105237 020765              INCB     LNRW+1          ;NO, INCREMENT COL. COUNT
3523 020736 000200                      RTS      R0          ;AND EXIT

```

```

3524 020740 123727 020764 000067 1$: CMPB LNRW,#67 ;AT BOTTOM ROW?
3525 020746 001405 BEQ 2$ ;YES, NO INCREMENT POSSIBLE-EXIT.
3526 020750 105237 020764 INCB LNRW ;NO, INCREMENT ROW COUNT AND
3527 020754 112737 000040 020765 MOVB #40,LNRW+1 ;SET COL. TO LEFT EDGE.
3528 020762 000200 2$: RTS R0
3529
3530 020764 000000 LNRW: .WORD 0 ;CONTAINS UPDATED CURSOR POSITION.
3531 ;:*****
3532
3533 ;SUBROUTINE TO XMIT, RECEIVE AND COMPARE. DATA ERRORS
3534 ;ARE REPORTED FROM SUBROUTINE. IF THE TRANSMIT OR
3535 ;RECEIVE LOOPS 'TIME OUT', EXIT FROM SUBROUTINE WILL
3536 ;BE NORMAL EXIT +2. SUBROUTINE ENTERED WITH (R1)=
3537 ;GOOD DATA BUFFER, (R2)-RECEIVE DATA BUFFER AND
3538 ;R3-COMPARE COUNT. IF THE VT61 DOES NOT HANG,THE ROUTINE
3539 ;WILL WAIT FOR END OF REC. MESSAGE(EOM).
3540
3541
3542 ;:*****
3543
3544 020766 XRCMP: MOV R4,-(SP) ;PUSH R4 ON STACK
(2) 020766 010446 CLR R4 ;USE R4 A RECEIVE COUNTER.
3545 020770 005004 MOV #1001,BLKM ;SET UP FOR A SOM/EOM TRANSMIT.
3546 020772 012737 001001 002422 BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
3547 021000 042737 077577 002420 BIS #XENA,@VZCSR ;SET INTERRUPT ENABLES.
3548 021006 052777 040000 161076 CLR HDFLG ;CLEAR ERROR 13 PRINT FLAG
3549 021014 005037 021312 MOV #TCRLB+450,R5 ;R5 IS ERROR STORAGE POINTER
3550 021020 012705 033644 1$: CLR DLAY ;SET UP TIME OUT DELAY
3551 021024 005037 002414 BIT #XMDNE,VSTAT ;XMIT DONE?
3552 021030 032737 000001 002420 BNE XREC ;YES-GO RECEIVE
3553 021036 001014 2$: CMP RBBUF,RBUFP ;HAS RECEIVE OPERATION BEGUN?
3554 021040 023737 017124 017130 BLO XREC ;YES-GO RECEIVE
3555 021046 103410 BIT #RXOFF,VSTAT ;XMIT XOFF SET?
3556 021050 032737 100000 002420 BEQ 1$ ;NO-KEEP LOOKING FOR XMIT DONE?
3557 021056 001762 DEC DLAY ;YES RUN DELAY AND LOOK
3558 021060 005337 002414 BNE 2$ ;FOR XON OR RECEIVED CH.
3559 021064 001365 BR XRERR ;TRANSMIT TIMEOUT-SET UP ERROR EXIT
3560 021066 000432
3561
3562 021070 005037 002414 XREC: CLR DLAY ;SET UP TIME OUT DELAY
3563 021074 020237 017130 1$: CMP R2,RBUFP ;INSURE COMPARE POINTER
3564 021100 103410 BLO 2$ ;LESS THAN RECEIVE POINTER
3565 021102 032737 020000 002420 BIT #REOM,VSTAT ;RECEIVE EOM?
3566 021110 001030 BNE XREXT ;YES-SET UP TO EXIT
3567 021112 005337 002414 DEC DLAY ;RUN TIMEOUT DELAY
3568 021116 001416 BEQ XRERR ;TIME OUT OCCURRED-ERROR EXIT
3569 021120 000765 BR 1$ ;RETURN TO CHECK RECEIVE COUNT
3570 021122 005204 2$: INC R4 ;ADD 1 TO RECEIVE COUNTER.
3571 021124 122122 CMPB (R1)+,(R2)+ ;COMPARE CHARACTERS
3572 021126 001407 BEQ 4$ ;EQUAL-COMPARE AGAIN
3573 021130 020527 033674 CMP R5,#TCRLB+500 ;ALLREADY STORED 50 ERRORS?
3574 021134 103004 BHIS 4$ ;YES-BYPASS STORAGE
3575 021136 114125 MOVB -(R1),(R5)+ ;STORE GOOD DATA
3576 021140 114225 MOVB -(R2),(R5)+ ;STORE BAD DATA
3577 021142 010425 MOV R4,(R5)+ ;LOAD RECEIVE COUNT
3578 021144 132122 BITB (R1)+,(R2)+ ;RESET POINTERS AND

```

```

3579 021146 005303 4$: DEC R3 ;CHECK COMPARE COUNT
3580 021150 001410 BEQ XREXT ;ALL DONE-EXIT
3581 021152 000746 BR XREC ;COMPARE ANOTHER
3582 021154 062700 000002 XRERR: ADD #2,R0 ;SET UP ERROR EXIT
3583 021160 005237 002374 INC FTLCNT ;INCREMENT FATAL XMIT COUNT.
3584 021164 004037 020502 JSR R0,RESPTR ;RESET INTERRUPT POINTERS.
3585 021170 000440 BR XROUT
3586 021172 XREXT: MOV #REOM,-(SP) ;;PUSH #REOM ON STACK
(2) 021172 012746 020000 MOV #4,-(SP) ;;PUSH #4 ON STACK
3587 021176 012746 000004 JSR R0,WTBGND
3588 021202 004037 023074 BR XROUT ;NO EOM-ISSUE ERROR AND EXIT.
3589 021206 000431 SUB #TCRLB+450,R5 ;NOW EXTRACT ERROR COUNT-IF ANY.
3590 021210 162705 033644 MOV R5,R1 ;AND STORE IT IN R1
3591 021214 010501 MOV #TCRLB+450,R5 ;RELOAD ERROR POINTER
3592 021216 012705 033644 TST R1 ;TEST FOR ERRORS
3593 021222 005701 BEQ XROUT ;NO-CHECK STATUS AND EXIT
3594 021224 001422 TST HDFLG ;;DATA ERROR HEADER PRINTED?
3595 021226 005737 021312 BNE 1$ ;YES-BYPASS HEADER PRINT
3596 021232 001003 ERROR 12 ;PRINT DATA ERROR HEADER
3597 021234 104012 INC HDFLG ;SET HEADER PRINT FLAG
3598 021236 005237 021312 1$: JSR R0,CLREG ;ERROR WAS LEGTIMATE. LOAD
3599 021242 004037 020074 MOV (R5)+,$GDDAT ;ERROR MESSAGE AND ISSUE
3600 021246 112537 001124 MOV (R5)+,$BDDAT ;IT.
3601 021252 112537 001126 MOV (R5)+,$GDADR ;LOAD RECEIVE COUNT
3602 021256 012537 001120 ERROR 4 ;ISSUE DATA COMPARE ERROR
3603 021262 104004 SUB #4,R1 ;DECREMENT ERROR COUNT
3604 021264 162701 000004 BNE 1$ ;PRINT ANOTHER IF NOT AT ZERO
3605 021270 001364 XRROUT: JSR R0,CKOFF ;SEE IS XOFF IS UP.
3606 021272 004037 023170 MOV #60001,-(SP) ;PUSH #60001 ON STACK
3607 021276 012746 060001 JSR R0,CKSFT ;CHECK FOR VSTAT /STATUS ERR.
3608 021302 004037 017732 MOV (SP)+,R4 ;POP STACK INTO R4
3609 021306 012604 RTS R0 ;EXIT SUBROUTINE
3610 021310 000200
3611
3612 021312 000000 HDFLG: 0 ;INHIBIT PRINT FLAG.
3613
3614 ;;*****
3615
3616 ;SUBROUTINE TO CREATE A 'RULER' IN LOCATIONS 200
3617 ;TO 317.
3618
3619 ;;*****
3620
3621 021314 012701 033374 CPRUL: MOV #TCRLB+200,R1 ;LOAD STARTING ADDRESS
3622 021320 012702 130461 MOV #130461,R2 ;LOAD INITIAL RULER ASCII CODES.
3623 021324 110221 1$: MOV R2,(R1)+ ;STORE A RULER BYTE IN XMIT BUF.
3624 021326 022701 033514 CMP #TCRLB+320,R1 ;RULER COMPLETE?
3625 021332 103001 BHS 2$ ;NO
3626 021334 000200 RTS R0 ;AND EXIT.
3627 021336 105202 2$: INCB R2 ;INCREMENT ASCII BYTE
3628 021340 122702 000272 CMPB #272,R2 ;END OF REVERSE VIDEO?
3629 021344 001003 BNE 3$ ;NO-SEE IF END OF NORMAL.
3630 021346 012702 030660 MOV #030660,R2 ;SET UP TO ISSUE REVERSE 0.
3631 021352 000405 BR 5$
3632 021354 122702 000072 3$: CMPB #72,R2 ;END OF NORMAL VIDEO?
3633 021360 001361 BNE 1$ ;NOT AT END OF A VIDEO STRING.

```

```

3634 021362 012702 130460      MOV      #130460,R2      ;YES-SET UP TO ISSUE NORMAL 0.
3635 021366 110221      5$:  MOV      R2,(R1)+      ;DO IT
3636 021370 105202      INCB      R2      ;SET BYTE TO NEXT ASCII CODE
3637 021372 000302      SWAB      R2      ;REVERSE VIDEO MODE.
3638 021374 000753      BR        1$      ;BEGIN NEXT STRING
3639
3640      ;*****
3641      ;SUBROUTINE TO DELAY 20 M.S. TIME THE NUMBER INLOCATION
3642      ;DCOUNT. THE PROCESSOR TYPE PRE-DETERMINES THE # OF LOOPS
3643      ;REQUIRED TO DELAY 20 M.S. FOR ONE ITERATION. LOCATION
3644      ;PMULT IS PRE-LOADED WITH : 11/45 = 4, 11/40 = 2
3645      ;AND 11/10 =1.
3646      ;*****
3647
3648 021376      DELAY:
3649 (2) 021376 010146      MOV      R1,-(SP)      ;PUSH R1 ON STACK
3650 (2) 021400 010246      MOV      R2,-(SP)      ;PUSH R2 ON STACK
3651 021402 013702 021436 1$:  MOV      PMULT,R2      ;LOAD PROCESSOR MULTIPLIER
3652 021406 012701 005360 2$:  MOV      #2800.,R1      ;LOAD 20 M.S. DELAY
3653 021412 005301      DEC      R1      ;RUN BASIC DELAY
3654 021414 001376      BNE      .-2
3655 021416 005302      DEC      R2      ;RUN MULTIPLIER DELAY
3656 021420 001372      BNE      2$
3657 021422 005337 021440      DEC      DCOUNT      ;RUN ITERATION COUNT
3658 021426 001365      BNE      1$
3659 021430 012602      MOV      (SP)+,R2      ;POP STACK INTO R2
3660 (2) 021432 012601      MOV      (SP)+,R1      ;POP STACK INTO R1
3661 021434 000200      RTS      R0
3662
3663      PMULT: 0      ;PROCESSOR MULTIPLIER
3664      DCOUNT: 0      ;ITERATION COUNT
3665
3666      ;*****
3667      ;SUBROUTINE TO GENERATE A INCREMENTING PATTERN AT
3668      ;(R1)+. ENTER WITH R3 EQUAL TO # OF CH. TO CREATE.
3669      ;R5 IS UTILIZED AS A WORK REGISTER.
3670      ;*****
3671 021442 012705 000041      BLDINC: MOV      #41,R5      ;LOAD R5 WITH INITIAL CH.
3672 021446 110521      BLDINA: MOVB      R5,(R1)+      ;MOVE A CH. TO BUFFER
3673 021450 005303      DEC      R3      ;DECREMENT BYTE COUNT
3674 021452 001001      BNE      2$      ;NOT DONE-UPDATE PATTERN
3675 021454 000200      RTS      R0      ;EXIT-DONE.
3676 021456 105205      2$:  INCB      R5      ;UPDATE CH. PATTERN
3677 021460 122705 000177      CMPB      #177,R5      ;PATTERN EXCEEDED MAX?
3678 021464 001766      BEQ      BLDINC      ;YES-RESET IT.
3679 021466 000767      BR        BLDINA      ;NO-ISSUE CURRENT PATTERN.
3680
3681      ;*****
3682      ;SUBROUTINE TO FILL THE SCREEN WITH INCREMENTING DATA
3683      ;*****
3684
3685
3686

```

