VT61/VT61T

VT61 EXERCISER

MD-11-DZVTH-B

EP-DZVTH-B-DL-B

FICHE 1 OF 1

APR 1977

MADE IN USA

.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZVTH-B-D

PRODUCT NAME: VT61 EXERCISOR

DATE: 30-JAN-77

MAINTENANCE: DIAGNOSTIC GROUP

AUTHOR: PAUL NELSON

TABLE OF CONTENTS

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 PRECEEDED BY A SOM AND TERMINATED WITH A EOM.

THERE ARE TWO DISTINCT MODES IN WHICH THE PROGRAM CAN BE OPERATED. IN "AUTO" MODE ALL DL11'S WITH 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 16 VT61'S CONNECTED TO THE HOST COMPUTER VIA DL11-A,B,C OR D. 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

FOLLOWING IS THE OPERATING PROCEDURE FOR THE "AUTO" AND "MANUAL" MODES OF TESTING.

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 DL11'S WITH VT61 TYPE UNITS ATTACHED AND SEQUENTIALLY TEST ALL UNITS REPETITIVELY WITH TESTS 1 THRU 20.

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

THIS MODE REQUIRES THE OPERATOR TO ENTER THE ADDRESSES OF THE DL11'S TO BE TESTED (FORMAT IS 17XXXX, ECT, -UP TO 16 ENTRIES). THE ENTRIES MUST BE SEPARATED BY COMMAS AND TERMINATED WITH A CARRIAGE RETURN. ENTERING AN ILLEGAL ADDRESS WILL RESULT IN A "?" BEING TYPED AND THE ADDRESS IGNORED. THE OPERATOR MUST THEN, UPON PROGRAM REQUEST, ENTER A LIST OF TESTS TO BE EXECUTED IN THE SAME FORMAT AS THE ADDRESS ENTRY (I.E.YY,ZZ,C/R). PRECEEDING THE TERMINATING CARRIAGE RETURN WITH A 377 OCTAL WILL RESULT IN THE TESTS BEING REPETITIVELY EXECUTED FOR ALL ADDRESSES ENTERED.

SIMPLY DEPRESSING A CARRIAGE RETURN WHEN UNIT ADDRESSES ARE REQUESTED WILL RESULT IN THE MAPPING AND TESTING OF ALL GOOD DL11'S WITH OPERATIONAL VT61'S ATTACHED. HOWEVER, THE TEST LIST MUST STILL BE ENTERED VIA THE CONSOLE!! WHEN RUNNING THE EXERCISOR IN MANUAL MODE A CONTROL C (03 OCTAL) WILL RESULT IN THE TERMINATION OF TESTING AT THE END OF THE CURRENT SUBTEST.

6. ERRORS-GENERAL

6.1 NO OPERATIONAL VT61 ATTACHED

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

6.2 EXCESSIVE "FATAL" ERRORS FROM UNIT UNDER TEST

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

```
236                                     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 DL11'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.

B. 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 (DL11 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 PRECEEDED BY A SOM AND TERMINATED WITH A EOM.

```
351
```

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 REPORTEDED 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 12 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 SCRCLL) ROW 00, COL.00 IS EXPECTED TO CONTAIN A 1. IN SUBTEST "B", THE CURSOR IS PLACED AT ROW23, COL.79 AND A DATA CHARACTER "A" IS ENTERED. THE CURSOR POSITION IS THEN READ AND SHOULD BE ROW23, COL.00. THE CHAR. AT HOME IS VERIFIED TO BE A NULL.

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

9.15 TEST 15 CHECK ALL SCREEN ADDRESSES.

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

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

9.16 TEST 16 CHECK LINE FEED AND CARRIAGE RETURN

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

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

9.17 TEST 17 CHECK ERASE TO END OF SCREEN

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

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

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

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

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

END OF AUTO-ACCEPTANCE TESTS

9.21 TEST 21 KEYBOARD ECHO TEST

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

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

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

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

9.23 TEST 23 UNIT SIMULATOR TEST

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

9.24 TEST 24 PRODUCTION KEYBOARD TEST

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

%

```
                                .NLIST  MD,MC,CND
                                .LIST   ME
                        .TITLE  MAINDEC-11-DZVTH-B
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY P. NELSON
                        ;*
                        ;*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
                        ;*        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 ***
001100                  STACK=  1100
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
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)
(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)
(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
678                             .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
679                             .SBTTL  ACT11 HOOKS
(1)
(2)                             ;;*****************************************************************
(1)                             ;HOOKS REQUIRED BY ACT11
(1)          000200                     $SVPC=.                 ;SAVE PC
(1)          000046                     .=46
(1)  000046  010602                     $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(1)          000052                     .=52
(1)  000052  000000                     .WORD   0               ;;2)SET LOC.52 TO ZERO
(1)          000200                     .=$SVPC                 ;; RESTORE PC
680          000200                     .=200
681  000200  000137  002232     START:  JMP     AUTO            ;USE AUTO SELECTION OF UNITS
682  000204  000137  002264     MSTRT:  JMP     MANS            ;ALLOW OPERATOR SELECTIONOF UNITS/TESTS
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102  000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001162  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001164  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001170  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001171  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001172  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
```

```
                        .SBTTL  ERROR POINTER TABLE

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

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


001174                  $ERRTB:
                                ;GENERAL ESCAPE SEQUENCE ERROR MESSAGE

001174  023361                 EM1                     ;AN ESCAPE SEQUENCE TO VT61   FAILED.
001176  023446                 DH1                     ;TEST#,ERROR PC,2 SEQUENCE BYTES,2 SEQUENCE BYTES.
001200  001424                 DT0
001202  001444                 DF0

                                ;RECEIVE STATUS ERROR MESSAGE

001204  023511                 EM2                     ;RECEIVE STATUS ERROR
001206  023541                 DH2                     ;ADDRESS,STATUS ,ERR. BITS,CHAR.
001210  001454                 DT2
001212  001444                 DF0


                                ;RECIEVE SOFTWARE STATUS ERROR MESSAGE.

001214  023600                 EM3                     ;SFTWARE (SIAT) STATUS ERROR
001216  023641                 DH3                     ;PASS#,TEST#,GOOD STATUS,RECEIVED STATUS
001220  001502                 DT4
001222  001545                 DF6

                                ;DATA ERROR

001224  023710                 EM4                     ;DATA EXPECTED DOES NOT MATCH RECEIVE DATA.
001226  023754                 DH4                     ;TEST#,REC.CNT.,EXPECTED DATA, RECEIVE DATA
001230  001514                 DT5
001232  001444                 DF0

                                ;RECEIVE BYTE COUNT ERROR

001234  024022                 EM5                     ;BYTES EXPECTED DOES NOT EQUAL BYTES RECEIVED.
001236  024101                 DH5                     ;BYTES EXPECTED, BYTES RECEIVED
001240  001436                 DT1
001242  001452                 DF2

                                ;GENERAL DIRECT CURSOR ADDRESS FAILURE

001244  024132                 EM6                     ;CURSOR POSITION ERROR
001246  024165                 DH6                     ;GD LINE, GD COL., BD LINE, BAD COL.
001250  001454                 DT2
```

```
                                DF3

                                ;DIRECT CURSOR ADDRESS ERROR

001254  024232                  EM7                     ;DIRECT CURSOR ADDRESS ERROR
001256  024332                  DH10                    ;PASS#,TEST#,BD. ROW,BD. COL.
001260  001502                  DT4
001262  001545                  DF6


                                ;LAST TEST-SELF TEST FAILED

001264  024534                  EM10                    ;VT61 FAILED SELF-TEST FUNCTION
001266  025103                  DH11                    ;CSR, VECTOR
001270  001436                  DT1
001272  001450                  DF1

                                ;VT61 FAIL/HUNG ERROR MESSAGE
001274  024371                  EM11                    ;LAST TRANSMISSION TO VT61 CAUSED VT61 TO FAIL/HANG
001276  024275                  DH7                     ;PASS#,TEST#,ERROR PC
001300  001466                  DT3
001302  001536                  DF4

                                ;GENERAL TEST FAILURE-PRECEEDS DATA/POSITION ERROR

001304  024457                  EM12                    ;VT61 UNDERR TEST FAILED-ERROR DATA FOLLOWS
001306  024275                  DH7                     ;PASS#,TEST#,ERROR PC.
001310  001466                  DT3
001312  001536                  DF4

                                ;RECEIVE CHECKSUM ERROR

001314  024710                  EM13                    ;VT61 RECEIVER CHECKSUM ERROR
001316  024575                  DH12                    ;PASS#,TEST#,GD.CKSUM,BD CKSUM
001320  001502                  DT4
001322  001545                  DF6

                                ;TRANSMITTER CHECKSUM ERRROR

001324  024757                  EM14                    ;VT61 TRANSMITTER CHECKSUM ERROR
001326  024575                  DH12
001330  001502                  DT4
001332  001545                  DF6


                                ;XOFF FAILED TO HALT BLOCK XMIT

001334  025176                  EM15                    ;XOFF TO VT61 FAILED TO HALT BLOCK XMIT
001336  025713                  DH13                    ;PASS,TEST,VSTAT
001340  001526                  DT6
001342  001536                  DF4