```

3687 021470 042737 077577 002420 DATSC: BIC #77577,VSTAT ;CLEAR INTERRUPT FLAGS.
3688 021476 013701 017460 MOV TBBUF,R1
3689 021502 012703 000500 MOV #320,R3 ;FILL XMIT BUFFER WITH INCRE-
3690 021506 004037 021442 JSP R0,BLDINC ;MENTING PATTERN
3691 021512 012737 003600 017466 10$: MOV #TOTCH,XMCNT ;SET UP TO XMIT 1920 BYTES
3692 021520 052777 040000 160364 BIS #XENA,@VZCSR
3693
3694 021526 032737 000001 002420 1$: BIT #XMDNE,VSTAT ;XMIT DONE?
3695 021534 001774 BEQ -6 ;NO
3696
3697 ;*****
3698
3699 ;SUBROUTINE TO RESET VT61 AND DISPLAY MESSAGE
3700 ;POINTED TO BY R2.
3701
3702 ;*****
3703
3704 021536 004037 017470 DSMES: JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
3705 021542 042737 077577 002420 BIC #77577,VSTAT ;CLEAR ALL FLAGS EXCEPT XOFF AND XMKIL.
3706 021550 012737 000005 017466 MOV #5,XMCNT ;PRE-LOAD XMIT COUNT.
3707 021556 013701 017460 MOV TBBUF,R1 ;LOAD XMIT BUFFER WITH:
3708 021562 012721 000002 MOV #SOM,(R1)+ ;START OF MESSAGE
3709 021566 013721 002250 MOV ESCO,(R1)+
3710 021572 013721 002204 MOV DRECT,(R1)+ ;DISABLE RECTANGULAR MODE
3711 021576 005237 017466 1$: INC XMCNT ;INCREMENT TRANSMIT COUNT
3712 021602 112221 MOV# (R2)+,(R1)+ ;DISPLAY MESSAGE
3713 021604 001374 BNE 1$
3714 021606 112711 000004 MOV# #EOM,(R1, ;TERMINATE WITH END OF MESSAGE.
3715 021612 052777 040000 160272 BIS #XENA,@VZCSR ;XMIT IT AND WAIT FOR
3716 021620 032737 000001 002420 2$: BIT #XMDNE,VSTAT ;DONE
3717 021626 001774 BEQ 2$
3718 021630 000200 RTS R0
3719
3720 ;*****
3721
3722 ;SUBROUTINE TO CONVERT A BINARY CHARACTER
3723 ;TO 3 OCTAL CHARACTERS. R1 CONTAINS BINARY
3724 ;NUMBER. RESULT IS STORED IN LOCATIONS SVER1,
3725 ;SVER2
3726
3727 ;*****
3728 021632 BINOC: MOV R5,-(SP) ;:PUSH R5 ON STACK
3729 (2) 021632 010546 MOV #2,R5 ;:LOAD ITERATION COUNT
3730 021634 012705 000002 BR 2$ ;:BYPASS SHIFTS FOR 1ST CONVERSION
3731 021640 000403 1$: ASRB R1
3732 021642 106201 ASRB R1 ;:SHIFT A CHAR INTO POSITION
3733 021644 106201 ASRB R1
3734 021646 106201 2$: MOV# R1,SVER1(R5) ;:STORE THE BINARY OFFSET
3735 021650 110165 002360 BIC# #370,SVER1(R5) ;:CLEAR NON ESSENTIAL BITS
3736 021654 142765 000370 002360 BIS# #60,SVER1(R5) ;:CONVERT OFFSET TO OCTAL
3737 021662 152765 000060 002360 DEC R5 ;:DECREMENT CONVERSION COUNT
3738 021670 005305 BPL 1$ ;:NOT DONE CONVERT ANOTHER
3739 021672 100363 MOV# #40,SVER2+1 ;:LOAD A SPACE
3740 021674 112737 000040 002363 MOV (R5)+,R5 ;:POP STACK INTO R5
3741 021702 012605 RTS R0
3741 021704 000200

```

```

3742
3743
3744
3745
3746
3747
3748 021706 042701 177770 OCTBIN: BIC #177770,R1 ;EXTRACT OCTAL COMPONENT
3749 021712 005702 TST R2 ;FIRST CONVERSION?
3750 021714 001403 BEQ NOSHFT ;YES - DO NOT SHIFT
3751 021716 006302 ASL R2 ;NO - SHIFT PREVIOUS CHAR.
3752 021720 006302 ASL R2
3753 021722 006302 ASL R2
3754 021724 060102 NOSHFT: ADD R1,R2 ;ADD CURRENT CHAR.
3755 021726 000200 RTS R0
3756
3757
3758
3759 021730 005037 002126
3760 021734 013737 002124 002406
3761 021742 032737 000001 002406
3762 021750 001005
3763 021752 006037 002406
3764 021756 005237 002126
3765 021762 000767
3766 021764 000200
3767
3768
3769
3770
3771
3772 021766 032777 000200 160116
3773 021774 001774
3774 021776 127737 160112 002146
3775 022004 001370
3776 022006 000200
3777
3778
3779
3780
3781
3782
3783
3784
3785 022010 004037 015640
3786 022014 012601
3787 022016 122701 000054
3788 022022 001411
3789 022024 123701 002146
3790 022030 001406
3791 022032 120127 000060
3792 022036 103421
3793 022040 120127 000067
3794 022044 101016
3795 022046 110137 022116
3796 022052 104401 022116
3797 022056 123701 002150

;*****
;SUBROUTINE TO CONVERT AN OCTAL CHAR. TO BINARY. REG
;R1 CONTAINS OCTAL AND REG R2 IS BINARY ASSEMBLY AREA.
;*****

;*****
;SUBROUTINE TO CONVERT A BINARY POSITION TO A OCTAL #.
;*****
CONVLN: CLR OCTLNE
MOV TSTLNE,TPREG ;LOAD OCTLNE WITH OCTAL EQUIVALENT
101$: BIT #BIT00,TPREG ;OF TSTLNE IN BITS 8 THRU 11.
BNE 102$
ROR TPREG
INC OCTLNE
BR 101$
102$: RTS R0

;*****
;ROUTINE TO WAIT FOR C/R FROM VT61 UNDER TEST
;*****

GTCR: BIT #RECDN,@VZCSR ;WAIT FOR REVEIVE DONE
BEQ .-6
CMPB @VRBUF,CARRT ;CHAR = CARRIAGE RETURN?
BNE GTCR ;NO-KEEP LOOKING
RTS R0 ;YES-EXIT

;*****
;SUBROUTINE TO GET A CHARACTER (NUMERIC) FROM THE
;CONSOLE. IF OTHER THAN A NUMERIC IS TYPED A
; '?' WILL BE ECHOED.
;*****

GTNUM: JSR R0,CONRD ;GET A CHAR
MOV (SP)+,R1 ;POP STACK INTO R1
CMPB #54,R1 ;CHAR. =COMMA?
BEQ 1$ ;YES-GO PRINT IT
CMPB CARRT,R1 ;CHAR. = CARRIAGE RETURN?
BEQ 1$
CMPB R1,#60
BLO QUST ;IF CHAR. IS LESS THAN 60
CMPB R1,#67 ;OR MORE THAN 67, TYPE
BHI QUST ;A QUESTION MARK
1$: MOVB R1,TPNUM
TYPE ,TPNUM
CMPB LNFEED,R1

```

```

3798 022062 001406      BEQ      GTEXT
3799 022064 123701 002146  CMPB    CARRT,R1      ;IF CHAR. - C/R SET UP TO ISSUE
3800 022070 001003      BNE      GTEXT      ;LINE FEED BEFORE EXITING.
3801 022072 113701 002150  MOVB    LNFED,R1
3802 022076 000763      BR       1$
3803 022100 000200      GTEXT:  RTS      R0      ;GOOD CHAR., EXIT
3804 022102 112737 000077 022116  QUST:  MOVB    #77,TYPNUM
3805 022110 104401 022116      TYPE    TYPNUM      ;TYPE QUESTION MARK AND
3806 022114 000735      BR       GTNUM      ;KEEP LOOKING.
3807 022116      000      TYPNUM: .BYTE    0
3808 022117      000      .BYTE    0
3809
3810
3811      ;*****
3812      ;SUBROUTINE TO CALCULATE CHECKSUM ON THE LOWER
3813      ;BYTE OF R5. R4 IS STORAGE FOR THE CHECKSUM
3814      ;CHARACTER. ALGORITHM FOR CHECKSUM IS ROTATE
3815      ;CURRENT ONE PLACE LEFT AND XOR NEW CHAR. CHECKSUM
3816      ;IS THE LOWER 7 BITS OF R4
3817
3818      ;*****
3819
3820 022120 042705 177400  CALCK:  BIC      #177400,R5      ;CLEAR UPPER BYTE OF R5
3821 022124 120527 000021  CMPB    R5,#XON      ;CHAR. -XON?
3822 022130 001415      BEQ      NOCALC      ;YES DO NOT CALCULATE CHECKSUM
3823 022132 120527 000023  CMPB    R5,#XOFF      ;CHAR =XOFF?
3824 022136 001412      BEQ      NOCALC      ;YES DO NOT CALCULATE CHECKSUM
3825
3826 022140 000241      CLC
3827 022142 105704      TSTR     R4      ;INSURE CARRY BIT INITIALLY CLEAR
3828 022144 100001      BPL      1$      ;SET UP TO ROTATE R4
3829      ;A FULL 8 BYTES
3830 022146 000261      SEC
3831 022150 106104 1$:      ROLB    R4      ;R4 WAS NEG. SO ROTATE A ONE
3832 022152 010403      MOV      R4,R3      ;INTO LOW ORDER BIT.
3833 022154 040503      BIC      R5,R3      ;NOT A AND B
3834 022156 040405      BIC      R4,R5      ;NOT B AND A
3835 022160 050305      BIS      R3,R5      ;ORED
3836 022162 010504      MOV      R5,R4      ;EQUAL NEW CHECKSUM
3837 022164 000200  NOCALC:  RTS      R0
3838
3839      ;*****
3840      ;SUBROUTINE TO LOAD XMIT BUFFER FROM R0 THRU R1
3841      ;*****
3842
3843 022166 112021  LDBUF:  MOVB    (R0)+,(R1)+      ;LOAD A BYTE
3844 022170 001376      BNE      -2      ; UNTIL ZERO BYTE FOUND.
3845 022172 000200      RTS      R0
3846
3847      ;*****
3848      ;SUBROUTINE TO CHECK THE VT61 FOR A PERIPHERAL ABORT.
3849      ;*****
3850 022174 032737 010000 002420  (KABRT:  BIT      #PABRT,VSTAT      ;ABORT FLAG RECEIVED?
3851 022202 001446      BEQ      2$      ;NO-EXIT
3852 022204 010037      MOV      R0,$GDDAT
3853 022210 162737 000004 001124      SUB      #4,$GDDAT      ;POINT ERR PC TO MAIN ROUTINE.

```

```

3854 022216 013737 002420 001126      MOV    VSTAT,$BDDAT
3855 022224 104020                ERROR  20      ;ISSUE PERIPHERAL ABORT ERROR
3856
3857 022226 013701 017460      MOV    TBBUF,R1
3858 022232 004037 022166      JSR    R0,LDBUF      ;LOAD THE XMIT BUFFER WITH:
3859 022236      033      117      137      .BYTE  .ESC,.O,.IABT,.ESC,.O,.RABT
      022241      033      117      140
3860 022244      033      117      145      .BYTE  .ESC,.O,.UNLKKB,.BEL,0,0
      022247      007      000      000
3861 022252 012737 000007 017466      MOV    #7,XMCNT      ;SET UP TO XMIT 7 BYTES.
3862 022260 004037 020116      JSR    R0,XMREC      ;XMIT AND RECEIVE.
3863 022264 000240      NOP
3864 022266 123727 017160 000170      CMPB   STRO,#NABRT      ;ABORT FLAG CLEARED?
3865 022274 001411      BEQ    2$      ;YES-EXIT
3866 022276 010037 001124      MOV    R0,$GDDAT      ;NO-SET UP AND ISSUE A LANT
3867 022302 162737 000004 001124      SUB    #4,$GDDAT      ;CLEAR ABORT FLAG ERROR MESSAGE.
3868 022310 013737 002420 001126      MOV    VSTAT,$BDDAT
3869 022316 104021      ERROR  21
3870 022320 000200      2$:      RTS    RC
3871
3872      ;*****
3873      ;SUBROUTINE TO COMPARE RECEIVED KEYBOARD DATA WITH
3874      ;DATA EXPECTED. ERRORS ARE REPORTED AS POSITIONAL
3875      ;ERROR ONLY.
3876      ;*****
3877
3878
3879 022322 105077 011346      CKKBD:  CLRB   @ABUFP      ;CLEAR RECEIVE BYTE
3880 022326 005037 002356      CLR    CHR      ;CLEAR INPUT STORAGE.
3881 022332 105777 011336      KBDLP:  TSTB  @ABUFP      ;WAIT FOR A INPUT.
3882 022336 001775      BEQ    -4
3883
3884 022340 117737 011330 002356      MOVB   @ABUFP ,CHR      ;STORE IT AND
3885 022346 105077 011322      CLRB   @ABUFP      ;CLEAR THE INPUT AREA.
3886 022352 123714 002356      1$:      CMPB   CHR      ,(R4)      ;RECEIVED EQUAL EXPECTED?
3887 022356 001500      BEQ    GDSTRK      ;NO-UPDATE POINTERS.
3888 022360 005237 002404      INC     BUUCT      ;INCREMENT ERROR COUNT.
3889 022364 023727 002404 0000*2      CMP     BUUCT,#10.      ;COUNT = 10?
3890 022372 103075      BHIS   CNTF      ;YES-EXIT SUBROUTINE.
3891 022374 010401      MOV     R4,R1
3892 022376 166501 013022      SUB     DTIBL(R5),R1      ;EXTRACT KEY POSITION FROM ROW LOC.
3893 022402 005201      INC     R1      ;CONVERT LOGICAL POS. TO ACTUAL.
3894 022404 004037 021632      JSR     R0,BINOC      ;GET KEY POSITION IN OCTAL.
3895 022410 113737 002362 002360      MOVB   SVER2,SVER1      ;RE-ASSEMBLE OCTAL BYTES.
3896 022416 123727 002361 000060      CMPB   SVER1+1,#60      ;POSITION LESS THAN 8?
3897 022424 001413      BEQ     LDPOS      ;YES-GO LOAD IT.
3898 022426 123727 002360 000062      CMPB   SVER1,#62      ;POSITION GREATER THAN 8 AND LESS THAN12?
3899 022434 103404      BLO     BOROW      ;YES-SET UP TO BORROW.
3900 022436 162737 000002 002360      SUB     #2,SVER1      ;NO-JUST SUBTRACT 2.
3901 022444 000403      BR      LDPOS
3902 022446 162737 000370 002360      BOROW: SUB     #370,SVER1      ;SUBTRACT AND BORROW.
3903 022454 113737 002360 032305      LDPOS: MOVB   SVER1,KYSTRK+1      ;LOAD THE CONVERTED DECIMAL #.
3904 022462 113737 002361 032304      MOVB   SVER1+1,KYSTRK
3905 022470 012703 032227      DMP OCT: MOV    #DKBERR ,R3
3906 022474 004037 020442      JSR     RC      ;ISSUE BODY OF KEYBOARD ERROR.
3907 022500 111401      MOVB   (R4) ,R1
  
```

```

3908 022502 004037 021632      JSR      R0      ,BINOCT
3909 022506 012703 002360      MOV      #SVER1 ,R3
3910 022512 004037 020442      JSR      R0      ,LDXMIT ;CONVERT AND ISSUE GOOD CHAR.
3911 022516 012703 032336      MOV      #DSPC6,R3
3912 022522 004037 020442      JSR      R0,LDXMIT      ;INSERT 6 SPACES IN MESSAGE.
3913 022526 113701 002356      MOVB     CHR      ,R1
3914 022532 004037 021632      JSR      R0      ,BINOCT
3915 022536 012703 002360      MOV      #SVER1 ,R3
3916 022542 004037 020442      JSR      R0      ,LDXMIT ;CONVERT AND ISSUE RECEIVED CHAR.
3917 022546 012703 001171      MOV      #CRLF   ,R3
3918 022552 004037 020442      JSR      R0      ,LDXMIT ;ISSUE C/R AND L/F.
3919 022556 000665              BR       KBDLP      ;LOOK FOR SAME KEY AGAIN.
3920
3921 022560 005204      GDSTRK: INC      R4      ;INCREMENT KEYBOARD ROW COUNTER.
3922 022562 105714      TSTB     (R4)      ;REACHED END OF ROW?
3923 022564 001262      BNE      KBDLP      ;NO-LOOK FOR NEXT INPUT
3924 022566 000200      NTF:   RTS      R0      ;YES-EXIT.
3925
3926      ;*****
3927
3928      ;SUBROUTINE TO LOOP DATA THROUGH HOST COMPUTER. ALL
3929      ;FUNCTIONS ARE ALLOWED, BUT BLOCK TRANSMITS WHICH
3930      ;EXCEED 552 BYTES WILL RESULT IN THE TERMINATION
3931      ;OF THE OPERATION AFTER 552 RECEIVED BYTES.
3932
3933      ;*****
3934
3935 022570 005237 023072      LOOP.   INC      XMZER      ;SET UP TO XMIT NULLS.
3936 022574 012737 033674 017126      MOV      #TCRLB+500,REBUF      ;RESET BUFFER POINTERS
3937 022602 012737 032474 017460      MOV      #RCRLB,TBBUF
3938 022610 004037 020502      JSR      R0,RESPTR      ;RELOAD ALL INTERRUPT POINTERS
3939 022614 042737 077577 002420      BIC      #77577,VSTAT      ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
3940 022622 013704 017130      LOOPPT: MOV      RBUF,R4      ;SET UP RECEIVE FLAG
3941 022626 032737 000001 002420      LOOPTA: BIT      #XMDNE,VSTAT      ;XMIT COMPLETE?
3942 022634 001407              BEQ      LOOPR      ;NO

```

3944	022636	042737	000001	002420		BIC	#XMDNE,VSTAT	;YES RESET FLAG
3945	022644	013737	017124	017130		MOV	RBBUF,RBUF	;RESET THE REC. BUFFER POINTER
3946	022652	000763				BR	LOPT	
3947	022654	032737	001400	002420	LOOPR:	BIT	#EPL+ESC,VSTAT	;RECEIVED AN ESC OR EPL?
3948	022662	001004				BNE	LPSTR	;YES-GO CHECK IT
3949	022664	023704	017130			CMP	RBUF,R4	;RECEIVED A DISPLAY CHAR?
3950	022670	001756				BEQ	LOPTA	;NO-LOOP
3951	022672	000426				BR	BUMPCT	
3952	022674	117777	010774	174226	LPSTR:	MOVB	@ABUF,@RBUF	;YES LOAD IT IN THE BUFFER
3953	022702	005237	017130			INC	RBUF	;AND INCREMENT BUFFER POINTER
3954	022706	005037	017132			CLR	ESAMB	;CLEAR ESC ASSEMBLY WORD
3955	022712	042737	001400	002420		BIC	#EPL+ESC,VSTAT	;CLEAR THE FLAGS
3956	022720	005237	017466			INC	XMCNT	;INCREMENT XMIT COUNT
3957	022724	123777	002326	010742		CMPB	ESCN,@ABUF	;CHAR. A ESC(033)?
3958	022732	001733				BEQ	LOPT	;YES WAIT FOR NEXT PART OF FUNCTION
3959	022734	113777	002150	174166		MOVB	LNFEED,@RBUF	;CHAR. WAS EPL ADD A LINE FEED.
3960	022742	005237	017130			INC	RBUF	
3961	022746	000407				BR	FRCECT	;AND ISSUE THEM.
3962	022750	023727	017466	000764	BUMPCT:	CMP	XMCNT,#500.	;BUFFER ABOUT FILLED?
3963	022756	103403				BLO	FRCECT	;NO
3964	022760	005337	017130		1\$:	DEC	RBUF	;YES-RESET THE RECEIVE POINTER

```

3968 022764 000716
3969 022766 005237 017466
3970 022772 023727 017466 000002
3971 023000 101003
3972 023002 052777 040000 157102
3973 023010 004037 023020
3974 023014 000702
3975 023016 000200
3976
3977
3978
3979
3980
3981
3982 023020 127727 010650 000003
3983 023026 001020
3984
3985 023030 012737 033173 017126
3986 023036 012737 033174 017460
3987 023044 004037 020502
3988 023050 012702 031115
3989 023054 004037 021536
3990 023060 005037 023072
3991 023064 062700 000002
3992 023070 000200
3993
3994 023072 000000
3995
3996
3997
3998
3999
4000
4001 023074
4002 023074 012637 002416
4003 023100 012637 023166
4004 023104 012637 023164
4005 023110 006337 023166
4006 023114 005037 002414
4007 023120 033737 023164 002420
4008 023126 001012
4009 023130 005337 002414
4010 023134 001371
4011 023136 005337 023166
4012 023142 001364
4013 023144 104011
4014 023146 005237 002374
4015 023152 000401
4016 023154 005720
4017 023156 013746 002416
4018 023162 000200
4019 023164 000000
4020 023166 000000
4021

FRCECT: BR      LOOPT
          INC     XMCNT      ;INCREMENT THE XMIT COUNT
          CMP     XMCNT,#2   ;FIRST CHAR TO XMIT?
          BHI     XMTT      ;NO
          BIS     #XENA,@VZCSR ;YES-SET THE XMIT ENABLE
          JSR     RO,EXTST    ;LOOK FOR END OF TEST COMMAND.
          BR      LOOPT      ;NONE FOUND.
          RTS     RO         ;AND EXIT

;*****
;SUBROUTINE TO CHECK FOR END OF TEST COMMAND. THE CONTROL
;C KEY EXITS ALL TESTS.
;*****

EXTST:  CMPB     @ABUFP,#3    ;LOOK FOR CONTROL C.
        BNE     NOROUT

ABSTXT: MOV     #RCRLB+477,REBUF ;RESET THE BUFFERS
        MOV     #TCRLB,TBBUF
        JSR     RO,RESPTR     ;RESET ALL POINTERS
        MOV     #DEXT,R2
        JSR     RO,DSMES      ;ISSUE EXIT MESSAGE
        CLR     XMZER         ;CLEAR THE ZERO TRANSMIT FLAG.
        ADD     #2,RO         ;SET UP TEST EXIT.
        NOROUT: RTS     RO    ;EXIT SUBROUTINE.

XMZER:  .WORD    0

;*****
;SUB-ROUTINE TO LOOK FOR VSTAT BIT ON THE STACK
;DELAY FACTOR IS FIRST WORD ON THE STACK AND VSTAT BIT
;IS THE SECOND. MIN. DELAY IS 4 U.S FOR A MOS 11/45.
;*****

WTBGND: MOV     (SP)+,ROSVE    ;POP STACK INTO ROSVE
        MOV     (SP)+,VDLAY    ;POP STACK INTO VDLAY
        MOV     (SP)+,VBIT     ;POP STACK INTO VBIT
        ASL     VDLAY          ;DOUBLE FACTOR FOR LOW BAUD
1$:     CLR     DLAY
2$:     BIT     VBIT,VSTAT     ;SENSED THE CONDITION?
        BNE     FNDBT         ;YES-EXIT.
        DEC     DLAY          ;NO-RUN DELAY.
        BNE     2$
        DEC     VDLAY         ;DELAY FACTOR EXPIRED?
        BNE     1$           ;NO-LOOP
        ERROR   11            ;DELAY EXPIRED-ISSUE HUNG NIT
        INC     FTLCNT        ;INCREMENT FATAL XMIT COUNT.
        BR      TIMEXT
FNDBT:  TST     (RO)+         ;SET UP FOR NORMAL EXIT
TIMEXT: MOV     ROSVE,-(SP)    ;PUSH ROSVE ON STACK
        RTS     RO
VBIT:   0
VDLAY:  0

;*****
;SUBROUTINE TO LOOK FOR XOFF BEFORE EXITING A RECEIVE ROUTINE.

```

```

4022      ;:*****
4023
4024 023170 005037 002414 CKOFF: CLR      DLAY
4025 023174 032737 100000 002420 1$: BIT      #RXOFF,VS*AT ;IS XOFF SET?
4026 023202 001403          BEQ      2$      ;NO-EXIT
4027 023204 005337 002414          DEC      DLAY      ;RUN DELAY.
4028 023210 001371          BNE      1$
4029 023212 000200          RTS      RO
4030
4031      .SBTTL  SCOPE HANDLER ROUTINE
(1)
(2)      ;:*****
(1)      ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(1)      ;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(1)      ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
(1)      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)      ;*SW14=1      LOOP ON TEST
(1)      ;*SW11=1      INHIBIT ITERATIONS
(1)      ;*SW09=1      LOOP ON ERROR
(1)      ;*SW08=1      LOOP ON TEST IN SWR<7:0>
(1)      ;*CALL
(1)      ;*      SCOPE      ;:SCOPE IOT
(1)
(1) 023214      $SCOPE:
(2) 023214 004037 015672      JSR      RO,MONIT
(1) 023220 032777 040000 155712 1$: BIT      #BIT14,@SWR      ;:LOOP ON PRESENT TEST?
(1) 023226 001111          BNE      $OVER      ;:YES IF SW14=1
(1)      ;:####START OF CODE FOR THE XOR TESTER####
(1) 023230 000416      $XTSTR: BR      6$      ;:IF RUNNING ON THE 'XOR' TESTER CHANGE
(1)      ;:THIS INSTRUCTION TO A 'NOP' (NOP 240)
(1) 023232 013746 000004      MOV      @#ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
(1) 023236 012737 023256 000004      MOV      #5$,@#ERRVEC ;:SET FOR TIMEOUT
(1) 023244 005737 177060      TST      @#177060 ;:TIME OUT ON XOR?
(1) 023250 012637 000004      MOV      (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
(1) 023254 000463          BR      $SVLAD ;:GO TO THE NEXT TEST
(1) 023256 022626      5$: CMP      (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
(1) 023260 012637 000004      MOV      (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
(1) 023264 000423          BR      7$      ;:LOOP ON THE PRESENT TEST
(1) 023266      6$:;####END OF CODE FOR THE XOR TESTER####
(1) 023266 032777 000400 155644      BIT      #BIT08,@SWR      ;:LOOP ON SPEC. TEST?
(1) 023274 001404          BEQ      2$      ;:BR IF NO
(1) 023276 127737 155636 001102      CMPB     @SWR,$STNM ;:ON THE RIGHT TEST? SWR<7:0>
(1) 023304 001462          BEQ      $OVER      ;:BR IF YES
(1) 023306 105737 001103      2$: TSTB     $ERFLG ;:HAS AN ERROR OCCURRED?
(1) 023312 001421          BEQ      3$      ;:BR IF NO
(1) 023314 123737 001115 001103      CMPB     $ERMAX,$ERFLG ;:MAX. ERRORS FOR THIS TEST OCCURRED?
(1) 023322 101015          BHI      3$      ;:BR IF NO
(1) 023324 032777 001000 155606      BIT      #BIT09,@SWR ;:LOOP ON ERROR?
(1) 023332 001404          BEQ      4$      ;:BR IF NO
(1) 023334 013737 001110 001106      7$: MOV      $LPERR,$LPADR ;:SET LOOP ADDRESS TO LAST SCOPE
(1) 023342 000443          BR      $OVER
(1) 023344 105037 001103      4$: CLRB     $ERFLG ;:ZERO THE ERROR FLAG
(1) 023350 005037 001160          CLR      $TIMES ;:CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1) 023354 000415          BR      1$      ;:ESCAPE TO THE NEXT TEST
(1) 023356 032777 004000 155554      3$: BIT      #BIT11,@SWR ;:INHIBIT ITERATIONS?
(1) 023364 001011          BNF      1$      ;:BR IF YES

```