                                ;XON FAILED TO RESTART BLOCK XMIT

001344  025247                  EM16                    ;XON TO VT61 FAILED TO RESTART BLOCK XMIT
```

```
   781  001346  025713                          DH13
   782  001350  001526                          DT6
   783  001352  001536                          DF4
   784
   785                                          ;NO XON AFTER UNIT WAS RESET
   786
   787  001354  025322                          EM17                    ;NO XON AFTER UNIT WAS RESET.
   788  001356  024275                          DH7                     ;PASS#,TEST#,ERROR PC
   789  001360  001526                          DT6
   790  001362  001536                          DF4
   791
   792                                          ;PERIPHERAL ABORT ERROR
   793
   794  001364  025400                          EM20                    ;LAST PERIPHERAL OPERATION ABORTED.
   795  001366  025745                          DH14                    ;PASS,TEST,ERROR PC, VSTAT
   796  001370  001502                          DT4
   797  001372  001545                          DF6
   798
   799                                          ;CANT CLEAR PERIPHERAL ABORT FLAG.
   800
   801  001374  025444                          EM21                    ;COULD NOT CLEAR LAST ABORT FLAG.
   802  001376  025745                          DH14
   803  001400  001502                          DT4
   804  001402  001545                          DF6
   805
   806                                          ;MAINTENANCE MODE DID NOT FORCE A SOM/EOM.
   807
   808  001404  025507                          EM22                    ;SOM OR EOM NOT REC. IN MAINT. MODE.
   809  001406  023641                          DH3                     ;PASS#,TEST#,EXP.STAT, ACT.STAT
   810  001410  001502                          DT4
   811  001412  001545                          DF6
   812
   813                                          ;LINE FEED OR CURSOR RIGHT AT ROW 23 DID NOT CAUSE A SCROLL.
   814
   815  001414  025575                          EM23                    ;NO SCROLL FROM LINE FEED OR CURSOR RIGHT.
   816  001416  024275                          DH7
   817  001420  001526                          DT6
   818  001422  001536                          DF4
   819
   820  001424  002230  001116  001124  DT0:    .WORD   TSTNM,$ERRPC,$GDDAT,$BDDAT,0
        001432  001126  000000
   821  001436  001124  001126  000000  DT1:    .WORD   $GDDAT,$BDDAT,0
   822  001444     000     000     000  DF0:    .BYTE   0,0,0,0
        001447     000
   823  001450     000     000          DF1:    .BYTE   0,0
   824
   825  001452     001     001          DF2:    .BYTE   1,1             ;DECIMAL TYPE
   826
   827  001454  001120  001124  001122  DT2:    .WORD   $GDADR,$GDDAT,$BDADR,$BDDAT,0
        001462  001126  000000
   828  001466  001100  002230  001116  DT3:    .WORD   $PASS,TSTNM,$ERRPC,0
        001474  000000
   829  001476     001     001     001  DF3:    .BYTE   1,1,1,1
        001501     001
   830  001502  001100  002230  001124  DT4:    .WORD   $PASS,TSTNM,$GDDAT,$BDDAT,0
        001510  001126  000000
```

```
831  001514  002230  001120  001124  DT5:    .WORD   TSTNM,$GDADR,$GDDAT,$BDDAT,0
     001522  001126  000000
832  001526  001100  002230  001120  DT6:    .WORD   $PASS,TSTNM,$GDADR,0
     001534  000000
833  001536     001     000     000  DF4:    .BYTE   1,0,0
834  001541     000     000     001  DF5:    .BYTE   0,0,1,1
     001544     001
835  001545     001     000     000  DF6:    .BYTE   1,0,0,0
     001550     000
836          001552                       .EVEN
837                                       ;INSTRUCTION DEFINITIONS
838          022626                  POP2SP  =22626
839          024646                  PUSH2SP =24646
840
841                                  ;;*****************************************************
842                                  ;DEFINITION SOFTWARE STATUS(VSTAT) REGISTER BITS
843                                  ;;*****************************************************
844
845          100000                  RXOFF   =100000                 ;SET FOR XOFF, CLEARED FOR XON
846          040000                  RSOM    =040000                 ;SET FOR SOM (START OF MESSAGE).
847          020000                  REOM    =020000                 ;SET FOR EOM (END OF MESSAGE).
848          010000                  PABRT   =010000                 ;SET FOR A PERIPHERAL ABORT.
849          004000                  RSTT    =004000                 ;SET FOR RECEIVE STATUS ERROR.
850          002000                  CKSUM   =002000                 ;SET TO CALCULATE 61 REC.  CHECKSUM
851          001000                  EPL     =001000                 ;SET WHEN END OF LINE DETECTED
852          000400                  ESC     =000400                 ;SET WHEN OCTAL 33 RECEIVED.
853          000200                  XMKIL   =000200                 ;SET WHEN TRANSMIT KILLLED.
854          000100                  TXSUM   =000100                 ;SET TO CALCULATE 61 XMIT CHECKSUM
855          000040                  REVID   =000040                 ;SET WHEN REVERSE VIDEO MODE RECEIVED.
856          000020                  COMGP   =000020                 ;SET TO CONVERT REC. CHAR. BY -137.
857          000004                  CURPOS  =000004                 ;SET WHEN CURSOR POS. RECEIVED
858          000002                  TRMID   =000002                 ;SET WHEN TERMINAL I.D. RECEVED.
859          000001                  XMDNE   =000001                 ;SET UPON TRANSMIT COMPLETE
860
861                                  ;;*****************************************************
862                                  ;DEFINITION OF DL11 CONTROL BITS
863                                  ;;*****************************************************
864
865          000200                  RECDN   =200
866          100000                  DSCHNG  =100000
867          000100                  RDENA   =000100
868          100000                  RERR    =100000
869          040000                  RORUN   =40000
870          020000                  RFMER   =20000
871          010000                  RPAR    =10000
872          000200                  TRDY    =00200
873          000100                  TENA    =00100
874          000004                  MAINT   =00004
875          000104                  TCOMB   =00104                  ;COMBINATION INTERRUPT ENABLE AND MAINT.
876
877          003600                  TOTCH   =1920.                  ;TOTAL CHARACTERS ON SCREEN
878          003601                  TOTC1   =1921.                  ;TOTAL SCREEN +1
879                                  ;;*****************************************************
880                                  ;FOLLOW ARE DL11 ADDRESS AND VECTOR STORAGE TABLES
881                                  ;;*****************************************************
882  001552  000020                  VVECT:  .BLKW   20              ;GOOD DL11 VECTOR TABLE
```

```
                                DLTBL:  .BLKW   20              ;GOOD DL11 ADDRESS TABLE
                                INTAB:  .BLKW   20              ;TABLE OF POSSIBLE DL11 ADDRESSES


                                        ;CURRENT POINTERS FOR ADDRESSES AND VECTORS
                                VECPT:  .WORD                   ;VECTOR INDEX
                                DLTPT:  .WORD                   ;ADDRESS INDEX
                                        ;ADDRESS TABLES FOR DL11 INTERFACES
                                STRTAB: .WORD   176500          ;DL11A/B
                                        .WORD   175610          ;DL11 C/D/E
                                        .WORD   0
                                ENDTAB: .WORD   176676          ;DL11 A/B
                                        .WORD   176170          ;DL11 C/D/E
                                        .WORD   0               ;END OF LIST MARKER
                                ;;**************************************************************
                                        ;VT61 ADDRESSES IN TABLE REFLECT UNIT UNDER TEST
                                ;;**************************************************************
                                VRCSR:  .WORD   0
                                VRBUF:  .WORD   0               ;RECEIVE DATA BUFFER
                                VXCSR:  .WORD   0               ;XMITTER CSR
                                VXBUF:  .WORD   0               ;XMITTER DATA BUFFER
                                VECT:   .WORD   0               ;VECTOR FOR UNIT UNDER TEST
                                CRCSR:  .WORD   0               ;CONSOLE RECEIVE CSR
                                CRBUF:  .WORD   0               ;CONSOLE DATA BUFFER


                                ;;**************************************************************
                                        ;TABLE OF VT61 COMMAND AND SEQUENCES
                                ;;**************************************************************
            000007              .BEL    =007
001750      000007              BEL:    .WORD   007             ;BELL
            000015              .CARRT  =015
001752      000015              CARRT:  .WORD   015             ;CARRIAGE RETURN
            000012              .LNFED  =012
001754      000012              LNFED:  .WORD   012             ;LINE FEED
            000011              .TAB    =011
001756      000011              TAB:    .WORD   011             ;TAB
                                ;;**************************************************************
001760      000001                      .WORD   01              ;TABLE DELIMITER (ESCN)
                                ;;**************************************************************

            000110              .CHOM   =110
001762      000110              CHOM:   .WORD   110             ;HOME CURSOR  H

            000103              .CRT    =103
001764      000103              CRT:    .WORD   103             ;CURSOR RIGHT  C

            000102              .CDWN   =102
001766      000102              CDWN:   .WORD   102             ;CURSOR DOWN  B

            000104              .CLFT   =104
001770      000104              CLFT:   .WORD   104             ;CURSOR LEFT  D

            000101              .CUP    =101
001772      000101              CUP:    .WORD   101             ;CURSOR UP  A
```

```
                    000112                  .EOS    =112
001774  000112      000112                  EOS:    .WORD   112             ;ERASE TO END OF SCREEN  J

                                            ;;**********************************************************************
001776  000002                                        .WORD   2               ;TABLE DELIMITER (ESC0)
                                            ;;**********************************************************************

                    000101                  .EMAIN  =101
002000  000101      000101                  EMAIN:  .WORD   101             ;ENTER MAINTENANCE MODE  A
                    000141                  .DMAIN  =141
002002  000141      000141                  DMAIN:  .WORD   141             ;EXIT MAINTENANCE MODE  SA

                    000105                  .LKKB   =105
002004  000105      000105                  LKKB:   .WORD   105             ;LOCK KEYBOARD  E
                    000145                  .UNLKKB =145
002006  000145      000145                  UNLKKB: .WORD   145             ;UNLOCK KEYBOARD  SE

                    000103                  .DRECT  =103
002010  000103      000103                  DRECT:  .WORD   103             ;ENABLE LINEAR MODE  C

                    000133                  .CLRCK  =133
002012  000133      000133                  CLRCK:  .WORD   133             ;CLEAR RECEIVER CHECKSUM [

                    000134                  .CLTCK  =134
002014  000134      000134                  CLTCK:  .WORD   134             ;CLEAR TRANSMITTER CHECKSUM

                    000112                  .EEMP   =112
002016  000112      000112                  EEMP:   .WORD   112             ;ENABLE REVERSE  VIDEO  J
                    000152                  .DEMP   =152
002020  000152      000152                  DEMP:   .WORD   152             ;DISABLE REVERSE VIDEO  SJ

                    000137                  .IABT   =137
002022  000137      000137                  IABT:   .WORD   137             ;INITIALIZE ABORT FLAG  -

                                            ;;**********************************************************************
002024  000003                                      .WORD   3               ;TABLE DELIMITER (ESCAPE P)
                                            ;;**********************************************************************

                    000131                  :EAPNT  =131
002026  000131      000131                  EAPNT:  .WORD   131             ;ENABLE AUTO PRINT MODE  Y
                    000171                  .DAPNT  =171
002030  000171      000171                  DAPNT:  .WORD   171             ;DISABLE AUTO PRINT MODE  SY

                    000111                  :EINST  =111
002032  000111      000111                  EINST:  .WORD   111             ;ENABLE INSERT  I
                    000151                  .ERPL   =151
002034  000151      000151                  ERPL:   .WORD   151             ;ENABLE REPLACE  SI


                                            ;;**********************************************************************
002036  000004                                      .WORD   4               ;TABLE DELIMITER (I/O)
                                            ;;**********************************************************************
```