```

(1) 023366 005737 001100      TST      $PASS      ;; IF FIRST PASS OF PROGRAM
(1) 023372 001406              BEQ      1$          ;; INHIBIT ITERATIONS
(1) 023374 005237 001104      INC      $ICNT      ;; INCREMENT ITERATION COUNT
(1) 023400 023737 001160 001104  CMP      $TIMES,$ICNT  ;; CHECK THE NUMBER OF ITERATIONS MADE
(1) 023406 002021              BGE      $OVER      ;; BR IF MORE ITERATION REQUIRED
(1) 023410 012737 000001 001104 1$:      MOV      #1,$ICNT  ;; REINITIALIZE THE ITERATION COUNTER
(1) 023416 013737 023466 001160      MOV      $MXCNT,$TIMES  ;; SET NUMBER OF ITERATIONS TO DO
(1) 023424 105237 001102      $SVLAD: INCB      $TSTNM  ;; COUNT TEST NUMBERS
(1) 023430 011637 001106      MOV      (SP),$LPADR  ;; SAVE SCOPE LOOP ADDRESS
(1) 023434 011637 001110      MOV      (SP),$LPERR  ;; SAVE ERROR LOOP ADDRESS
(1) 023440 005037 001162      CLR      $ESCAPE  ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
(1) 023444 112737 000001 001115      MOV      #1,$ERMAX  ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1) 023452 013777 001102 155462 $OVER:  MOV      $TSTNM,@DISPLAY  ;; DISPLAY TEST NUMBER
(1) 023460 013716 001106      MOV      $LPADR,(SP)  ;; FUDGE RETURN ADDRESS
(1) 023464 000002              RTI              ;; FIXES PS
(1) 023466 000005              $MXCNT: 5        ;; MAX. NUMBER OF ITERATIONS
4032 .SBTTL  ERROR HANDLER ROUTINE
(1)
(2)
(1) *****
(1) *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.
(1) *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1) *AND GO TO $ERRTYP ON ERROR
(1) *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1) *SW15=1      HALT ON ERROR
(1) *SW13=1      INHIBIT ERROR TYPEOUTS
(1) *SW10=1      BELL ON ERROR
(1) *SW09=1      LOOP ON ERROR
(1) *CALL
(1) *      ERROR      N      ;; ERROR EMT AND N=ERROR ITEM NUMBER
(1)
(1) 023470      $ERROR:
(1) 023470 105237 001103      7$:      INCB      $ERFLG      ;; SET THE ERROR FLAG
(1) 023474 001775              BEQ      7$          ;; DON'T LET THE FLAG GO TO ZERO
(1) 023476 013777 001102 155436      MOV      $TSTNM,@DISPLAY  ;; DISPLAY TEST NUMBER AND ERROR FLAG
(1) 023504 032777 002000 155426      BIT      #BIT10,@SWR  ;; BELL ON ERROR?
(1) 023512 001402              BEQ      1$          ;; NO - SKIP
(1) 023514 104401 001164      TYPE      ,SBELL  ;; RING BELL
(1) 023520 005237 001112      1$:      INC      $ERTTL  ;; COUNT THE NUMBER OF ERRORS
(1) 023524 011637 001116      MOV      (SP),$ERRPC  ;; GET ADDRESS OF ERROR INSTRUCTION
(1) 023530 162737 000002 001116      SUB      #2,$ERRPC
(1) 023536 117737 155354 001114      MOV      @ $ERRPC,$ITEMB  ;; STRIP AND SAVE THE ERROR ITEM CODE
(1) 023544 032777 020000 155366      BIT      #BIT13,@SWR  ;; SKIP TYPEOUT IF SET
(1) 023552 001004              BNE      20$        ;; SKIP TYPEOUTS
(1) 023554 004737 024056      JSR      PC,$ERRTYP  ;; GO TO USER ERROR ROUTINE
(1) 023560 104401 001171      TYPE      ,CRLF
(1) 023564      20$:
(1) 023564 005777 155350      2$:      TST      @SWR      ;; HALT ON ERROR
(1) 023570 100001              BPL      3$          ;; SKIP IF CONTINUE
(1) 023572 000000              HALT
(1) 023574 032777 001000 155336 3$:      BIT      #BIT09,@SWR  ;; LOOP ON ERROR SWITCH SET?
(1) 023602 001402              BEQ      4$          ;; BR IF NO
(1) 023604 013716 001110      MOV      $LPERR,(SP)  ;; FUDGE RETURN FOR LOOPING
(1) 023610 005737 001162      4$:      TST      $ESCAPE  ;; CHECK FOR AN ESCAPE ADDRESS
(1) 023614 001402              BEQ      5$          ;; BR IF NONE
(1) 023616 013716 001162      5$:      MOV      $ESCAPE,(SP)  ;; FUDGE RETURN ADDRESS FOR ESCAPE
(1) 023622

```

```

(1) 023622 022737 011742 000042      CMP      #SENDAD,@#42      ;;ACT-11 AUTO-ACCEPT?
(1) 023630 001001                      BNE      6$              ;;BRANCH IF NO
(1) 023632 000000                      HALT                      ;;YES
(1) 023634                                6$:
(1) 023634 000002                      RTI                          ;;RETURN
4033 .SBTTL TYPE ROUTINE

(1)
(2)
(1) *****
(1) *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(1) *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(1) *NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(1) *NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(1) *NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(1) *
(1) *CALL:
(1) *1) USING A TRAP INSTRUCTION
(1) *      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(1) *OR
(1) *      TYPE
(1) *      MESADR
(1) *
(1)
(1) 023636 105737 001157      $TYPE:  TSTB      $TPFLG      ;;IS THERE A TERMINAL?
(1) 023642 100002                      BPL      1$              ;;BR IF YES
(1) 023644 000000                      HALT                      ;;HALT HERE IF NO TERMINAL
(1) 023646 000407                      BR      3$              ;;LEAVE
(1) 023650 010046      1$:      MOV      R0,-(SP)      ;;SAVE R0
(1) 023652 017600 000002      MOV      @2(SP),R0      ;;GET ADDRESS OF ASCIZ STRING
(1) 023656 112046      2$:      MOVB     (R0)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1) 023660 001005                      BNE      4$              ;;BR IF IT ISN'T THE TERMINATOR
(1) 023662 005726                      TST      (SP)+      ;;IF TERMINATOR POP IT OFF THE STACK
(1) 023664 012600      60$:     MOV      (SP)+,R0      ;;RESTORE R0
(1) 023666 062716 000002      3$:      ADD      #2,(SP)      ;;ADJUST RETURN PC
(1) 023672 000002                      RTI                          ;;RETURN
(1) 023674 122716 000011      4$:      CMPB     #HT,(SP)      ;;BRANCH IF <HT>
(1) 023700 001430                      BEQ      8$
(1) 023702 122716 000200      CMPB     #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1) 023706 001006                      BNE      5$
(1) 023710 005726                      TST      (SP)+      ;;POP <CR><LF> EQUIV
(1) 023712 104401                      TYPE                      ;;TYPE A CR AND LF
(1) 023714 001171      $CRLF
(1) 023716 105037 024052      CLRB     $CHARCNT      ;;CLEAR CHARACTER COUNT
(1) 023722 000755                      BR      2$              ;;GET NEXT CHARACTER
(1) 023724 004737 024006      5$:      JSR      PC,$TYPEPC      ;;GO TYPE THIS CHARACTER
(1) 023730 123726 001156      6$:      CMPB     $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
(1) 023734 001350                      BNE      2$              ;;IF NO GO GET NEXT CHAR.
(1) 023736 013746 001154      MOV      $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
(1)                                ;;AND THE NULL CHAR.
(1) 023742 105366 000001      7$:      DECB     1(SP)      ;;DOES A NULL NEED TO BE TYPED?
(1) 023746 002770                      BLT      6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1) 023750 004737 024006      JSR      PC,$TYPEPC      ;;GO TYPE A NULL
(1) 023754 105337 024052      DECB     $CHARCNT      ;;DO NOT COUNT AS A COUNT
(1) 023760 000770                      BR      7$              ;;LOOP
(1)
(1) ;HORIZONTAL TAB PROCESSOR

```

```

(1) 023762 112716 000040 8$: MOVB #' (SP) ;;REPLACE TAB WITH SPACE
(1) 023766 004737 024006 9$: JSR PC,$TYPEC ;;TYPE A SPACE
(1) 023772 132737 000007 024052 BITB #7,$CHARCNT ;;BRANCH IF NOT AT
(1) 024000 001372 BNE 9$ ;;TAB STOP
(1) 024002 005726 TST (SP)+ ;;POP SPACE OFF STACK
(1) 024004 000724 BR 2$ ;;GET NEXT CHARACTER
(1) 024006 105777 155136 $TYPEC: TSTB @STPS ;;WAIT UNTIL PRINTER IS READY
(1) 024012 100375 BPL $TYPEC
(1) 024014 116677 000002 155130 MOVB 2(SP),@STPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1) 024022 122766 000015 000002 CMPB #CR,2(SP) ;;IS CHARACTER A CARRIAGE RETURN?
(1) 024030 001003 BNE 1$ ;;BRANCH IF NO
(1) 024032 105037 024052 CLRB $CHARCNT ;;YES--CLEAR CHARACTER COUNT
(1) 024036 000406 BR $TYPEX ;;EXIT
(1) 024040 122766 000012 000002 1$: CMPB #LF,2(SP) ;;IS CHARACTER A LINE FEED?
(1) 024046 001402 BEQ $TYPEX ;;BRANCH IF YES
(1) 024050 105227 INCB (PC)+ ;;COUNT THE CHARACTER
(1) 024052 000000 $CHARCNT: .WORD 0 ;;CHARACTER COUNT STORAGE
(1) 024054 000207 $TYPEX: RTS PC
  
```

4034 .SBTTL ERROR MESSAGE TYPEOUT ROUTINE

```

(1) ;;*****
(2) ;;THIS ROUTINE USES THE 'ITEM CONTROL BYTE' ($ITEMB) TO DETERMINE WHICH
(1) ;;ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE 'ERROR TABLE' ($ERRTB),
(1) ;;AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
(1) $ERRTYP:
(1) 024056 104401 001171 TYPE $CRLF ;;'CARRIAGE RETURN' & 'LINE FEED'
(1) 024062 010046 MOV R0,-(SP) ;;SAVE R0
(1) 024064 005000 CLR R0 ;;PICKUP THE ITEM INDEX
(1) 024066 153700 001114 BISB @4$ITEMB,R0
(1) 024072 001004 BNE 1$ ;;IF ITEM NUMBER IS ZERO, JUST
(2) 024074 013746 001116 MOV $ERRPC,-(SP) ;;TYPE THE PC OF THE ERROR
(2) ;;SAVE $ERRPC FOR TYPEOUT
(2) 024100 104402 TYPOC ;;ERROR ADDRESS
(1) 024102 000445 BR 10$ ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1) 024104 005300 1$: DEC R0 ;;GET OUT
(1) 024106 006300 ASL R0 ;;ADJUST THE INDEX SO THAT IT WILL
(1) 024110 006300 ASL R0 ;;WORK FOR THE ERROR TABLE
(1) 024112 006300 ASL R0
(1) 024114 062700 001174 ADD # $ERRTB,R0 ;;FORM TABLE POINTER
(1) 024120 012037 024130 MOV (R0)+,2$ ;;PICKUP 'ERROR MESSAGE' POINTER
(1) 024124 001404 BEQ 3$ ;;SKIP TYPEOUT IF NO POINTER
(1) 024126 104401 TYPE ;;TYPE THE 'ERROR MESSAGE'
(1) 024130 000000 2$: .WORD 0 ;;'ERROR MESSAGE' POINTER GOES HERE
(1) 024132 104401 001171 TYPE $CRLF ;;'CARRIAGE RETURN' & 'LINE FEED'
(1) 024136 012037 024146 3$: MOV (R0)+,4$ ;;PICKUP 'DATA HEADER' POINTER
(1) 024142 001404 BEQ 5$ ;;SKIP TYPEOUT IF 0
(1) 024144 104401 TYPE ;;TYPE THE 'DATA HEADER'
(1) 024146 000000 4$: .WORD 0 ;;'DATA HEADER' POINTER GOES HERE
(1) 024150 104401 001171 TYPE $CRLF ;;'CARRIAGE RETURN' & 'LINE FEED'
(1) 024154 010146 5$: MOV R1,-(SP) ;;SAVE R1
(1) 024156 012001 MOV (R0)+,R1 ;;PICKUP 'DATA TABLE' POINTER
(1) 024160 001415 BEQ 9$ ;;BR IF NO DATA TO BE TYPED
(1) 024162 012000 MOV (R0)+,R0 ;;PICKUP 'DATA FORMAT' POINTER
  
```