```
                    054433                  .DCRAD  =054433
002040              054433          DCRAD:  .WORD   054433          ;DIRECT CURSOR ADDRESSING
                    067467          .R23C79 =067467
002042              067467          R23C79: .WORD   067467          ;CURSOR TO LOWER RIGHT
002044              000000                  .WORD   0

002046              047433          RCUR:   .WORD   047433          ;DIRECT CURSOR ADDRESSING
                    000131          .Y      =131
                    000131          .RDCUR  =00131
002050              000131          RDCUR:  .WORD   00131           ;READ CURSOR POSITION  Y
002052              000000                  .WORD   0

                    000117          .O      =117
002054              047433          ESCO:   .WORD   047433          ;ESCAPE O
                    000126          .XMTAL  =000126
002056              000126          XMTAL:  .WORD   000126          ;TRANSMIT ALL  V
002060              000000                  .WORD   0

002062              047433                  .WORD   047433          ;ESCAPE O
                    000127          .TCUCH  =127
002064              000127          TCUCH:  .WORD   127             ;XMIT CHARACTER AT CURSOR.  W
002066              000000                  .WORD   0

002070              047433                  .WORD   047433          ;ESCAPE O
                    000135          .TXRCK  =135
002072              000135          TXRCK:  .WORD   135             ;XMIT RECIEVER CHECKSUM ]
002074              000000                  .WORD   0

002076              047433                  .WORD   047433          ;ESCAPE O
                    000136          .TXTCK  =136
002100              000136          TXTCK:  .WORD   136             ;XMIT TRANSMITTER CHECKSUM
002102              000000                  .WORD   0

002104              147433                  .WORD   147433          ;ESCAPE O
                    000140          .RABT   =140
002106              000140          RABT:   .WORD   140             ;READ THE ABORT FLAG.  \
002110              000000                  .WORD   0

                                ;;**************************************************************
002112              177777                  .WORD   -1              ;END OF TABLE TERMINATOR
                                ;;**************************************************************


                                ;;**************************************************************
                                ;PERIPHERAL COMMANDS
                                ;;**************************************************************

                    000127      .EPNT   =127
002114              000127      EPNT:   .WORD   127             ;ENABLE PRINT MODE.  W
                    000130      .DPNT   =130
002116              000130      DPNT:   .WORD   130             ;DISABLE PRINT MODE  X

                    000135      .CPYSC  =135            ;COPY SCREEN  ]
                    000136      .ENAC   =136            ;ENABLE AUTO COPY MODE  ESC ↑
                    000137      .DISAC  =137            ;DISABLE AUTO COPY MODE  ESC -
                    000150      .PSCRN  =000150         ;PRINT THE SCREEN  H/SH
```

```
1051
1052
1053                            ;;*****************************************************************
1054                                    ;ESCAPE CODE EQUIVALENCES AND IDENTIFIERS
1055                            ;;*****************************************************************
1056
1057            000033          .ESC    =033                    ;PRIMARY ESCAPE CODE.
1058            000120          .P      =120
1059  002120    050033          ESCP:   .WORD   050033          ;ESCAPE P
1060            000124          .TSTER  =124
1061  002122    000124          TSTER:  .WORD   124             ;TEST TERMINAL(ESC 0 T)
1062            002040          ESCYI   =DCRAD                  ;ESCYI EQUALS DCRAD/DCRADI
1063            000057          SLSH    =000057                 ;SLASH CODE FOR TERMINAL  IDENT ESC.
1064            000106          CKGP    =106                    ;ENABLE REC.TO SUB 137 FROM ALL REC DATA
1065            000107          NCKGP   =107                    ;ENABLE NORMAL RECEIVED DATA.
1066            000171          CPABRT  =171                    ;COPIER ABORT
1067            000172          PRABRT  =172                    ;PRINTER ABORT
1068            000170          NABRT   =170                    ;NO ABORT  SX
1069  002124    000000          IDENT:  .WORD   0               ;VT61 IDENT CODE
1070            002054          ESCOI   =ESCO
1071            002120          ESCPI   =ESCP
1072            002126          ESCZI   =ESCZ
1073            055033          .ESCZ   =055033
1074  002126    055033          ESCZ:   .WORD   055033          ;OCTAL EQUIV. OF ESZ SEQUENCE
1075            000122          .RESET  =122
1076  002130    000122          RESET:  .WORD   122             ;VT61 INITIALIZE R
1077
1078  002132    000033          ESCN:   .WORD   000033          ;ESCAPE N-FLAG
1079  002134    020041          R01C00: .WORD   020041          ;ROW1,COL. 0
1080  002136    032041          R01C20: .WORD   032041          ;ROW01,COLUMN 20
1081  002140    020066          R22C00: .WORD   020066          ;ROW22,COL.00
1082  002142    020054          R12C00: .WORD   020054          ;ROW 12,COLUMN 00
1083            020067          .R23C00 =020067
1084  002144    020067          R23C00: .WORD   020067          ;ROW23,COL.00
1085            025440          .R00C11 =025440
1086  002146    025440          R00C11: .WORD   025440          ;ROW,COL.11
1087            032040          .R00C20 =032040
1088  002150    032040          R00C20: .WORD   032040          ;ROW 0,COLUMN 20
1089  002152    024040          R00C08: .WORD   024040          ;ROW 00,COLUMN 8
1090  002154    020040          CUHME:  .WORD   020040          ;OCTAL EQUIV. OF CURSOR HOME.
1091  002156    067440          R00C80: .WORD   067440          ;ROW 0, COLUMN 80.
1092  002160    067067          R23C78: .WORD   067067          ;ROW 23,COL. 78.
1093            000040          .R00    =40                     ;ROW 0
1094            000041          .R01    =41                     ;ROW 1
1095            000054          .R12    =54                     ;ROW 12
1096            000066          .R22    =66                     ;ROW 22
1097            000067          .R23    =67                     ;ROW 23
1098            000040          .C00    =40                     ;COLUMN 0
1099            000043          .C03    =43                     ;COL. 3
1100            000050          .C08    =50                     ;COL. 8
1101            000053          .C11    =53                     ;COL. 11
1102            000064          .C20    =64                     ;COL. 20
1103            000065          .C21    =65                     ;COL. 21
1104            000110          .C40    =110                    ;COL. 40
1105            000157          .C79    =157                    ;COL. 79
1106
```

```
                                ;;*****************************************************************
                                ;TEMPORARY STORAGE LOCATIONS AND
                                ;SPECIAL RECEIVE CODE EQUIVALENCES.
                                ;;*****************************************************************
000002                          SOM     =02                     ;START OF MESSAGE
000004                          EOM     =04                     ;END OF MESSAGE
000023                          XOFF    =23                     ;TURN OFF TRANSMISSION
000021                          XON     =21                     ;TURN ON TRANSMISSION
002162  000000                  CHRD:   .WORD   0               ;STORAGE FOR SINGLE CH. READ
002164  000000                  SVER1:  .WORD                   ;TEMP. STORAGE R1.
002166  000000                  SVER2:  .WORD                   ;TEMP. STORAGE R2.
002170  000000                  ZERO:   .WORD   0               ;MUST BE LEFT AS ZERO.
002172  003000                  TYP6:   .WORD   3000            ;TYPE 6 OCTAL CHAR-NO ZEROS
002174  000000                  TSTPTR: .WORD   0               ;TEST POINTER IN MANUAL SELECT MODE
002176  000000                  MODE:   .WORD   0               ;BYTE0=TESTING MODE,BYTE1=INTERFACE TYPE
002200  000000                  FTLCNT: .WORD   0               ;COUNT OF INCOMPLETE XMITS.
002202  000012                  ALWCNT: .WORD   10.             ;# OF ALLOWABLE INCOMPLETE XMITS.
002204  000001                  ONE:    .WORD   1
002206  000000                  TOADD:  .WORD
002210  000000                  BUBCT:  .WORD
002212  000000                  TPREG:  0
002214  000000                  PRESC:  .WORD                   ;PRIMARY ESC COMMAND
002216  000000                  ESSEQ:  .WORD                   ;SEQUENCE ASSEMBLY AREA
002220  000000                  DLAY:   .WORD
002222  000000                  ROSVE:  .WORD                   ;TEMP STORAGE FOR R0 ONLY.
002224  000000                  VSTAT:  .WORD   0
002226  000000                  BLKM:   .WORD   0               ;FLAG LOCATION FOR BLOCK MODE XMITS.
002230  000000                  TSTNM:  .WORD   0               ;DISPLAY STORAGE FOR TEST NUMBER.


                                ;;*****************************************************************
                                ;AUTOMATIC SELECTION OF UNITS. TESTS 1 THROUGH 33 WILL BE
                                ;REPITIVELY EXECUTED FOR ALL UNITS.
                                ;;*****************************************************************

002232  005037  002176          AUTO:   CLR     MODE            ;ZERO THE MODE SWITCH
002236  000137  011616                  JMP     SETA            ;;DO VECTOR SETUP
002242  004037  012260          AUTOA:  JSR     R0,TRPVEC       ;GO FIND GOOD DL11S
002246  004037  012376                  JSR     R0,CDEV         ;CHECK DL11S FOUND
002252  004037  012754                  JSR     R0,INITA        ;INSURE VT61S ON DL11
002256  000137  002514                  JMP     MODCK           ;VT61 PRESENT -BEGIN TESTING
002262  000767                          BR      AUTOA           ;NO VT61 FOUND LOOP IN CHECKING


                                ;;*****************************************************************

                                ;MANUAL UNIT AND TEST SELECTION.  UNITS CAN BE
                                ;SELECTED VIA CONSOLE OR AUTO SELECTION CAN
                                ;BE UTILIZED.  TESTS ENTERED VIA CONSOLE WILL
                                ;BE EXECUTED IN THE ORDER ENTERED.