```

(1) 024164 105720      6$:  TSTB      (R0)+      ;; 'OCTAL' OR 'DECIMAL'
(1) 024166 001003      BNE      7$      ;; BR IF DECIMAL
(2) 024170 013146      MOV      @ (R1)+, -(SP)  ;; SAVE @ (R1)+ FOR TYPEOUT
(2) 024172 104402      TYPDC      8$      ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
(1) 024174 000402      BR      8$
(1) 024176      7$:      MOV      @ (R1)+, -(SP)  ;; SAVE @ (R1)+ FOR TYPEOUT
(2) 024176 013146      TYPDS      8$      ;; GO TYPE--DECIMAL ASCII WITH SIGN
(2) 024200 104405      TST      (R1)      ;; IS THERE ANOTHER NUMBER?
(1) 024202 005711      BEQ      9$      ;; BR IF NO
(1) 024204 001403      TYPE      ,11$      ;; TYPE TWO(2) SPACES
(1) 024206 104401 024226 BR      6$      ;; LOOP
(1) 024212 000764
(1) 024214 012601      9$:  MOV      (SP)+, R1      ;; RESTORE R1
(1) 024216 012600      10$:  MOV      (SP)+, R0      ;; RESTORE R0
(1) 024220 104407 001171 TYPE      , $CRLF      ;; 'CARRIAGE RETURN' & 'LINE FEED'
(1) 024224 000207      RTS      PC      ;; RETURN
(1) 024226 020040 000      11$:  .ASCIIZ  / /      ;; TWO(2) SPACES
(1) 024232 024232      .EVEN
4035 .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(2)
(1)
(1) *****
(1) *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1) *OCTAL (ASCII) NUMBER AND TYPE IT.
(1) *$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1) *CALL:
(1) *      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
(1) *      TYPOS      ;; CALL FOR TYPEOUT
(1) *      .BYTE      N      ;; N 1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1) *      .BYTE      M      ;; M-1 OR 0
(1) *      ;; 1=TYPE LEADING ZEROS
(1) *      ;; 0-SUPPRESS LEADING ZEROS
(1)
(1) *$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1) *$TYPOS OR $TYPOC
(1) *CALL:
(1) *      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
(1) *      TYPON      ;; CALL FOR TYPEOUT
(1)
(1) *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1) *CALL:
(1) *      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
(1) *      TYPOC      ;; CALL FOR TYPEOUT
(1)
(1) 024232 017646 000000      $TYPOS: MOV      @ (SP), -(SP)      ;; PICKUP THE MODE
(1) 024236 116637 000001 024455 MOVB      1 (SP), $OFILL      ;; LOAD ZERO FILL SWITCH
(1) 024244 112637 024457      MOVB      (SP)+, $OMODE+1      ;; NUMBER OF DIGITS TO TYPE
(1) 024250 062716 000002      ADD      #2, (SP)      ;; ADJUST RETURN ADDRESS
(1) 024254 000406      BR      $TYPON
(1) 024256 112737 000001 024455 $TYPOC: MOVB      #1, $OFILL      ;; SET THE ZERO FILL SWITCH
(1) 024264 112737 000006 024457      MOVB      #6, $OMODE+1      ;; SET FOR SIX(6) DIGITS
(1) 024272 112737 000005 024454 $TYPON: MOVB      #5, $OCNT      ;; SET THE ITERATION COUNT
(1) 024300 010346      MOV      R3, -(SP)      ;; SAVE R3
(1) 024302 010446      MOV      R4, -(SP)      ;; SAVE R4
(1) 024304 010546      MOV      R5, -(SP)      ;; SAVE R5
(1) 024306 113704 024457      MOVB      $OMODE+1, R4      ;; GET THE NUMBER OF DIGITS TO TYPE

```



```

(3) 024466 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
(3) 024470 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
(1) 024472 012746 020200 MOV      #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1) 024476 016605 000020 MOV      20(SP),R5      ;;GET THE INPUT NUMBER
(1) 024502 100004      BPL      1$          ;;BR IF INPUT IS POS.
(1) 024504 005405      NEG      R5          ;;MAKE THE BINARY NUMBER POS.
(1) 024506 112766 000055 000001 1$: MOVB   #'-,1(SP)      ;;MAKE THE ASCII NUMBER NEG.
(1) 024514 005000      CLR      R0          ;;ZERO THE CONSTANTS INDEX
(1) 024516 012703 024674 MOV      #SDBLK,R3      ;;SETUP THE OUTPUT POINTER
(1) 024522 112723 000040 MOVB   #' ,(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
(1) 024526 005002      CLR      R2          ;;CLEAR THE BCD NUMBER
(1) 024530 016001 024664 2$: MOV      $DTBL(R0),R1      ;;GET THE CONSTANT
(1) 024534 160105      SUB      R1,R5      ;;FORM THIS BCD DIGIT
(1) 024536 002402      BLT      4$          ;;BR IF DONE
(1) 024540 005202      INC      R2          ;;INCREASE THE BCD DIGIT BY 1
(1) 024542 000774      BR       3$
(1) 024544 060105      ADD      R1,R5      ;;ADD BACK THE CONSTANT
(1) 024546 005702      TST      R2          ;;CHECK IF BCD DIGIT-0
(1) 024550 001002      BNE      5$          ;;FALL THROUGH IF 0
(1) 024552 105716      TSTB     (SP)        ;;STILL DOING LEADING 0'S?
(1) 024554 100407      BMI      7$          ;;BR IF YES
(1) 024556 106316      ASLB     (SP)        ;;MSD?
(1) 024560 103003      BCC      6$          ;;BR IF NO
(1) 024562 116663 000001 177777 6$: MOVB   1(SP),-1(R3)      ;;YES--SET THE SIGN
(1) 024570 052702 000060 7$: BIS      #'0,R2      ;;MAKE THE BCD DIGIT ASCII
(1) 024574 052702 000040 BIS      #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1) 024600 110223      MOVB     R2,(R3)+      ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1) 024602 005720      TST      (R0)+      ;;JUST INCREMENTING
(1) 024604 020027 000010      CMP      R0,#10      ;;CHECK THE TABLE INDEX
(1) 024610 002746      BLT      2$          ;;GO DO THE NEXT DIGIT
(1) 024612 003002      BGT      8$          ;;GO TO EXIT
(1) 024614 010502      MOV      R5,R2      ;;GET THE LSD
(1) 024616 000764      BR       6$          ;;GO CHANGE TO ASCII
(1) 024620 105726      8$: TSTB     (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
(1) 024622 100003      BPL      9$          ;;BR IF NO
(1) 024624 116663 177777 177776 9$: MOVB   -1(SP),-2(R3)      ;;YES--SET THE SIGN FOR TYPING
(1) 024632 105013      CLRB     (R3)        ;;SET THE TERMINATOR
(3) 024634 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
(3) 024636 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
(3) 024640 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
(3) 024642 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
(3) 024644 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
(1) 024646 104401 024674      TYPE     ,SDBLK      ;;NOW TYPE THE NUMBER
(1) 024652 016666 000002 000004 MOV      2(SP),4(SP)      ;;ADJUST THE STACK
(1) 024660 012616      MOV      (SP)+,'SP)
(1) 024662 000002      RTI
(1) 024664 023420      $DTBL: 10000.      ;;RETURN TO USER
(1) 024666 001750      1000.
(1) 024670 000144      100.
(1) 024672 000012      10.
(1) 024674 000004      $SDBLK: .BLKW 4
4037 .SBTTL POWER DOWN AND UP ROUTINES
(1)
(2)
(1)
(1) 024704 012737 025056 000024 $PWRDN: MOV      #SILLUP,#PWRVEC ;;SET FOR FAST UP

```

```

(1) 024712 012737 000340 000026      MOV      #340,@#PWRVEC+2 ;;PRIO:7
(3) 024720 010046                      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
(3) 024722 010146                      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
(3) 024724 010246                      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
(3) 024726 010346                      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
(3) 024730 010446                      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
(3) 024732 010546                      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
(3) 024734 017746 154200              MOV      @SWR,-(SP)    ;;PUSH @SWR ON STACK
(1) 024740 010637 025062              MOV      SP,$SAVR6    ;;SAVE SP
(1) 024744 012737 024756 000024      MOV      #SPWRUP,@#PWRVEC ;;SET UP VECTOR
(1) 024752 000000                      HALT
(1) 024754 000776                      BR        .-2          ;;HANG UP
(1)
(2)
(1)
(1) 024756 012737 025056 000024      $PWRUP: MOV      #SILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1) 024764 013706 025062              MOV      $SAVR6,SP    ;;GET SP
(1) 024770 005037 025062              CLR      $SAVR6       ;;WAIT LOOP FOR THE TTY
(1) 024774 005237 025062              '$: INC      $SAVR6    ;;WAIT FOR THE INC
(1) 025000 001375                      BNE      1$           ;;OF WORD
(3) 025002 012677 154132              MOV      (SP)+,@SWR   ;;POP STACK INTO @SWR
(3) 025006 012605                      MOV      (SP)+,R5     ;;POP STACK INTO R5
(3) 025010 012604                      MOV      (SP)+,R4     ;;POP STACK INTO R4
(3) 025012 012603                      MOV      (SP)+,R3     ;;POP STACK INTO R3
(3) 025014 012602                      MOV      (SP)+,R2     ;;POP STACK INTO R2
(3) 025016 012601                      MOV      (SP)+,R1     ;;POP STACK INTO R1
(3) 025020 012600                      MOV      (SP)+,R0     ;;POP STACK INTO R0
(1) 025022 012737 024704 000024      MOV      #SPWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1) 025030 012737 000340 000026      MOV      #340,@#PWRVEC+2 ;;PRIO:7
(1) 025036 104401                      TYPE      $POWER      ;;REPORT THE POWER FAILURE
(1) 025040 025064                      $PWRMG: .WORD $POWER   ;;POWER FAIL MESSAGE POINTER
(1) 025042 042766 000020 000002      BIC      #20,2(SP)    ;;CLEAR 'T' BIT
(1) 025050 005037 012014              CLR      $TBIT        ;;CLEAR THE 'T' BIT FLAG
(1) 025054 000002                      RTI
(1) 025056 000000                      $SILLUP: HALT          ;;THE POWER UP SEQUENCE WAS STARTED
(1) 025060 000776                      BR        .-2          ;;BEFORE THE POWER DOWN WAS COMPLETE
(1) 025062 000000                      $SAVR6: 0              ;;PUT THE SP HERE
(1) 025064 005015 047520 042527      $POWER: .ASCIIZ <15><12>'POWER'
(1) 025072 000122
(1)
4038 (1) .EVEN
(1) .SBTTL TRAP DECODER
(2)
(1)
(1)
(1)
(1)
(1)
(1)
(1) 025074 010046                      $TRAP: MOV      R0,-(SP) ;;SAVE R0
(1) 025076 016600 000002              MOV      2(SP),R0     ;;GET TRAP ADDRESS
(1) 025102 005740                      TST      -(R0)         ;;BACKUP BY 2
(1) 025104 111000                      MOVB     (R0),R0       ;;GET RIGHT BYTE OF TRAP
(1) 025106 006300                      A=L      R0           ;;POSITION FOR INDEXING
(1) 025110 016000 025130              MOV      $TRPAD(R0),R0 ;;INDEX TO TABLE
(1) 025114 000200                      RTS      R0           ;;GO TO ROUTINE

```