                                ;;*****************************************************************

002264  012737  000001  002176  MANS:   MOV     #1,MODE         ;SET MODE TO MANUAL SELECT.
002272  000137  011616                  JMP     SETA            ;GO SET UP CONSTANTS
002276  104401  023231          MANSA:  TYPE    ,DMANA
002302  004037  012260                  JSR     R0,TRPVEC       ;FIND GOOD DL11'S
```

```
1164  002306  012703  001652            MOV     #INTAB,R3
1165  002312  005002            BLDADD: CLR     R2
1166
1167  002314  004037  017536    BLDADA: JSR     R0,GTNUM        ;GET A KEYBOARD INPUT
1168  002320  120127  000054            CMPB    R1,#54          ;CHAR. = COMMA?
1169  002324  001004                    BNE     1$              ;NO
1170  002326  004037  012214            JSR     R0,TMNAD        ;YES, VERIFY ENTERED ADDRESS,
1171  002332  010223                    MOV     R2,(R3)+        ; STORE THIS ADDRESS,
1172  002334  000766                    BR      BLDADD          ;AND LOOK FOR ANOTHER ADDRESS.
1173  002336  120137  001754    1$:     CMPB    R1,LNFED        ;CHAR. = LINE FEED?
1174  002342  001024                    BNE     3$              ;NO
1175  002344  005702                    TST     R2              ;ANY ENTRIES CREATED?
1176  002346  001413                    BEQ     2$              ;NO USE AUTO SELECTION OF UNITS
1177  002350  004037  012214            JSR     R0,TMNAD        ;YES, VERIFY ENTERED ADDRESS,
1178  002354  010223                    MOV     R2,(R3)+        ;STORE LAST ADDRESS,
1179  002356  013723  002170            MOV     ZERO,(R3)+      ;AND SET A TERMINATOR IN TABLE.
1180  002362  004037  012376            JSR     R0,CDEV         ;CHECK DL11 ON VT 61 SELECTED
1181  002366  005737  001612            TST     DLTBL           ;ANY DL11S GOOD?
1182  002372  001741                    BEQ     MANSA           ;NO-BACK TO SQUARE ONE
1183  002374  000412                    BR      BLDTST          ;YES- GO GET TESTS
1184  002376  004037  012376    2$:     JSR     R0,CDEV         ;CHECK DL11'S
1185  002402  004037  012754            JSR     R0,INITA        ;VERIFY DL11 HAVE VT61 ATTACHED
1186  002406  000137  002422            JMP     BLDTST          ;BEGIN TEST SELECTION
1187  002412  000731                    BR      MANSA           ;NO UNIT FOUND-LOOP
1188  002414  004037  017472    3$:     JSR     R0,OCTBIN       ;KEEP BUILDING ADDRESS
1189  002420  000735                    BR      BLDADA
1190
1191  002422  104401  023331    BLDTST: TYPE    ,DMANB          ;TYPE 2ND PART OF MANUAL MESSAGE
1192  002426  012703  001652            MOV     #INTAB,R3       ;USE INTAB AS TEST # STORAGE.
1193  002432  005004                    CLR     R4              ;CLEAR TEST COUNTER
1194  002434  005002            11$:    CLR     R2              ;CLEAR ASSEMBL WORD
1195  002436  004037  017536    10$:    JSR     R0,GTNUM        ;GET A NUMERIC CHAR.
1196  002442  120127  000054            CMPB    R1,#54          ;CHAR.=COMMA?
1197  002446  001006                    BNE     1$              ;NO
1198  002450  110223                    MOVB    R2,(R3)+        ;YES STORE A TEST #
1199  002452  005204                    INC     R4              ;AND INCREMENT TEST COUNT.
1200  002454  020437  000040            CMP     R4,32.          ;COUNT =32?
1201  002460  001415                    BEQ     MODCK           ;YES ACCEPT NO MORE ENTRIES.
1202  002462  000764                    BR      11$             ;NO KEEP LOOKING
1203  002464  120137  001754    1$:     CMPB    R1,LNFED        ;CHAR. = LINE FEED?
1204  002470  001006                    BNE     2$              ;NO
1205  002472  110223                    MOVB    R2,(R3)+        ;LOAD THE LAST TEST
1206  002474  105013                    CLRB    (R3)            ;AND INSERT TEST TABLE TERMINATOR
1207  002476  112737  000001  002176    MOVB    #1,MODE         ;SET MODE SWITCH TO MANUAL
1208  002504  000403                    BR      MODCK           ;AND BEGIN TESTING.
1209
1210  002506  004037  017472    2$:     JSR     R0,OCTBIN       ;CONVERT CHAR.
1211  002512  000751                    BR      10$
1212
1213                        ;;*****************************************************************
1214                                ;THIS ROUTINE LOOKS FOR THE OPERATIONAL MODE REQUESTED AND
1215                                ;SELECTS THE NEXT UNIT TO BE TESTED.
1216
1217                                ;MODE 0 = ACCEPTANCE TYPE TEST
1218                                ;MODE 1 = OPERATOR SELECTION OF UNITS AND SEQUENCE OF TESTS.
1219                        ;;*****************************************************************
```

```
1220
1221  002514  012737  001612  001714  MODCK:  MOV     #DLTBL,DLTPT    ;INITIAL SETUP OF ADDRESS
1222  002522  012737  001552  001712          MOV     #VVECT,VECPT    ;AND VECTOR POINTERS.
1223  002530  012701  001732          MODCA:  MOV     #VRCSR,R1       ;LOAD ADDRESS DESTINATION
1224  002534  013702  001714                  MOV     DLTPT,R2        ;LOAD CURRENT ADDRESS POINTER
1225  002540  017703  177146                  MOV     @VECPT,R3       ;LOAD CURRENT VECTOR POINTER
1226  002544  005712                          TST     (R2)            ;ALL UNITS CHECKED?
1227  002546  001013                          BNE     1$              ;NO - CONTINUE
1228  002550  005737  002176                  TST     MODE            ;CHECK MODE
1229  002554  001002                          BNE     10$
1230  002556  000137  002242                  JMP     AUTOA           ;GO RESTART AUTO MODE
1231  002562  105777  177406          10$:    TSTB    @TSTPTR         ;MANUAL LOOP REQUESTED?
1232  002566  100001                          BPL     2$              ;NO
1233  002570  000751                          BR      MODCK           ;YES-RESTART COMPLETE TEST.
1234  002572  000137  002276          2$:     JMP     MANSA           ;GO RESTART MANUAL MODE
1235  002576  004037  013174          1$:     JSR     R0,LDADD        ;NO-LOAD NEXT ADDRESSES
1236  002602  010337  001742                  MOV     R3,VECT         ;STORE VECT. OF UNIT UNDER TEST
1237  002606  012723  014102                  MOV     #INTRC,(R3)+    ;YES - NOW SET UP RECEIVE VECTOR
1238  002612  012723  000340                  MOV     #340,(R3)+      ;AND SET RECEIVER PSW TO 7
1239  002616  012723  015024                  MOV     #INTXM,(R3)+    ;SET UP TRANSMIT VECTOR
1240  002622  012723  000340                  MOV     #340,(R3)+      ;AND SET PSW TO 7.
1241  002626  005046                          CLR     -(SP)           ;CLEAR THE PSW,LSI11 STYLE.
1242  002630  012746  002636                  MOV     #100$,-(SP)
1243  002634  000002                          RTI
1244  002636  010237  001714          100$:   MOV     R2,DLTPT        ;SAVE ADDRESS POINTER.
1245  002642  012737  030511  014766          MOV     #RCRLB+477,REBUF        ;SET UP END OF BUFFER
1246  002650  012737  031211  015274          MOV     #TCRLB+477,TEBUF
1247  002656  012737  030012  014764          MOV     #RCRLB,RBBUF    ;INITIIALIZE REC.BUFFER.
1248  002664  012737  030512  015272          MOV     #TCRLB,TBBUF    ;INITIALIZE TRANSMIT BUFFER.
1249  002672  004037  016266                  JSR     R0,RESPTR       ;RESET INTERRUPT POINTERS.
1250  002676  005037  002226                  CLR     BLKM            ;CLEAR BLOCK MODE FLAG.
1251  002702  005037  020616                  CLR     XMZER           ;CLEAR ZERO TRANSMIT FLAG
1252  002706  005037  002224                  CLR     VSTAT           ;CLEAR ALL INTERRUPT FLAGS
1253  002712  004037  015456                  JSR     R0,ZFLAG        ;ISSUE ESC Z TO VT61
1254  002716  012637  002124                  MOV     (SP)+,IDENT     ;;POP STACK INTO IDENT
1255  002722  100002                          BPL     11$             ;IF IDENT IS -1,CLEAR IT.
1256  002724  005037  002124                  CLR     IDENT
1257  002730                          11$:
 (2)  002730  012637  002162                  MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
1258  002734  001375                          BNE     11$
1259  002736  104401  001171                  TYPE    ,$CRLF
1260  002742  104401  025032                  TYPE    ,DVUNIT         ;ISSUE UNIT UNDER TEST MESSAGE
1261  002746  013746  001732                  MOV     VRCSR,-(SP)     ;;SAVE VRCSR FOR TYPEOUT
 (1)                                                                  ;;TYPE THE ADDRESS
 (1)  002752  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  002754     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
 (1)  002755     001                          .BYTE   1               ;;TYPE LEADING ZEROS
1262  002756  017746  176730                  MOV     @VECPT,-(SP)    ;;SAVE @VECPT FOR TYPEOUT
 (1)                                                                  ;;TYPE THE VECTOR
 (1)  002762  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  002764     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
 (1)  002765     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1263  002766  013746  002124                  MOV     IDENT,-(SP)     ;;SAVE IDENT FOR TYPEOUT
 (1)                                                                  ;;TYPE THE IDENT
 (1)  002772  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  002774     006                          .BYTE   6               ;;TYPE 6 DIGITS
```