```

(1)
(1)
(1)
(1) 025116 011646
(1) 025120 016666 000004 000002
(1) 025126 000002
(1)
(3)
(3)
(3)
(3)
(3)
(3)
(3)
(3)
(3) 025130 025116
(3) 025132 023636
(3) 025134 024256
(3) 025136 024232
(3) 025140 024272
(3) 025142 024460
(1)
(1)
4039
4040 025144 003760 004352 004600 TSTADD: TST1,TST2,TST3
4041
4042 025152 005002 005142 005364
4043
4044 025160 005622 006506 007174
4045
4046 025166 007414 007576 010032
4047
4048 025174 010262 010756 011202
4049
4050 025202 011356 012036 012350
4051
4052 025210 012554 012634
4053
4054
4055
4056 025214 042523 020124 052126 STUPM: .ASCII /SET VT61S TO FULL DUPLEX, /<15><12>
      025222 030466 020123 047524
      025230 020040 052506 046114
      025236 042040 050125 042514
      025244 026130 006440 012
4057 025251 071 030066 041060
      025256 052501 026104 051040
      025264 046505 052117 026105
      025272 040520 044522 054524
      025300 046440 052101 044103
      025306 042105 052040 020117
      025314 047111 042524 043122
      025322 041501 006505 000012
4058
4059
4060
4061 025330 005015 042101 051104 DUNTST: .ASCII <15><12>/ADDRESSES WITH RESPONSIVE VT61S ARE:/<15><12>

```

;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO

\$TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
 RTI ;;RESTORE THE PSW

.SBTTL TRAP TABLE

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

ROUTINE

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

	025336	051505	042523	020123	
	025344	044527	044124	051040	
	025352	051505	047520	051516	
	025360	053111	020105	052126	
	025366	030466	020123	051101	
	025374	035105	005015	000	
4062	025401	116	020117	052126	NOVT: .ASCIZ /NO VT61 RESPONDED TO ESCZ SEQ. AUTO RETRY IN 30 SEC./<15><12>
	025406	030466	051040	051505	
	025414	047520	042116	042105	
	025422	052040	020117	051505	
	025430	055103	051440	050505	
	025436	020056	052501	047524	
	025444	051040	052105	054522	
	025452	044440	020116	030063	
	025460	051440	041505	006456	
	025466	000012			
4063					
4064					
4065	025470	005015	055104	030461	DLERR: .ASCIZ <15><12>/DZ11 FAILED AT ADDRESS/
	025476	043040	044501	042514	
	025504	020104	052101	040440	
	025512	042104	042522	051523	
	025520	000			
4066					
4067	025521	115	047101	040525	DMANA: .ASCII /MANUAL TEST SELECTED -/<15><12>
	025526	020114	042524	052123	
	025534	051440	046105	041505	
	025542	042524	020104	006455	
	025550	012			
4068	025551	105	052116	051105	.ASCIZ /ENTER ADDRESSES OF VT61S TO BE TESTED/<15><12>
	025556	040440	042104	042522	
	025564	051523	051505	047440	
	025572	020106	052126	030466	
	025600	020123	047524	041040	
	025606	020105	042524	052123	
	025614	042105	005015	000	
4069					
4070	025621	105	052116	051105	DMANB: .ASCIZ /ENTER TESTS TO BE RUN/<15><12>
	025626	052040	051505	051524	
	025634	052040	020117	042502	
	025642	051040	047125	005015	
	025650	000			
4071	025651	105	052116	051105	DMANL: .ASCIZ /ENTER LINES TO BE TESTED IN BINARY FORMAT(I.E.0=1,10=2000)/<15><12>
	025656	046040	047111	051505	
	025664	052040	020117	042502	
	025672	052040	051505	042524	
	025700	020104	047111	041040	
	025706	047111	051101	020131	
	025714	047506	046522	052101	
	025722	044450	042456	030056	
	025730	030475	030454	036460	
	025736	030062	030060	006451	
	025744	000012			
4072	025746	047105	042524	020122	BRATE: .ASCIZ /ENTER BAUD RATE IF DIFFERENT THAN DEFAULT (9600)/<15><12>
	025754	040502	042125	051040	
	025762	052101	020105	043111	

	025770	042040	043111	042506	
	025776	042522	052116	052040	
	026004	040510	020116	042504	
	026012	040506	046125	020124	
	026020	034450	030066	024460	
	026026	005015	000		
4073	026031	077	005015	000	QUES: .ASCIZ /?/<15><12>
4074					
4075	026035	101	020116	051505	EM1: .ASCIZ /AN ESC SEQ. TO THE VT61 FAILED - OCTAL EQUIV. IS:/<15><12>
	026042	020103	042523	027121	
	026050	052040	020117	044124	
	026056	020105	052126	030466	
	026064	020040	040506	046111	
	026072	042105	026440	047440	
	026100	052103	046101	042440	
	026106	052521	053111	020056	
	026114	051511	006472	000012	
4076	026122	042524	052123	020043	DH1: .ASCIZ /TEST# ERR PC BYTE 1+2 BYTE 3+4/<15><12>
	026130	042440	051122	050040	
	026136	020103	041040	052131	
	026144	020105	025461	020062	
	026152	054502	042524	031440	
	026160	032053	005015	000	
4077					
4078	026165	122	041505	044505	EM2: .ASCIZ /RECEIVE STATUS ERROR./<15><12>
	026172	042526	051440	040524	
	026200	052524	020123	051105	
	026206	047522	027122	005015	
	026214	000			
4079	026215	101	042104	020056	DH2: .ASCIZ /ADD. STAT. ERR.BITS CHAR./<15><12>
	026222	051440	040524	027124	
	026230	020040	051105	027122	
	026236	044502	051524	020040	
	026244	044103	051101	006456	
	026252	000012			
4080					
4081	026254	047523	052106	040527	EM3: .ASCIZ /SOFTWARE (VSTAT) STATUS ERROR./<15><12>
	026262	042522	024040	051526	
	026270	040524	024524	051440	
	026276	040524	052524	020123	
	026304	051105	047522	027122	
	026312	005015	000		
4082	026315	040	040520	051523	DH3: .ASCIZ / PASS#, TEST#, EXP.STAT, ACT.STAT/<15><12>
	026322	026043	020040	042524	
	026330	052123	026043	020040	
	026336	054105	027120	052123	
	026344	052101	020054	040440	
	026352	052103	051456	040524	
	026360	006524	000012		
4083					
4084	026364	042107	020056	040504	EM4: .ASCIZ /GD. DATA DOES NOT MATCH REC. DATA/<15><12>
	026372	040524	042040	042517	
	026400	020123	047516	020124	
	026406	040515	041524	020110	
	026414	042522	027103	042040	
	026422	052101	006501	000012	

4085	026430	042524	052123	020043	DH4:	.ASCIZ /TEST# .REC.CNT.,GD. DATA, REC. DATA/<15><12>
	026436	051054	041505	041456		
	026444	052116	026056	042107		
	026452	020056	040504	040524		
	026460	020054	042522	027103		
	026466	042040	052101	006501		
	026474	000012				
4086						.EVEN
4087						
4088	026476	054502	042524	020123	EM5:	.ASCIZ /BYTES EXPECTED DOES NOT EQUAL BYTES RECEIVED/<15><12>
	026504	054105	042520	052103		
	026512	042105	042040	042517		
	026520	020123	047516	020124		
	026526	050505	040525	020114		
	026534	054502	042524	020123		
	026542	042522	042503	053111		
	026550	042105	005015	000		
4089	026555	102	052131	051505	DH5:	.ASCIZ /BYTES EXP., BYTES REC./<15><12>
	026562	042440	050130	026056		
	026570	041040	052131	051505		
	026576	051040	041505	006456		
	026604	000012				
4090						
4091	026606	052503	051522	051117	EM6:	.ASCIZ /CURSOR POSITIONING ERROR/<15><12>
	026614	050040	051517	052111		
	026622	047511	044516	043516		
	026630	042440	051122	051117		
	026636	005015	000			
4092	026641	107	020104	044514	DH6:	.ASCIZ /GD LINE GD COL. BD LINE BD COL/<15><12>
	026646	042516	020040	042107		
	026654	041440	046117	020056		
	026662	020040	042102	046040		
	026670	047111	020105	041040		
	026676	020104	047503	006514		
	026704	000012				
4093						
4094	026706	044504	042522	052103	EM7:	.ASCIZ /DIRECT CURSOR ADDRESSING FAILURE/<15><12>
	026714	041440	051125	047523		
	026722	020122	042101	051104		
	026730	051505	044523	043516		
	026736	043040	044501	052514		
	026744	042522	005015	000		
4095	026751	120	051501	021523	DH7:	.ASCIZ /PASS# TEST# ERROR PC /<15><12>
	026756	020040	042524	052123		
	026764	021440	020040	051105		
	026772	047522	020122	041520		
	027000	020040	006440	000012		
4096	027006	040520	051523	020043	DH10:	.ASCIZ /PASS# TEST# BD.ROW BD.COL/<15><12>
	027014	052040	051505	021524		
	027022	020040	042102	051056		
	027030	053517	020040	042102		
	027036	041456	046117	005015		
	027044	000				
4097						
4098	027045	114	051501	020124	EM11:	.ASCIZ /LAST TRANSMISSION TO VT61 CAUSED UNIT TO FAIL-HANG./<15><12>
	027052	051124	047101	046523		

	027060	051511	044523	047117	
	027066	052040	020117	052126	
	027074	030466	041440	052501	
	027102	042523	020104	047125	
	027110	052111	052040	020117	
	027116	040506	046111	044055	
	027124	047101	027107	005015	
	027132	000			
4099					
4100	027133	126	033124	020061	EM12: .ASCIZ /VT61 UNDER TEST FAILED- ERROR DATA FOLLOWS/<15><12>
	027140	047125	042504	020122	
	027146	042524	052123	043040	
	027154	044501	042514	026504	
	027162	042440	051122	051117	
	027170	042040	052101	020101	
	027176	047506	046114	053517	
	027204	006523	000012		
4101					
4102	027210	052126	030466	043040	EM10: .ASCIZ /VT61 FAILED SELF TEST FUNCTION/<15><12>
	027216	044501	042514	020104	
	027224	042523	043114	052040	
	027232	051505	020124	052506	
	027240	041516	044524	047117	
	027246	005015	000		
4103					
4104					
4105	027251	120	051501	021523	DH12: .ASCIZ /PASS#, TEST#, GD.CKSUM, BD.CKSUM/<15><12>
	027256	020054	052040	051505	
	027264	021524	020054	042107	
	027272	041456	051513	046525	
	027300	020054	042102	041456	
	027306	051513	046525	005015	
	027314	000			
4106					
4107	027315	124	051505	044524	DABRT: .ASCIZ /TESTING ABORTED-TOO MANY FATAL XMITs/<15><12>
	027322	043516	040440	047502	
	027330	052122	042105	052055	
	027336	047517	046440	047101	
	027344	020131	040506	040524	
	027352	020114	046530	052111	
	027360	006523	000012		
4108					
4109	027364	052126	030466	051040	EM13: .ASCIZ /VT61 RECEIVER CHECKSUM COMPARE ERROR/<15><12>
	027372	041505	044505	042526	
	027400	020122	044103	041505	
	027406	051513	046525	041440	
	027414	046517	040520	042522	
	027422	042440	051122	051117	
	027430	005015	000		
4110					
4111	027433	126	033124	020061	EM14: .ASCIZ /VT61 TRANSMITTER CHECKSUM COMPARE ERROR/<15><12>
	027440	051124	047101	046523	
	027446	052111	042524	020122	
	027454	044103	041505	051513	
	027462	046525	041440	046517	
	027470	040520	042522	042440	

	027476	051122	051117	005015	
	027504	000			
4112					
4113		027506			
4114	027506	047125	052111	052440	DVUNIT: .EVEN .ASCII /UNIT UNDER TEST /<15><12>
	027514	042116	051105	052040	
	027522	051505	020124	005015	
4115	027530	041522	051123	020040	.ASCIIZ /RCSR VECT. LINE IDENT/<15><12>
	027536	053040	041505	027124	
	027544	020040	046040	047111	
	027552	020105	044440	042504	
	027560	052116	005015	000	
4116	027565	040	041522	051123	DH11: .ASCIIZ / RCSR VECT./<15><12>
	027572	020040	053040	041505	
	027600	027124	005015	000	
4117	027605	120	044522	052116	DPRTR: .ASCIIZ /PRINTER IS ATTACHED/<15><12>
	027612	051105	044440	020123	
	027620	052101	040524	044103	
	027626	042105	005015	000	
4118	027633	103	050117	042511	DCOPYR: .ASCIIZ /COPIER IS ATTACHED/<15><12>
	027640	020122	051511	040440	
	027646	052124	041501	042510	
	027654	006504	000012		
4119	027660	047530	043106	052040	EM15: .ASCIIZ /XOFF TO VT61 FAILED TO HALT BLOCK XMIT/<15><12>
	027666	020117	052126	030466	
	027674	043040	044501	042514	
	027702	020104	047524	044040	
	027710	046101	020124	046102	
	027716	041517	020113	046530	