```
   (1)  002775     000                                .BYTE   0               ;;SUPPRESS LEADING ZEROS
1264    002776  104401  001171                        TYPE    ,$CRLF          ;CARRIAGE RETURN AND LINE FEED
1265    003002  032737  000001  002124                BIT     #BIT00,IDENT    ;UNIT HAVE A COPIER?
1266    003010  001402                                BEQ     20$             ;NO
1267    003012  104401  025151                        TYPE    ,DCOPYR         ;YES-ISSUE COPIER MESSAGE
1268    003016  032737  000002  002124  20$:   BIT     #BIT01,IDENT    ;UNIT HAVE A PRINTER?
1269    003024  001402                                BEQ     21$             ;NO
1270    003026  104401  025123                        TYPE    ,DPRTR          ;YES-ISSUE PRINTER MESSAGE.
1271    003032  062737  000002  001712  21$:   ADD     #2,VECPT        ;LEAVE WITH VECPOINT AT NEXT VECTOR.
1272    003040  005037  002200                        CLR     FTLCNT          ;CLEAR COUNT OF FATAL XMITS.
1273    003044  012737  031214  031212                MOV     #ABBUF,ABUFP    ;RESET THE REC. DATA POINTER
1274    003052  052777  000100  176652                BIS     #RDENA,@VRCSR   ;SET THE REC. INT. ENABLE  FOR TESTS
1275    003060  105737  002176                        TSTB    MODE            ;CHECK TESTING MODE
1276    003064  001403                                BEQ     ASTRT           ;AUTO MODE
1277    003066  012737  001652  002174                MOV     #INTAB,TSTPTR   ;LOAD THE INITIAL TEST NUMBER
1278
1279                                        ;;**********************************************************************
1280                                        ;;**********************************************************************
1281                                                ;THIS TEST ISSUES ALL ESCAPE SEQUENCES AND
1282                                                ;INSURES THE VT61 HAS NOT FAILED DURING AN
1283                                                ;ESC SEQUENCE BY ISSUING A ESC Z TO FORCE A
1284                                                ;VT61 RESPONSE.  THE PURPOSE OF THE TEST IS TO ATTEMPT TO
1285                                                ;INSURE THAT SUBSEQUENT TESTS WILL NOT RESULT IN
1286                                                ;A "HUNG" UNIT.  DATA IS NOT EVALUATED.
1287                                        ;;**********************************************************************
1288
1289    003074                              ASTRT:
   (3)                                      ;;**********************************************************************
   (2)  003074  000004                      TST1:   SCOPE
   (1)  003076  012737  000001  001160              MOV     #1,$TIMES       ;;DO 1 ITERATION
   (1)  003104  012737  003112  001106              MOV     #ESTST,$LPADR   ;;SET SCOPE LOOP ADDRESS
1290
1291    003112  012701  001750              ESTST:  MOV     #BEL,R1         ;POINT TO FIRST COMMAND
1292    003116  042777  000100  176606              BIC     #RDENA,@VRCSR   ;CLEAR REC. INT. ENABLE
1293    003124  113737  001102  002230              MOVB    $TSTNM,TSTNM    ;LOAD THE TEST NUMBER.
1294    003132  005037  002214                       CLR     PRESC
1295    003136  005004                              CLR     R4
1296    003140                              ZERST:
   (2)  003140  013746  002170                      MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
1297    003144  012702  002214                      MOV     #PRESC,R2       ;SET UP SEQNENCE ADDRESS
1298    003150  012103                      GCMD:   MOV     (R1)+,R3        ;LOAD THE COMMAND
1299    003152  001405                              BEQ     1$              ;IF CHAR. ZERO,MUST BE XMIT TERMINATOR
1300    003154  100535                              BMI     ESTEX           ;TABLE EXPENDED - EXIT TEST.
1301    003156  120327  000004                      CMPB    R3,#4           ;IS COMMAND ACTUALLY A DELIMITER?
1302    003162  103442                              BLO     DELIM           ;YES, GO UPDATE FUNCTIONS
1303    003164  001471                              BEQ     SPTN            ;NO, ITS A "10" - SPECIAL CASE.
1304    003166  005704                      1$:     TST     R4              ;SEE IF FLAG INDICATING SEQ.
1305    003170  100472                              BMI     SEQ4            ;4 IS SET. - YES EXIT
1306    003172  010337  002216              2$:     MOV     R3,ESSEQ        ;PUSH THE SEQUENCE TO BE TESTED
1307    003176                              INXMT:
   (2)  003176  013746  002216                      MOV     ESSEQ,-(SP)     ;;PUSH ESSEQ ON STACK
1308    003202  005704                              TST     R4              ;DOES THIS SEQUENCE REQUIRE
1309    003204  001402                              BEQ     3$              ;ADDITIONAL ESC?
1310    003206  013746  002214                      MOV     PRESC,-(SP)     ;;PUSH PRESC ON STACK
1311
1312    003212  004037  013456              3$:     JSR     R0,TESC         ;GO TRANSMIT THIS SEQUENCE.
```

```
1313
1314  003216  005704                    4$:     TST     R4              ;IN I/O SEQUENCES?
1315  003220  100007                            BPL     40$             ;NO
1316  003222  012737  000054  017224            MOV     #44.,DCOUNT     ;YES,SET UP TO DELAY 1+ SEC.
1317  003230  004037  017162                    JSR     R0,DELAY
1318  003234  005777  176474                    TST     @VRBUF          ;CLEAR ANY RECEIVE FLAGS
1319
1320  003240  004037  015456            40$:    JSR     R0,ZFLAG        ;ISSUE ESC Z SEQUENCE-GET IDENT
1321  003244                            5$:
  (2) 003244  012637  002162                    MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
1322  003250  123737  002162  002124            CMPB    CHRD,IDENT      ;HAVE WE POPPED THE IDENT?
1323  003256  001045                            BNE     T1ERR           ;NO-ERROR CONDITION
1324  003260                            POPIT:
  (2) 003260  012637  002162                    MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
1325  003264  001375                            BNE     .-4
1326  003266  000724                            BR      ZERST           ;GET NEXT COMMAND
1327
1328  003270  120327  000001            DELIM:  CMPB    R3,#1
1329  003274  001407                            BEQ     1$              ;FIRST DELIMITER - SET ESCN
1330
1331  003276  120327  000002                    CMPB    R3,#BIT01       ;SECOND DELIMITER - SET ESC0
1332  003302  001412                            BEQ     2$
1333
1334  003304  120327  000003                    CMPB    R3,#3           ;THIRD DELIMITER - SET ESCP
1335  003310  001413                            BEQ     3$
1336  003312  000716                            BR      GCMD            ;INVALID CHARACTER - GET ANOTHER
1337
1338  003314  012704  000001            1$:     MOV     #1,R4           ;SET FIRST DELIMITER FLAG.
1339  003320  013737  002132  002214            MOV     ESCN,PRESC      ;INSERT ESCN.
1340  003326  000710                            BR      GCMD
1341
1342  003330  013737  002054  002214    2$:     MOV     ESC0,@#PRESC    ;INSERT ESC0
1343  003336  000704                            BR      GCMD
1344
1345  003340  013737  002120  002214    3$:     MOV     ESCP,@#PRESC    ;INSERT ESCP
1346  003346  000700                            BR      GCMD
1347
1348  003350  012704  177777            SPTN:   MOV     #-1,R4          ;SET FLAG INDICATING I/O
1349  003354  000675                            BR      GCMD            ;SEQUENCES.
1350
1351  003356  005703                    SEQ4:   TST     R3              ;CHECK IF COMMAND = 0
1352  003360  001706                            BEQ     INXMT           ;YES, COMPLETE SEQUENCE ASSEMBLED
1353  003362  110322                            MOVB    R3,(R2)+        ;NO - KEEP ASSEMBLING
1354  003364  000303                            SWAB    R3              ;POSITION HIGH  ORDER BIT
1355  003366  110322                            MOVB    R3,(R2)+        ;AND ASSEMBLE IT
1356  003370  000667                            BR      GCMD            ;GET ANOTHER BYTE
1357
1358  003372  004037  015660            T1ERR:  JSR     R0,CLREG
1359  003376  013737  002214  001124            MOV     PRESC,$GDDAT    ;AND INSERT IN ERROR
1360  003404  000337  001124                    SWAB    $GDDAT          ;REASSEMBLE FAILING SEQUENCES
1361  003410  013737  002216  001126            MOV     ESSEQ,$BDDAT
1362  003416  105737  002217                    TSTB    ESSEQ+1         ;IF UPPER BYTE IS CLEAR DO NOT SWAP
1363  003422  001402                            BEQ     1$
1364  003424  000337  001126                    SWAB    $BDDAT          ;MESSAGE 1
1365  003430  104001                    1$:     ERROR   1               ;ISSUE ERROR MESSAGE
1366  003432  005237  002200                    INC     FTLCNT          ;INCREMENT FATAL XMIT COUNT.
```

```
1367  003436  023737  002200  002202          CMP     FTLCNT,ALWCNT   ;FATAL XMITS EXCEEDED ALLOWED?
1368  003444  103003                          BHIS    FTEX1           ;YES-EXIT.
1369  003446  000704                          BR      POPIT           ;CLEAR THE STACK AND TRY ANOTHER COMMAND
1370  003450                          ESTEX:
 (2)  003450  012637  002162                  MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
1371  003454  052777  000100  176250  FTEX1:  BIS     #RDENA,@VRCSR   ;SET THE REC. INT. ENABLE FOR TESTS
1372
1373                                  ;;**************************************************************
1374                                          ;ROUTINE TO INSURE ENTERING MAINTENANCE MODE CAUSES SOM AND
1375                                          ;EOM TO BE APPENDED TO ALL TRANSMITS FROM VT61 UNDER TEST.
1376                                          ;MAINTENANCE MODE IS ENTERED , THEN AN ESCAPE Z SEQUNECE
1377                                          ;IS ISSUED TO THE UNIT AND THE RESULTING TRANSMISSION IS
1378                                          ;CHECKED OF SOM/EOM.
1379                                  ;;**************************************************************
1380
1381                                  ;;**************************************************************
 (2)  003462  000004                  TST2:   SCOPE
 (1)  003464  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
 (1)  003472  012737  003500  001106          MOV     #CKMNT,$LPADR   ;;SET SCOPE LOOP ADDRESS
1382
1383  003500  004037  015302          CKMNT:  JSR     R0,RESETV       ;RESET THE UNIT AND SETMAINT. MODE.
1384  003504  112777  000002  011564          MOVB    #SOM,@TBUFP     ;ISSUE START OF MESSAGE.
1385  003512  004037  016156                  JSR     R0,XMIT1
1386  003516  113777  002126  011552          MOVB    ESCZ,@TBUFP
1387  003524  004037  016156                  JSR     R0,XMIT1        ;SEND AN IDENT REQUEST.
1388  003530  113777  002127  011540          MOVB    ESCZ+1,@TBUFP
1389  003536  004037  016156                  JSR     R0,XMIT1
1390  003542  112777  000004  011526          MOVB    #EOM,@TBUFP     ;ISSUE END OF MESSAGE.
1391  003550  004037  016156                  JSR     R0,XMIT1
1392  003554  005037  002220                  CLR     DLAY            ;SET UP SOM DELAY OF 100M.S.
1393  003560  032737  040000  002224  1$:     BIT     #RSOM,VSTAT     ;RECEIVED THE START OF MESSAGE?
1394  003566  001003                          BNE     CKEOM           ;YES-GO LOOK FOR EOM.
1395  003570  005337  002220                  DEC     DLAY            ;NO-RUN TIMEOUT DELAY
1396  003574  001371                          BNE     1$              ;AND KEEP LOOKING.
1397
1398  003576  012701  000062          CKEOM:  MOV     #50.,R1         ;SET MAX DELAY FOR 500 M.S.
1399  003602  032737  020000  002224  1$:     BIT     #REOM,VSTAT     ;RECEIVED END OF MESSAGE?
1400  003610  001007                          BNE     10$             ;YES-CHECK FOR BOTH RECEIVED.
1401  003612  012737  000001  017224          MOV     #1,DCOUNT       ;DELAY FOR 10 M..S.
1402  003620  004037  017162                  JSR     R0,DELAY
1403  003624  005301                          DEC     R1              ;AND KEEP LOOKING.
1404  003626  001365                          BNE     1$
1405  003630  032737  040000  002224  10$:    BIT     #RSOM,VSTAT     ;RECEIVED SOM?
1406  003636  001404                          BEQ     2$              ;NO ISSUE ERROR
1407  003640  032737  020000  002224          BIT     #REOM,VSTAT     ;RECEIVED EOM?
1408  003646  001007                          BNE     EXMNT           ;YES, NO ERRORS-EXIT.
1409  003650  012737  006001  001124  2$:     MOV     #6001,$GDDAT    ;LOAD ERROR WITH EXPECTED
1410  003656  013737  002224  001126          MOV     VSTAT,$BDDAT    ;AND ACTUAL STATUS.
1411  003664  104022                          ERROR   22
1412
1413  003666  000240                  EXMNT:  NOP
1414                                  ;;**************************************************************
1415                                          ;THIS TEST INSURES THAT THE CURSOR WILL RESPOND
1416                                          ;TO DIRECT CUSROR ADDRESSING. THE UNIT IS RESET AND THE CURSOR
1417                                          ;POSITION IS VERIFIED TO BE HOME . THE CURSOR IS THEN MOVED
1418                                          ;TO POSITION ROW 23 COLUMN 80 AND THE POSITION IS AGAIN
```

```
1419                                    ;VERIFIED. ERRORS ARE REPORTED IF THE POSITIONS ARE INCORRECT.
1420
1421                                    ;;*****************************************************************
1422
1423
1424                                    ;;*****************************************************************
 (2) 003670  000004                     TST3:   SCOPE
 (1) 003672  012737  000005  001160             MOV     #5,$TIMES       ;;DO 5 ITERATIONS
 (1) 003700  012737  003706  001106             MOV     #CURS1,$LPADR   ;;SET SCOPE LOOP ADDRESS
1425
1426  003706  013701  015272             CURS1:  MOV     TBBUF,R1        ;USE R1 AS XMIT BUFFER POINTER.
1427  003712  004037  015302                     JSR     R0,RESETV       ;RESET THE UNIT AND WAIT FOR XON.
1428  003716  013721  002054                     MOV     ESCOI,(R1)+     ;CLFT. RESET, READ CURSOR
1429  003722  113721  002050                     MOVB    RDCUR,(R1)+     ;POSITION, CURSOR LEFT.
1430  003726  012737  000003  015300             MOV     #3,XMCNT        ;XMIT 3 BYTES
1431
1432  003734  004037  015702                     JSR     R0,XMREC        ;XMIT AND RECEIVE.
1433  003740  000402                             BR      10$             ;NORMAL EXIT.
1434  003742  104011                             ERROR   11              ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
1435  003744  000446                             BR      2$              ;EXIT TEST.
1436  003746  013701  030012             10$:    MOV     RCRLB,R1        ;GET THE CURRENT CURSOR POSITION.
1437  003752  020137  002154                     CMP     R1,CUHME        ;CURSOR REALLY HOME?
1438  003756  001405                             BEQ     1$              ;YES EXIT
1439  003760  104012                             ERROR   12              ;VT61 FAILURE MESSAGE
1440  003762  013746  002154                     MOV     CUHME,-(SP)     ;;PUSH CUHME ON STACK
1441  003766  004037  016346                     JSR     R0,CURER        ;GO LOAD AND ISSUE CURSOR ERROR
1442
1443  003772  013701  015272             1$:     MOV     TBBUF,R1        ;LOAD XMIT BUFFER WITH
1444  003776  013721  002040                     MOV     DCRAD,(R1)+
1445  004002  013721  002042                     MOV     R23C79,(R1)+    ;CURSOR TO ROW 23,COL.79
1446  004006  013721  002054                     MOV     ESCOI,(R1)+     ;READ CURSOR POSITION
1447  004012  013721  002050                     MOV     RDCUR,(R1)+     ;IT AND CURSOR RIGHT
1448  004016  012737  000007  015300             MOV     #7,XMCNT        ;XMIT 7 BYTES.
1449  004024  004037  015702                     JSR     R0,XMREC        ;XMIT AND RECEIVE
1450  004030  000402                             BR      20$             ;NORMAL EXIT.
1451  004032  104011                             ERROR   11              ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
1452  004034  000412                             BR      2$              ;EXIT TEST.
1453  004036  012701  030012             20$:    MOV     #RCRLB,R1
1454
1455  004042  023711  002042                     CMP     R23C79,(R1)     ;CHECK CURSOR POSITION TO LOWER RT.
1456  004046  001405                             BEQ     2$              ;OK, EXIT
1457  004050  104012                             ERROR   12              ;VT61 FAILURE MESSAGE
1458  004052  013746  002042                     MOV     R23C79,-(SP)    ;;PUSH R23C79 ON STACK
1459  004056  004037  016346                     JSR     R0,CURER        ;LOAD AND ISSUE CURSOR ERROR .
1460  004062  000240                     2$:     NOP
1461                                    ;;*****************************************************************
1462                                            ;ROUTIME TO INSURE THE UNIT CAN ENTER LINEAR ADDRESSING
1463                                            ;MODE.  81 CHARACTERS ARE ISSUED TO THE UNIT UNDER TEST
1464                                            ;THEN THE CURSOR POSITION IS READ AND MUST BE ROW1,COL.0.
1465                                    ;;*****************************************************************
1466
1467                                    ;;*****************************************************************
 (2) 004064  000004                     TST4:   SCOPE
 (1) 004066  012737  000005  001160             MOV     #5,$TIMES       ;;DO 5 ITERATIONS
 (1) 004074  012737  004102  001106             MOV     #CKLIN,$LPADR   ;;SET SCOPE LOOP ADDRESS
1468
```

```
1469  004102  000037  015302          CKLIN:  JSR     R0,RESETV       ;RESET THE UNIT-SET MAINT AND LINEAR MODES
1470  004106  013701  015272                  MOV     TBBUF,R1
1471  004112  012703  000120                  MOV     #80.,R3
1472  004116  004037  017226                  JSR     R0,BLDINC       ;LOAD XMIT BUFFER WITH 80 CHAR AND
1473  004122  013721  002046                  MOV     RCUR,(R1)+
1474  004126  013721  002050                  MOV     RDCUR,(R1)+     ;READ CURSOR POSINION.
1475  004132  012737  000123  015300          MOV     #83.,XMCNT
1476  004140  004037  015702                  JSR     R0,XMREC        ;XMIT THE BUFFER.
1477  004144  000402                          BR      1$
1478  004146  104011                          ERROR   11              ;LAST XMIT CAUSED UNIT TO HANG.
1479  004150  000421                          BR      LINXT           ;EXIT TEST
1480  004152  023777  002134  010604  1$:     CMP     R01COO,@RBBUF   ;CURSOR  AT ROW1,COL. 0?
1481  004160  001415                          BEQ     LINXT           ;YES-EXIT
1482  004162  013737  002054  001124          MOV     ESCO,$GDDAT
1483  004170  000337  001124                  SWAB    $GDDAT
1484  004174  013737  002010  001126          MOV     DRECT,$BDDAT    ;ISSUE ESC SEQUENCE AND CURSOR
1485  004202  104001                          ERROR   1
1486  004204  013746  002134                  MOV     R01COO,-(SP)    ;;PUSH R01COO ON STACK
1487  004210  004037  016346                  JSR     R0,CURER
1488  004214  000240                  LINXT:  NOP
1489
1490
1491                                  ;;**************************************************************
1492                                          ;TEST TO INSURE OPERATION OF XON/XOFF COMMANDS
1493                                          ;FROM VT61. XOFF IS FORCED BY TRANSMITTING LINE 23 WHILE SIMUL-
1494                                          ;TANEOUSLY FILLING THE SILO WITH DATA. AFTER SENSING
1495                                          ;THE XOFF, THE TEST WAITS FOR THE TRANSMIT TO FINISH AND
1496                                          ;INSURES XON OCCURS BEFORE THE MAX. TRANSFER TIME HAS ELAPSED.
1497                                          ;(30 SECONDS)
1498                                  ;;**************************************************************
1499
1500                                  ;;**************************************************************
 (2)  004216  000004                  ST5:    SCOPE
 (1)  004220  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 (1)  004226  012737  004234  001106          MOV     #BASC3,$LPADR   ;;SET SCOPE LOOP ADDRESS
1501  004234  013701  015272          BASC3:  MOV     TBBUF,R1        ;R1 = 1ST XMIT BUFFER ADDRESS.
1502  004240  012737  001001  002226          MOV     #1001,BLKM      ;SET UP TO XMIT A SOM -DATA- EOM.
1503  004246  005037  002224                  CLR     VSTAT
1504  004252  004037  015302                  JSR     R0,RESETV       ;RESET THE UNIT AND WAIT FOR XON.
1505  004256  013721  002040                  MOV     DCRAD,(R1)+
1506  004262  013721  002144                  MOV     R23COO,(R1)+    ;CURSOR TO ROW 23, COL.0
1507  004266  013721  002054                  MOV     ESCO,(R1)+
1508  004272  013721  002056                  MOV     XMTAL,(R1)+     ;TRANSMIT THE LINE.
1509  004276  012703  000050                  MOV     #40.,R3
1510  004302  004037  017226                  JSR     R0,BLDINC       ;40 CHAR. OF INCREMENTING CHAR.
1511  004306  012737  000057  015300          MOV     #47.,XMCNT      ;SET UP TO XMIT 47 BYTES
1512  004314  052777  000100  175414          BIS     #TENA,@VXCSR    ;TRANSMIT ENABLES
1513  004322  012703  000050                  MOV     #40.,R3         ;MAXIMUM DELAY EQUAL 400 M.S.
1514  004326  012737  000001  017224  2$:     MOV     #1,DCOUNT
1515  004334  004037  017162                  JSR     R0,DELAY        ;DELAY FOR 10 MILLISEC.
1516  004340  032737  100000  002224          BIT     #RXOFF,VSTAT    ;CHECK FOR XOFF
1517  004346  001007                          BNE     3$              ;FOUND IT EXIT THIS SECTION.
1518  004350  005303                          DEC     R3              ;DELAYED 400 M.S.?
1519  004352  001365                          BNE     2$              ;NO-KEEP LOOKING FOR XOFF.
1520  004354  104012                          ERROR   12              ;GENERAL VT61 FAILURE MESSAGE
1521  004356  012746  100000                  MOV     #100000,-(SP)   ;;PUSH #100000 ON STACK
```