	027724	052111	005015	000	
4120	027731	130	047117	052040	EM16: .ASCIIZ /XON TO VT61 FAILED TO RESTART BLOCK XMIT/<15><12>
	027736	020117	052126	030466	
	027744	043040	044501	042514	
	027752	020104	047524	051040	
	027760	051505	040524	052122	
	027766	041040	047514	045503	
	027774	054040	044515	006524	
	030002	000012			
4121	030004	047516	054040	047117	EM17: .ASCIIZ /NO XON RECEIVED WITHIN 3 SEC. AFTER A RESET/<15><12>
	030012	051040	041505	044505	
	030020	042526	020104	044527	
	030026	044124	047111	031440	
	030034	051440	041505	020056	
	030042	043101	042524	020122	
	030050	020101	042522	042523	
	030056	006524	000012		
4122	030062	040514	052123	050040	EM20: .ASCIIZ /LAST PERIPHERAL OPERATION ABORTED/<15><12>
	030070	051105	050111	042510	
	030076	040522	020114	050117	
	030104	051105	052101	047511	
	030112	020116	041101	051117	
	030120	042524	006504	000012	
4123	030126	047503	046125	020104	EM21: .ASCIIZ /COULD NOT CLEAR LAST ABORT FLAG./<15><12>
	030134	047516	020124	046103	
	030142	040505	020122	040514	
	030150	052123	040440	047502	

	030156	052122	043040	040514	
	030164	027107	005015	000	
4124	030171	123	046517	047440	EM22: .ASCIIZ /SOM OR EOM NOT RECEIVED DURING MAINT. MODE TRANSMIT/<15><12>
	030176	020122	047505	020115	
	030204	047516	020124	042522	
	030212	042503	053111	042105	
	030220	042040	051125	047111	
	030226	020107	040515	047111	
	030234	027124	046440	042117	
	030242	020105	051124	047101	
	030250	046523	052111	005015	
	030256	000			
4125	030257	114	047111	020105	EM23: .ASCIIZ /LINE FEED OR CURSOR RIGHT ISSUED FROM ROW 23 DID NOT CAUSE SCREEN TO SC
	030264	042506	042105	047440	
	030272	020122	052503	051522	
	030300	051117	051040	043511	
	030306	052110	044440	051523	
	030314	042525	020104	051106	
	030322	046517	051040	053517	
	030330	031040	020063	044504	
	030336	020104	047516	020124	
	030344	040503	051525	020105	
	030352	041523	042522	047105	
	030360	052040	020117	041523	
	030366	047522	046114	005015	
	030374	000			
4126	030375	120	051501	020123	DH13: .ASCIIZ /PASS , TEST , VSTAT/<15><12>
	030402	020054	020040	042524	
	030410	052123	026040	020040	
	030416	053040	052123	052101	
	030424	005015	000		
4127	030427	120	051501	026123	DH14: .ASCIIZ /PASS, TEST, ERR PC, VSTAT/<15><12>
	030434	020040	052040	051505	
	030442	026124	020040	042440	
	030450	051122	050040	026103	
	030456	020040	053040	052123	
	030464	052101	005015	000	
4128					
4129	030471	105	041523	000040	DESC: .ASCIIZ /ESC /
4130					
4131					
4132					
4133	030476	042513	041131	040517	DKYBD: .ASCII /KEYBOARD TEST/<15><12>
	030504	042122	052040	051505	
	030512	006524	012		
4134	030515	113	054505	052123	.ASCII /KEYSTROKES ECHO:/<15><12>
	030522	047522	042513	020123	
	030530	041505	047510	006472	
	030536	012			
4135	030537	101	042040	051511	.ASCII /A DISPLAY CHAR. = A DISPLAY CHAR./<15><12>
	030544	046120	054501	041440	
	030552	040510	027122	036440	
	030560	040440	042040	051511	
	030566	046120	054501	041440	
	030574	040510	027122	005015	
4136	030602	031463	036440	042440	.ASCII /33 = ESC/<15><12>

4137	030610	041523	005015		
	030614	032461	036440	041440	.ASCII /15 = C-R/<15><12>
	030622	051055	005015		
4138	030626	031061	036440	046040	.ASCII /12 = L-F/<15><12>
	030634	043055	005015		
4139	030640	033460	036440	041040	.ASCII /07 = BELL/<15><12>
	030646	046105	006514	012	
4140	030653	061	020060	020075	.ASCII /10 = TAB/<15><12>
	030660	040524	006502	012	
4141	030665	116	047117	042055	.ASCIIZ /NON-DISPLAY CHAR.= OCTAL EQUIV/<15><12>
	030672	051511	046120	054501	
	030700	041440	040510	027122	
	030706	020075	041517	040524	
	030714	020114	050505	044525	
	030722	006526	000012		
4142					
4143	030726	040524	020102	000	DTAB: .ASCIIZ /TAB /
4144	030733	103	051055	000040	DCR: .ASCIIZ /C-R /
4145	030740	026514	020106	000	DLF: .ASCIIZ /L-F /
4146	030745	102	046105	020114	DBELL: .ASCIIZ /BELL /
	030752	000			
4147					
4148	030753	114	047517	020120	DLOOP: .ASCII /LOOP TEST - LOOP COMMANDS AND DATA THRU/<15><12>
	030760	042524	052123	026440	
	030766	046040	047517	020120	
	030774	047503	046515	047101	
	031002	051504	040440	042116	
	031010	042040	052101	020101	
	031016	044124	052522	005015	
4149	031024	047510	052123	041040	.ASCII /HOST BACK TO VT61 UNDER TEST. /<15><12>
	031032	041501	020113	047524	
	031040	053040	033124	020061	
	031046	047125	042504	020122	
	031054	042524	052123	020056	
	031062	005015			
4150	031064	047503	052116	047522	DNTZ: .ASCIIZ /CONTROL C EXITS TEST./<15><12>
	031072	020114	020103	042440	
	031100	044530	051524	052040	
	031106	051505	027124	005015	
	031114	000			
4151					
4152	031115	105	044530	020124	DEXT: .ASCIIZ /EXIT TEST./
	031122	042524	052123	000056	
4153					
4154	031130	051120	047111	042524	DPRNT: .ASCII /PRINTER TEST -/<15><12>
	031136	020122	042524	052123	
	031144	026440	005015		
4155	031150	031461	020062	047503	.ASCII /132 COLUMNS OF A SLIDING PATTERN WILL BE/
	031156	052514	047115	020123	
	031164	043117	040440	051440	
	031172	044514	044504	043516	
	031200	050040	052101	042524	
	031206	047122	053440	046111	
	031214	020114	042502		
4156	031220	047503	052116	047111	.ASCII /CONTINUOUSLY OUTPUTTED TO PRINTER/<15><12>
	031226	052517	046123	020131	

	031234	052517	050124	052125	
	031242	042524	020104	047524	
	031250	050040	044522	052116	
	031256	051105	005015		
4157	031262	040503	027122	051040	DCRST: .ASCIZ /CAR. RET. TO START/<15><12>
	031270	052105	020056	047524	
	031276	051440	040524	052122	
	031304	005015	000		
4158					
4159	031307	114	051501	020124	DEVERR: .ASCIZ /LAST XMIT CAUSED VT61 HANG/<15><12>
	031314	046530	052111	041440	
	031322	052501	042523	020104	
	031330	052126	030466	044040	
	031336	047101	006507	000012	
4160	031344	000077			QMRK: .ASCIZ /?/
4161	031346	051120	042117	041525	DKBD: .ASCII /PRODUCTION KEYBOARD TEST. 10 ERRORS CAUSES TEST EXIT./<15><12>
	031354	044524	047117	045440	
	031362	054505	047502	051101	
	031370	020104	042524	052123	
	031376	020056	030061	042440	
	031404	051122	051117	020123	
	031412	040503	051525	051505	
	031420	052040	051505	020124	
	031426	054105	052111	006456	
	031434	012			
4162	031435	104	050105	042522	.ASCIZ /DEPRESS KEYS FROM LEFT TO RIGHT/<15><12>
	031442	051523	045440	054505	
	031450	020123	051106	046517	
	031456	046040	043105	020124	
	031464	047524	051040	043511	
	031472	052110	005015	000	
4163	031477	104	050105	042522	DLSHFT: .ASCIZ /DEPRESS LEFT SHIFT KEY AND THE 'A' KEY /<15><12>
	031504	051523	046040	043105	
	031512	020124	044123	043111	
	031520	020124	042513	020131	
	031526	047101	020104	044124	
	031534	020105	040442	020042	
	031542	042513	020131	005015	
	031550	000			
4164	031551	104	050105	042522	DTOP: .ASCIZ /DEPRESS KEYS IN TOP ROW/<15><12>
	031556	051523	045440	054505	
	031564	020123	047111	052040	
	031572	050117	051040	053517	
	031600	005015	000		
4165					
4166	031603	104	050105	042522	DRSHFT: .ASCIZ /DEPRESS RIGHT SHIFT KEY AND THE 'A' KEY /<15><12>
	031610	051523	051040	043511	
	031616	052110	051440	044510	
	031624	052106	045440	054505	
	031632	040440	042116	052040	
	031640	042510	021040	021101	
	031646	045440	054505	006440	
	031654	000012			
4167	031656	042504	051120	051505	DSEC: .ASCIZ /DEPRESS KEYS IN SECOND ROW/<15><12>
	031664	020123	042513	051531	
	031672	044440	020116	042523	

	031700	047503	042116	051040	
	031706	053517	005015	000	
4168					
4169	031713	104	050105	042522	DTHRD: .ASCIZ /DEPRESS KEYS IN THIRD ROW BEGINNING WITH 'A' /<15><12>
	031720	051523	045440	054505	
	031726	020123	047111	052040	
	031734	044510	042122	051040	
	031742	053517	041040	043505	
	031750	047111	044516	043516	
	031756	053440	052111	020110	
	031764	040447	006447	000012	
4170	031772	042504	051120	051505	DCONT: .ASCIZ /DEPRESS CONTROL KEY ,AND THE 'A' KEY /<15><12>
	032000	020123	047503	052116	
	032006	047522	020114	042513	
	032014	020131	040454	042116	
	032022	052040	042510	021040	
	032030	021101	045440	054505	
	032036	006440	000012		
4171	032042	042504	051120	051505	DBOT: .ASCIZ /DEPRESS KEYS IN FORTH ROW EXCEPT SHIFT KEYS/<15><12>
	032050	020123	042513	051531	
	032056	044440	020116	047506	
	032064	052122	020110	047522	
	032072	020127	054105	042503	
	032100	052120	051440	044510	
	032106	052106	045440	054505	
	032114	006523	000012		
4172	032120	042504	051120	051505	DSPCE: .ASCIZ /DEPRESS SPACE BAR/<15><12>
	032126	020123	050123	041501	
	032134	020105	040502	006522	
	032142	000012			
4173					
4174	032144	042504	051120	051505	DKPD: .ASCIZ /DEPRESS KEYPAD KEYS,LEFT TO RIGHT, TOP TO BOTTOM/<15><12>
	032152	020123	042513	050131	
	032160	042101	045440	054505	
	032166	026123	042514	052106	
	032174	052040	020117	044522	
	032202	044107	026124	052040	
	032210	050117	052040	020117	
	032216	047502	052124	046517	
	032224	005015	000		
4175					
4176	032227	113	054505	047502	DKBERR: .ASCII /KEYBOARD ERROR,KEY POSITION IN ROW SHOULD BE /
	032234	051101	020104	051105	
	032242	047522	026122	042513	
	032250	020131	047520	044523	
	032256	044524	047117	044440	
	032264	020116	047522	020127	
	032272	044123	052517	042114	
	032300	041040	020105		
4177	032304	020040	005015		KYSTRK: .ASCII / /<15><12>
4178	032310	041517	040524	020114	.ASCIZ /OCTAL GD, OCTAL BAD/<15><12>
	032316	042107	020054	041517	
	032324	040524	020114	040502	
	032332	006504	000012		
4179	032336	020040	020040	020040	DSPC6: .ASCIZ / /
	032344	000			

4180						
4181	032345	036	076	020	ROW1:	.BYTE 36,76,20,13,32,12,54,44,14,41,71,57,63,64,3,114,0
	032350	013	032	012		
	032353	054	044	014		
	032356	041	071	057		
	032361	063	064	003		
	032364	114	000			
4182						
4183	032366	026	056	030	ROW2:	.BYTE 26,56,30,73,52,22,55,34,24,31,51,77,62,61,2,0
	032371	073	052	022		
	032374	055	034	024		
	032377	031	051	077		
	032402	062	061	002		
	032405	000				
4184						
4185	032406	046	040	053	ROW3:	.BYTE 46,40,53,23,72,42,45,74,11,21,47,27,66,0
	032411	023	072	042		
	032414	045	074	011		
	032417	021	047	027		
	032422	066	000			
4186						
4187	032424	016	070	060	ROW4:	.BYTE 16,70,60,50,33,43,25,35,75,65,37,115,67,0
	032427	050	033	043		
	032432	025	035	075		
	032435	065	037	115		
	032440	067	000			
4188						
4189	032442	001	000		CNTRA:	.BYTE 01,0
4190						
4191	032444	101	000		SHFIA:	.BYTE 101,0
4192						
4193	032446	015	000		SPCB:	.BYTE 15,0
4194						
4195	032450	113	004	103	KYPD:	.BYTE 113,04,103,104,1,112,101,102,6,7,106,100,5
	032453	104	001	112		
	032456	101	102	006		
	032461	007	106	100		
	032464	005				
4196	032465	010	105	107		.BYTE 10,105,107,110,17,111,0
	032470	110	017	111		
	032473	000				
4197						
4198						
4199	032474	000500			RCRLB:	.EVEN .BLKB 500 ;RECEIVE CIRCULAR BUFFER
4200						
4201	033174	000500			TCRLB:	.BLKB 500 ;TRANSMIT CIRCULAR BUFFER
4202	033674	000000			ABUFP:	.WORD 0
4203	033676	000062			ABBUF:	.BLKB 50.
4204	033760	000000				0
4205		000001				.END

ABBUF	033676	BLKM	002422	CRT	002160	DKBERR	032227	EM17	030004
ABSXT	023030	BOROW	022446	CUHME	002350	DKPD	032144	EM2	026165
ABUFP	033674	BPTVEC=	000014	CUP	002166	DKYBD	030476	EM20	030062
AESCO	017036	BRATE	025746	CURER	020562	DLAY	002414	EM21	030126
AESCP	017030	BUBCT	002404	CURPOS=	000004	DLERR	025470	EM22	030171
ALEXT	013566	BUMPCT	022750	CURS1	004616	DLF	030740	EM23	030257
ALWCNT	002376	BYSTOR	020370	CURS1A	007212	DLOOP	030753	EM3	026254
AOUT	014324	CALCK	022120	CURS1B	007432	DLSHFT	031477	EM4	026364
ASESC	016652	CARRT	002146	CURS2	010300	DMAIN	002176	EM5	026476
ASTRT	003760	CDEV	013704	CUR1XT	007412	DMANA	025521	EM6	026606
AUTO	002426	CDEVA	013756	C2CK	010712	DMANB	025621	EM7	026706
AUTOA	002450	CDWN	002162	C2XT	010754	DMANL	025651	ENDPS	011636
AUTOB	002454	CESAM	017022	DABRT	027315	DMPDCT	022470	ENDSEL	011620
BASC3	005160	CHKITT	020332	DAPNT	002224	DOAROW	012676	ENDTAB	002110
BBLUP	015412	CHOM	002156	DATSC	021470	DPNT	002312	ENSRT	010024
BDEXT	013556	CHRD	002356	DBELL	030745	DPRNT	031130	EOM =	000004
BDEXTA	013554	CKABRT	022174	DBOT	032042	DPRTR	027605	EOS	002170
BDRATE	002134	CKDAT	006230	DCNTZ	031064	DRECT	002204	EPL =	001000
BEL	002144	CKEOM	004506	DCONT	031772	DRSHFT	031603	EPNT	002310
BINOCT	021632	CKEXT	007170	DCOPYR	027633	DSEC	031656	ERLNE	016622
BIT0 =	000001	CKGP =	000106	DCOUNT	021440	DSMES	021536	ERPL	002230
BIT00 =	000001	CKKBD	022322	DCR	030733	DSPCE	032120	ERRVEC=	000004
BIT01 =	000002	CKLIN	005020	DCRAD	002234	DSPC6	032336	ERSE	011220
BIT02 =	000004	CKMEM	006270	DCRST	031262	DSWR =	177570	ERSXT	011354
BIT03 =	000010	CKMNT	004370	DDISP =	177570	DTAB	030726	ESAMB	017132
BIT04 =	000020	CKOFF	023170	DELAY	021376	DTHRD	031713	ESC =	000400
BIT05 =	000040	CKSCRA	010050	DELIM	004160	DTOP	031551	ESCN	002326
BIT06 =	000100	CKSCRB	010146	DEMP	002214	DTTBL	013022	ESCO	002250
BIT07 =	000200	CKSFT	017732	DESC	030471	DT0	001424	ESCOI =	002250
BIT08 =	000400	CKSRC	007036	DEVERR	031307	DT1	001436	ESCP	002314
BIT09 =	001000	CKSTR	020300	DEXT	031115	DT2	001454	ESCP1 =	002314
BIT1	000002	CKSUM =	002000	DF0	001444	DT3	001466	ESCYI =	002234
BIT10 =	002000	CKSUMA	006524	DF1	001450	DT4	001502	ESCZ	002322
BIT11 =	004000	CKSUMB	006736	DF2	001452	DT5	001514	ESCZ1 =	002322
BIT12 =	010000	CKVST	020344	DF3	001476	DT6	001526	ESSEQ	002412
BIT13 =	020000	CLFT	002164	DF4	001536	DUNTST	025330	ESTEX	004340
BIT14 =	040000	CLMAIN	012754	DF5	001541	DVUNIT	027506	ESTST	003776
BIT15 =	100000	CLRCK	002206	DF6	001545	DZAPT	002100	EXINT	015066
BIT2	000004	CLREG	020074	DH1	026122	DZLNE	001632	EXIT3	005360
BIT3 =	000010	CLTCK	022210	DH10	027006	DZTBL	001562	EXMAIN	012344
BIT4 =	000020	CMPOS	020660	DH11	027565	EAPNT	002222	EXMNT	004576
BIT5 =	000040	CNTF	022566	DH12	027251	EEMP	002212	EXTST	023020
BIT6 =	000100	CNTRA	032442	DH13	030375	EINST	002226	FADD	013650
BIT7 =	000200	COMGP =	000020	DH14	030427	EMAIN	002174	FADD1	013666
BIT8 =	000400	CONRD	015640	DH2	026215	EMTVEC=	000030	FEXIT	012740
BIT9 =	001000	CONVLN	021730	DH3	026315	EM1	026035	FNDBT	023154
BLDADA	002540	CPABRT=	000171	DH4	026430	EM10	027210	FRCECT	022766
BLDADD	002536	CPPOS	020722	DH5	026555	EM11	027045	FTEX1	004344
BLDINA	021446	CR	000015	DH6	026641	EM12	027133	FTLCNT	002374
BLDINC	021442	CRBUF	002132	DH7	026751	EM13	027364	FTLEXT	020434
BLDLNA	002660	CRCSR	002130	DISPLA	001142	EM14	027433	GBDRT	003032
BLDLNE	002650	CRLF	000200	DISPRE	000174	FM15	027660	GCMD	004034
BLDTST	002724	CRUL	021314	DKBD	031346	FM16	027731	GDCURP	007402

GDSCRL 010260	MANS 002476	PRO = 000000	R00C11 002342	SW14 = 040000
GDSTRK 022560	MANS 002516	PR1 = 000040	R00C20 002344	SW15 = 100000
GETON 017622	MAXST= 000024	PR2 = 000100	R00C80 002352	SW2 = 000004
GOTON 017670	MCLR - 000020	PR3 = 000140	R01C00 002330	SW3 = 000010
GTCR 021766	MEMA 005654	PR4 = 000200	R01C20 002332	SW4 = 000020
GTEXT 022100	MEMB 005722	PR5 = 000240	R12C00 002336	SW5 = 000040
GTNUM 022010	MEMC 006110	PR6 = 000300	R22C00 002334	SW6 = 000100
HDFLG 021312	MEMD 006160	PR7 = 000340	R23C00 002340	SW7 = 000200
HT = 000011	MEMXT 006504	PS = 177776	R23C78 002354	SW8 = 000400
IABT 002216	MEM1 005640	PSW = 177776	R23C79 002236	SW9 = 001000
IDENT 002320	MNEPT 002104	PUSH2S= 024646	R6 = %000006	TAB 002152
ILLNE = 000010	MODCA 003234	PWRVEC= 000024	R7 = %000007	TBBUF 017460
INAG 007660	MODCK 003102	QMRK 031344	SCAN = 000040	TBITVE= 000014
INCLCT 006500	MODCO 003132	QUES 026031	SCL = 000070	TBLCK 013700
INCSCH 006502	MODE 002372	QUST 022102	SCRTH 002072	TBUF 017464
INITA 014416	MODEM 002074	RABT 002302	SEQ4 004246	TCOMB = 040010
INRPL 007614	MONIT 015672	RBBUF 017124	SETA 013046	TCRLB 033174
INRXT 010030	MPATT = 006460	RBUF 017130	SETREV 006474	TCUCH 002260
INTAB 001572	MSTBL 012774	RCRLB 032474	SHFTA 032444	TDATA 010020
INTRC 016176	MSTRT 000204	PCUR 002242	SLSH = 000057	TDEV 014052
INTXM 017164	MZLNE 001752	RDCUR 002244	SOM = 000002	TEBUF 017462
INTXT 015014	NABRT = 000170	REBUF 017126	SPCB 032446	TESC 015464
INXMT 004062	NCKGP = 000107	RECAD 016064	SPTN 004240	TFUNCT 010012
IOTVEC= 000020	NOCALC 022164	RECDN = 000200	STACK = 001100	TIMEXT 023156
ITSUMA 007160	NOER 020050	RECEX 015400	START 000200	TKVEC = 000060
JMPADD 016060	NOKIL 017202	RECEXA 015404	STKMT= 177774	TMNAD 013532
KBDLP 022332	NOROUT 023070	RECITT 020364	STRBYT 020322	TOADD 002402
KIENA 017450	NORXT 020440	RECTM 015212	STRO 017160	TOFF 016110
KYBD 012054	NOSHFT 021724	RECTX 016644	STRP 017162	TOTCH = 003600
KYBXMT 012330	NOSOM 017230	REEX 016124	STRTAB 002106	TOTC1 = 003601
KYPD 032450	NOUNIT 015042	RENA = 000100	STTEP 017156	TPENT 013674
KYSTRK 032304	NOVT 025401	REQM - 020000	STTER 017134	TPREG 002406
KYSTRT 012116	NULL 002142	RESET 002324	STUPM 025214	TPRNT 012366
LDADD 015172	NWLN 010774	RESETV 017470	SVER1 002360	TPVEC = 000064
LDBUF 022166	OCTBIN 021706	RESPTR 020502	SVER2 002362	TRAPVE= 000034
LDOUT 020466	OCTLNE 021226	RESVEC= 000010	SWR 001140	TRDY = 100000
LDPOS 022454	OFFLP 005472	REVID - 000040	SWREG 000176	TRMID = 000002
LDXMIT 020442	ONE 002400	ROW1 032345	SW0 = 000001	TRPA 013452
LF - 000012	ONLP 005550	ROW2 032366	SW00 = 000001	TRPB 013476
LINXT 005140	ONOF 005532	ROW3 032406	SW01 = 000002	TRPE 013654
LKKB 002200	ONOF 005442	ROW4 032424	SW02 = 000004	TRPVEC 013576
LNECK 014216	ONOF 005620	RRDY = 100000	SW03 = 000010	TRTVEC= 000014
LNEPT 002102	ONOF 005402	RSCN - 000040	SW04 = 000020	TSMAD 016160
LNFE 002150	PABRT = 010000	RSMAIN 012770	SW05 = 000040	TSTADD 025144
LNFW 020764	PATGN 006360	RSOM = 040000	SW06 = 000100	TSTER 002316
LOOP 022570	PDKBD 012652	RSTER 016606	SW07 = 000200	TSTLNE 002124
LOOPR 022654	PIRQ - 177772	RSTT = 004000	SW08 = 000400	TSTNM 002424
LOOP 022622	PIRQVE - 000240	RTRP 013506	SW09 = 001000	TSTPTR 002370
LOOPTA 022626	PMULT 021436	RXE - 010000	SW1 = 000002	TST1 003760
LPSTR 022674	POPT 004150	RXOFF - 100000	SW10 = 002000	TST10 006506
LPTST 012572	POP2SP- 022626	ROSVE 002416	SW11 = 004000	TST11 007174
LSTST 011374	PRABRT 000172	ROSV1 017730	SW12 = 010000	TST12 007414
MAINT 000010	PRESC 002410	R00C08 002346	SW13 = 020000	TST13 007576

TST14	010032	XMAD2	020262	\$ERROR	023470	\$TIMES	001160	.DEMP =	000152
TST15	010262	XM CNT	017466	\$ERRPC	001116	\$TKB	001146	.DISAC=	000137
TST16	010756	XMDNE =	000001	\$ERRTB	001174	\$TKS	001144	.DMAIN=	000141
TST17	011202	XMITT	020152	\$ERRTY	024056	\$TN =	000025	.DPNT =	000130
TST2	004352	XMIT1	020372	\$ERTTL	001112	\$TPB	001152	.DIRECT=	000103
TST20	011356	XMKIL -	000200	\$ESCAP	001162	\$TPFLG	001157	.EAPNT=	000131
TST21	012036	XMREC	020116	\$FILLC	001156	\$TPS	001150	.EEMP =	000112
TST22	012350	XMTAL	002252	\$FILLS	001155	\$TRAP	025074	.EINST=	000111
TST23	012554	XMWT	023010	\$GDADR	001120	\$TRAP2	025116	.EMAIN=	000101
TST24	012634	XMXT	020362	\$GDDAT	001124	\$TRP =	000006	.ENAC =	000136
TST3	004600	XMZER	023072	\$GET42	011714	\$TRPAD	025130	.EOS =	000112
TST4	005002	XOFF =	000023	\$HD =	000000	\$TSTNM	001102	.EPNT =	000127
TST5	005142	XON =	000021	\$ICNT	001104	\$TYPDS	024460	.ERPL =	000151
TST6	005364	XRCMP	020766	\$ILLUP	025056	\$TYPE	023636	.ESC =	000033
TST7	005622	XREC	021070	\$INTAG	001135	\$TYPEC	024006	.ESCZ =	055033
TXEX	017456	XRERR	021154	\$ITEMB	001114	\$TYPEX	024054	.IABT =	000137
TXRCK	002266	XREXT	021172	\$LF	001172	\$TYPOC	024256	.LKKB =	000105
TXSUM =	000100	XROUT	021272	\$LOOP	012010	\$TYPON	024272	.LNFEDE	000012
TXTCK	002274	XSCN	000040	\$LPADR	001106	\$TYPOS	024232	.O =	000117
TYPDS =	104405	Y	002140	\$LPERR	001110	\$XTSTR	023230	.P =	000120
TYPE -	104401	ZERO	002364	\$MXCNT	023466	\$GET4=	000001	.PSCRN=	000150
TYPNUM	022116	ZERST	004024	\$NULL	001154	\$OFILL	024455	.RABT =	000140
TYPOC =	104402	ZFLAG	017672	\$NWTST=	000000	.	033762	.RDCUR=	000131
TYPON =	104404	\$AUTOB	001134	\$OCNT	024454	.BEL =	000007	.RESET=	000122
TYPOS =	104403	\$BDADR	001122	\$OMODE	024456	.CARRT=	000015	.R00 =	000040
TYP6	002366	\$BDDAT	001226	\$OVER	023452	.CDWN =	000102	.R00C1=	025440
T1ERR	004262	\$BELL	001164	\$PASS	001100	.CHOM =	000110	.R00C2=	032040
UNLKKB	002202	\$CHARC	024052	\$POWER	025064	.CLFT =	000104	.R01 -	000041
UPD4	016126	\$CLR.T	011732	\$PWDN	024704	.CLRCK=	000133	.R12 =	000054
VBIT	023164	\$CMTAG	001100	\$PWDMG	025040	.CLTCK=	000134	.R22 =	000066
VDLAY	023166	\$CM3 -	000000	\$PWUP	024756	.CPYSC=	000135	.R23 =	000067
VECT	002076	\$CRLF	001171	\$QUES	001170	.CRT =	000103	.R23C0=	020067
VECT	002122	\$DBLK	024674	\$RTNAD	012012	.CUP =	000101	.R23C7=	067467
VRBUF	002114	\$DOAGN	011752	\$RTRN	012006	.C00 =	000040	.TAB =	000011
VSTAT	002420	\$DTBL	024664	\$SAVR6	025062	.C03 -	000043	.TCUCH=	000127
VVECT	001552	\$ENDAD	011742	\$SCOPE	023214	.C08	000050	.TSTER=	000124
VXBUF	002120	\$ENDCT	011672	\$SETUP-	000037	.C11 =	000053	.TXRCK=	000135
VXTCR	002116	\$ENDMG	012021	\$STUP =	177777	.C20 -	000064	.TXTCK=	000136
VZCSR	002112	\$ENULL	012016	\$SVLAD	023424	.C21 =	000065	.UNLKK=	000145
WDSTOR	020366	\$EOP	011636	\$SVPC	000200	.C40 -	000110	.XMTAL	000126
WTBGND	023074	\$EOPCT	011664	\$SWR =	177400	.C79 -	000157	.Y	000131
X	002136	\$ERFLG	001103	\$SWRMK=	000000	.DAPNT-	000171		
XENA	040000	\$ERMAX	001115	\$TBIT	012014	.DCRAD	054433		

. ABS. 033762 000

ERRORS DETECTED: 0

DSKZ:CZVTMA,DSKZ:CZVTMA=DSKM:CZVTMA.P11
 RUN-TIME: 25 19.9 SECONDS
 RUN-TIME RATIO: 481/46-10.4
 CORE USED: 28K (55 PAGES)