```
1522  004362  004037  015516            3$:   JSR     R0,CKSFT        ;GO REPORT ERROR
1523  004366
 (2)  004366  012746  000001                  MOV     #XMDNE,-(SP)    ;;PUSH #XMDNE ON STACK
1524  004372  012746  000062                  MOV     #50.,-(SP)      ;;PUSH #50. ON STACK
1525  004376  004037  020620                  JSR     R0,WTBGND
1526  004402  000411                          BR      EXIT3           ;TIMEOUT-EXIT TEST.
1527  004404  127727  024602  000021          CMPB    @ABUFP,#XON     ;RECEIVED A XON?
1528  004412  001405                          BEQ     EXIT3           ;YES-NO ERROR-EXIT
1529
1530  004414  104012                          ERROR   12              ;GENERAL VT61 FAILURE MESSAGE
1531  004416  012746  000001                  MOV     #000001,-(SP)   ;;PUSH #000001 ON STACK
1532  004422  004037  015516                  JSR     R0,CKSFT
1533  004426  004037  016266          EXIT3:  JSR     R0,RESPTR       ;RESET INTERRUPT POINTERS.
1534
1535                                  ;;**********************************************************************
1536                                          ;ROUTINE TO VERIFY OPERATION OF XOFF AND XON TO THE VT61.
1537                                          ;A FULL SCREEN TRANSMIT IS INITIATED AND A SERIES OF XOFF AND
1538                                          ;XON ARE ISSUED TO THE TERMINAL SEQUENTIALLY.
1539                                          ;ERRORS ARE REPORTED IF XOFF DOES NOT STOP,OR XON RESTART
1540                                          ;THE TRANSMISSION. TEST IS ENDED WHEN EOM IS SENSED.
1541                                  ;;**********************************************************************
1542
1543                                  ;;**********************************************************************
 (2)  004432  000004            TST6:  SCOPE
 (1)  004434  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
 (1)  004442  012737  004450  001106          MOV     #ONOF61,$LPADR  ;;SET SCOPE LOOP ADDRESS
1544
1545  004450  004037  015302          ONOF61: JSR     R0,RESETV       ;RESET THE UNIT AND WAIT FOR XON.
1546  004454  042737  077577  002224          BIC     #77577,VSTAT    ;CLEAR THE FLAGS
1547  004462  013746  002170                  MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
1548  004466  013746  002056                  MOV     XMTAL,-(SP)     ;;PUSH XMTAL ON STACK
1549  004472  013746  002054                  MOV     ESCO,-(SP)      ;;PUSH ESCO ON STACK
1550  004476  004037  013456                  JSR     R0,TESC
1551  004502  012737  000010  017224  ONOFLP: MOV     #10,DCOUNT      ;ALLOW 100 M.S. FOR OPERATION
1552  004510  004037  017162                  JSR     R0,DELAY        ;TO BEGIN.
1553  004514  112777  000023  010554          MOVB    #XOFF,@TBUFP
1554  004522  004037  016156                  JSR     R0,XMIT1        ;SEND A XOFF TO VT61.
1555  004526  012704  000036                  MOV     #30.,R4
1556  004532  013705  031212          OFFLP:  MOV     ABUFP,R5        ;ALLOW 300M.S. FOR XMIT TO CEASE
1557  004536  012737  000001  017224          MOV     #1,DCOUNT
1558  004544  004037  017162                  JSR     R0,DELAY
1559  004550  023705  031212                  CMP     ABUFP,R5
1560  004554  001406                          BEQ     ONOFA           ;XMIT STOPPED-GO RESTART IT.
1561  004556  005304                          DEC     R4
1562  004560  001364                          BNE     OFFLP           ;COUNTER NO EQUAL 300 MS-LOOP
1563  004562  013737  002224  001120          MOV     VSTAT,$GDADR    ;UNIT DID NOT RESPOND TO XOFF
1564  004570  104015                          ERROR   15              ;ISSUE ERROR
1565
1566  004572  112777  000021  010476  ONOFA: MOVB    #XON,@TBUFP
1567  004600  004037  016156                  JSR     R0,XMIT1        ;SEND A XON TO THE VT61.
1568  004604  012704  000036                  MOV     #30.,R4         ;SET UP FOR 300MS DELAY.
1569  004610  032737  020000  002224  ONLP:  BIT     #REOM,VSTAT     ;EOM RECEIVED?
1570  004616  001020                          BNE     ONOFXT          ;YES-EXIT
1571  004620  013705  031212                  MOV     ABUFP,R5
1572  004624  012737  000001  017224          MOV     #1,DCOUNT
1573  004632  004037  017162                  JSR     R0,DELAY        ;ALLOW 300 MS FOR XMIT TO RESTART
```

```
1574  004636  023705  031212            CMP     ABUFP,R5
1575  004642  001317                    BNE     ONOFLP          ;IT RESTARTED-GO STOP IT.
1576  004644  005304                    DEC     R4
1577  004646  001360                    BNE     ONLP            ;NOT YET 300 MS LOOP.
1578  004650  013737  002224  001120    MOV     VSTAT,$GDADR    ;XMIT DIT NOT RESTART-ISSUE
1579  004656  104016                    ERROR   16              ;ERROR AND EXIT
1580  004660  000240            ONOFXT: NOP
1581
1582                                    ;;*****************************************************************
1583                                    ;ROUTINE TO TEST VT61 RAM AND THE COMMUNICATION PATHS.
1584                                    ;THIS ROUTINE ISSUES A SERIES OF PATTERNS(77/100,100/77,
1585                                    ;52/125,INCREMENTING,AND REV. VIDEO INCREMENTING) TO THE VT61.
1586                                    ;THE SCREEN IS THEN TRANSMITTED TO THE HOST AND AFTER EACH
1587                                    ;ITERATION RECEIVED DATA IS CHECKED AND ALL ERRORS(INCLUDING
1588                                    ;TRANSMISSION) ARE REPORTED.
1589                                    ;MITTED TO THE HOST COMPUTER AND THE RESULTS ARE CHECKED AND
1590                                    ;ALL ERRORS(INCLUDING TRANSMISSION) REPORTED.
1591                                    ;;*****************************************************************
1592
1593                                    ;;*****************************************************************
 (2)  004662  000004            TST7:   SCOPE
 (1)  004664  012737  000001  001160    MOV     #1,$TIMES       ;;DO 1 ITERATION
 (1)  004672  012737  004700  001106    MOV     #MEM1,$LPADR    ;;SET SCOPE LOOP ADDRESS
1594
1595  004700  004037  015302    MEM1:   JSR     R0,RESETV       ;RESET THE UNIT AND WAIT FOR XON.
1596  004704  005005                    CLR     R5              ;CLEAR PATTERN OFFSET.
1597  004706  016504  005414    MEMA:   MOV     MPATT(R5),R4    ;LOAD PATTERN TO BE TRANSMMITTED
1598  004712  004037  016266            JSR     R0,RESPTR       ;RESET POINTERS
1599  004716  042737  077577  002224    BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1600  004724  012702  003600            MOV     #TOTCH,R2       ;LOAD A COUNT OF SCREEN
1601  004730  112777  000002  010340    MOVB    #SOM,@TBUFP     ;ISSUE START OF MESSAGE.
1602  004736  004037  016156            JSR     R0,XMIT1
1603  004742  005302            MEMB:   DEC     R2              ;DECREMENT XMIT COUNT
1604  004744  001414                    BEQ     10$             ;COUNT = ZERO?
1605
1606  004746  004037  005362    12$:    JSR     R0,PATGN        ;NO-GENERATE NEXT BYTE TO XMIT.
1607  004752  110477  010320            MOVB    R4,@TBUFP       ;LOAD THE CHARACTER.
1608  004756  004037  016156            JSR     R0,XMIT1        ;NO-XMIT ANOTHER BYTE.
1609  004762  023737  002200  002202    CMP     FTLCNT,ALWCNT   ;EXCEEDED FATAL ERROR COUNT?
1610  004770  103764                    BLO     MEMB            ;NO-CHECK IF ANOTHER TRANSMISSION REQUIRED.
1611  004772  000137  005434            JMP     MEMXT           ;YES-GO ABORT TEST.
1612  004776  112777  000004  010272  10$:    MOVB    #EOM,@TBUFP     ;ISSUE END OF MESSAGE.
1613  005004  004037  016156            JSR     R0,XMIT1
1614  005010  004037  016266            JSR     R0,RESPTR       ;RESET INTERRUPT POINTERS.
1615
1616  005014  013701  015272            MOV     TBBUF,R1        ;LOAD XMIT BUFFER WITH
1617  005020  013721  002132            MOV     ESCN,(R1)+
1618  005024  013721  001762            MOV     CHOM,(R1)+      ;CURSOR HOME
1619  005030  013721  002126            MOV     ESCZI,(R1)+     ;ESCAPE Z
1620  005034  013721  002054            MOV     ESCO,(R1)+
1621  005040  013721  002056            MOV     XMTAL,(R1)+     ;TRANSMIT ALL
1622  005044  013711  001754            MOV     LNFED,(R1)      ;LINE FEED.
1623  005050  012737  000010  015300    MOV     #8.,XMCNT       ;SET UP TO XMIT 8 BYTES
1624  005056  004037  015702            JSR     R0,XMREC        ;XMIT,WAIT FOR REC. EOM
1625  005062  000402                    BR      1$              ;NORMAL EXIT
1626  005064  104011                    ERROR   11              ;LAST TRANSMIT CAUSED VT61 TO HANG
```

```
1627  005066  000562                    BR      MEMXT           ;EXIT TEST
1628  005070  042737  077577  002224  1$:  BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1629  005076  005002                    CLR     R2              ;CLEAR RECEIVE COUNTER.
1630  005100  016504  005414            MOV     MPATT(R5),R4    ;LOAD PATTERN
1631  005104  012703  031012            MOV     #TCRLB+300,R3   ;SET UT ERROR STORAGE
1632  005110  013701  014764            MOV     RBBUF,R1        ;SET UP RECEIVE POINTER
1633  005114  005037  002220        MEMC:  CLR     DLAY            ;SET UP TIME OUT DELAY
1634  005120  013737  014764  014770    MOV     RBBUF,RBUFP     ;RESET RECEIVE POINTER
1635  005126  023701  014770        1$:  CMP     RBUFP,R1        ;RECEIVED A CHAR?
1636  005132  001013                    BNE     MEMD            ;YES-GO CHECK IT.
1637  005134  032737  020000  002224    BIT     #REOM,VSTAT     ;HAVE WE RECEIVED EOM?
1638  005142  001033                    BNE     CKDAT           ;YES, GO CHECK FOR DATA ERRORS
1639  005144  005337  002220            DEC     DLAY            ;RUN TIME OUT DELAY.
1640  005150  001366                    BNE     1$              ;NOT EXPIRED-KEEP LOOKING.
1641  005152  005237  002200            INC     FTLCNT          ;TRANSMISSION FAILED-INCR. FATAL COUNT
1642  005156  104011                    ERROR   11
1643  005160  000525                    BR      MEMXT
1644  005162  005202                MEMD:  INC     R2              ;DATA IN. INCREMENT COUNTER
1645  005164  004037  005362            JSR     R0,PATGN        ;GET GOOD CHARACTER ,PUT IN R4 AND
1646  005170  122705  000010            CMPB    #10,R5          ;CHECKING REV. VIDEO DATA?
1647  005174  001002                    BNE     1$              ;NO-DO NOT MODIFY
1648  005176  052704  000200            BIS     #BIT07,R4       ;YES-FORCE BIT 7.
1649  005202  121104                1$:  CMPB    (R1),R4         ;COMPARE DATA
1650  005204  001743                    BEQ     MEMC
1651  005206  020227  003600            CMP     R2,#TOTCH       ;COMPARING LAST CHAR?
1652  005212  001740                    BEQ     MEMC            ;YES-NEVER COUNT AS A ERROR.
1653
1654  005214  020327  031062            CMP     R3,#TCRLB+350   ;STORED 20 ERRORS?
1655  005220  103335                    BHIS    MEMC            ;YES-STORE NO MORE.
1656  005222  110423                    MOVB    R4,(R3)+        ;STORE THE GOOD DATA.
1657  005224  111123                    MOVB    (R1),(R3)+      ;STORE THE BAD DATA.
1658  005226  010223                    MOV     R2,(R3)+        ;STORE THE RECEIVE COUNT.
1659  005230  000731                    BR      MEMC
1660  005232  022703  031012        CKDAT:  CMP     #TCRLB+300,R3
1661  005236  001415                    BEQ     CKMEM
1662  005240  012701  031012            MOV     #TCRLB+300,R1   ;LOAD FIRST ERROR ADDRESS.
1663  005244  004037  015660        1$:  JSR     R0,CLREG        ;CLEAR ERROR REGISTERS
1664  005250  112137  001124            MOVB    (R1)+,$GDDAT    ;LOAD THE GOOD DATA.
1665  005254  112137  001126            MOVB    (R1)+,$BDDAT    ;LOAD THE ERROR BUFFER
1666  005260  012137  001120            MOV     (R1)+,$GDADR    ;LOAD RECEIVE COUNT
1667  005264  104004                    ERROR   4               ;ISSUE DATA ERROR MESSAGE.
1668  005266  020103                    CMP     R1,R3           ;ISSUED ALL ERRORS?
1669  005270  103765                    BLO     1$              ;NO-CONTINUE
1670
1671  005272  020227  003600        CKMEM:  CMP     R2,#TOTCH       ;DID WE XFER 1920 TIMES?
1672  005276  001406                    BEQ     1$              ;YES - GO CHECK STATUS
1673  005300  012737  003600  001124    MOV     #TOTCH,$GDDAT   ;NO, PUT GOOD COUNT IN GDDAT
1674  005306  010237  001126            MOV     R2,$BDDAT       ;AND ACTUAL COUNT IN BDDAT.
1675  005312  104005                    ERROR   5               ;ISSUE COUNT ERROR.
1676
1677  005314                        1$:
  (2) 005314  012746  060000            MOV     #60000,-(SP)    ;;PUSH #60000 ON STACK
1678  005320  004037  015516            JSR     R0,CKSFT
1679  005324  062705  000002            ADD     #2,R5           ;INCREMENT PATTERN POINTER
1680  005330  005765  005414            TST     MPATT(R5)       ;TEST NEXT PATTERN
1681  005334  001437                    BEQ     MEMXT           ;ZERO-END OF TEST EXIT.
```

```
                                        BPL     2$              ;NOT INCRMENTING PATTERN.
                                        CMPB    #10,R5          ;SET REVERSE VIDEO?
                                        BNE     2$              ;NO.
                                        MOV     #SETREV,R3      ;YES-ENTER REVERSE VIDEO
                                        JSR     R0,LDXMIT       ;AND RE-ISSUE INCREMENTING PATTERN.
                                2$:     JMP     MEMA            ;NOT ZERO, GO EXERCISE IT.

                                PATGN:  BIC     #200,R4         ;CLEAR REV. VIDEO BIT IF SET.
                                        TST     R4              ;CHECK R4 FOR PATTERN
                                        BMI     1$              ;IF MINUS, DO INCREMENTING.
                                        SWAB    R4              ;OTHERWISE SWAP BYTES AND
                                        RTS     R0              ;EXIT.
                                1$:     INCB    R4              ;ADD ONE TO INCREMENTING
                                        CMPB    R4,#177         ;HAVE WE EXCEEDED LIMIT
                                        BLO     2$              ;NO, EXIT
                                        MOV     MPATT(R5),R4    ;YES, RESET PATTERN AND
                                2$:     RTS     R0              ;EXIT.
                                        .EVEN
                                MPATT   =.
                                        .WORD   037500          ;PATTERN 77,100
                                        .WORD   040077          ;PATTERN 100,77
                                        .WORD   025125          ;PATTERN 52,125
                                        .WORD   100040          ;PATTERN INCREMENTING
                                        .WORD   100040          ;PATTERN INCREMENTING-REV. VIDEO.
                                        .WORD   0               ;PATTERN TABLE TERMINATOR
                                        ;SEQUENCE TO ENTER REVERSE VIDEO.
                                SETREV: .BYTE   .ESC,.O,.EEMP,0
                                MEMXT:  NOP

                                ;;**********************************************************************

                                        ;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.

                                ;;**********************************************************************

                                ;;**********************************************************************
  (2)   005436  000004          TST10:  SCOPE
  (1)   005440  012737  000003  001160          MOV     #3,$TIMES       ;;DO 3 ITERATIONS
  (1)   005446  012737  005454  001106          MOV     #CKSUMA,$LPADR  ;;SET SCOPE LOOP ADDRESS

                                CKSUMA: JSR     R0,RESETV       ;RESET THE UNIT AND WAIT FOR XON.
                                        MOV     #1001,BLKM      ;SET UP TO XMIT A SOM -DATA- EOM.
                                        JSR     R0,RESPTR       ;RESET INTERRUPT POINTERS
                                        MOV     #TSUMA,R3       ;DIS. RECT. MODE AND CLEAR CHECKSUM
                                        JSR     R0,LDXMIT
                                        BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
                                        MOV     TBBUF,R1        ;LOAD XMIT BUFFER WITH
```

```
1734  005514  012703  000473                MOV     #315.,R3
1735  005520  004037  017226                JSR     R0,BLDINC       ;314 INCREMENTING CHAR.
1736  005524  113721  002132                MOVB    ESCN,(R1)+
1737  005530  113721  001762                MOVB    CHOM,(R1)+      ;CURSOR HOME
1738  005534  113721  002054                MOVB    ESCO,(R1)+
1739  005540  113721  002055                MOVB    ESCO+1,(R1)+
1740  005544  113711  002072                MOVB    TXRCK,(R1)      ;TRANSMIT RECEIVER CHECKSUM.
1741  005550  005004                        CLR     R4              ;CLEAR CHECKSUM REGISTER
1742  005552  012705  000004                MOV     #EOM,R5         ;PRELOAD CHECKSUM REG. WITH
1743  005556  004037  017646                JSR     R0,CALCK        ;EOM FROM PRIOR XMIT.
1744  005562  052737  002000  002224        BIS     #CKSUM,VSTAT    ;REQUEST CHECKSUM CALCULATIONS.
1745  005570  012737  000500  015300        MOV     #320.,XMCNT     ;SETUP TO XMIT 320 BYTES
1746  005576  052777  000100  174132        BIS     #TENA,@VXCSR    ;ENABLE XMIT INTERRUPTS
1747  005604  012746  020000                MOV     #REOM,-(SP)     ;;PUSH #REOM ON STACK
1748  005610  012746  000012                MOV     #10.,-(SP)      ;;PUSH #10. ON STACK
1749  005614  004037  020620                JSR     R0,WTBGND       ;LOOK FOR EOM.
1750  005620  000534                        BR      CKEXT           ;ERROR EXIT IF NOT FOUND
1751  005622  127704  007136                CMPB    @RBBUF,R4       ;COMPARE CHECKSUMS
1752  005626  001414                        BEQ     CKSUMB          ;GOOD GO TO SUBTEST B
1753  005630  004037  015660                JSR     R0,CLREG        ;BAD COMPARE
1754  005634  110437  001124                MOVB    R4,$GDDAT       ;LOAD CALCULATED CHECKSUM
1755  005640  117737  007120  001126        MOVB    @RBBUF,$BDDAT   ;AND VT61 RECEIVER CHECKSUM
1756  005646  104013                        ERROR   13              ;ISSUE ERROR
1757  005650  012746  060001                MOV     #60001,-(SP)    ;;PUSH #60001 ON STACK
1758  005654  004037  015516                JSR     R0,CKSFT        ;ERROR.
1759
1760  005660  042737  077577  002224  CKSUMB: BIC    #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1761  005666  005004                        CLR     R4              ;CLEAR CHECKSUM REGISTER
1762  005670  012737  001001  002226        MOV     #1001,BLKM      ;SET UP TO XMIT A SOM -DATA- EOM.
1763  005676  052737  000100  002224        BIS     #TXSUM,VSTAT    ;SET UP FOR XMIT CHECKSUM GENERATION.
1764  005704  013701  015272                MOV     TBBUF,R1        ;LOAD XMIT BUFFER WITH
1765  005710  004037  017714                JSR     R0,LDBUF        ;LOAD THE BUFFER WITH:
1766  005714      033     117      134       .BYTE   .ESC,.O,.CLTCK,.ESC,.O,.XMTAL,.ESC,.O,.TXTCK,0
      005717      033     117      126
      005722      033     117      136
      005725      000
1767  005726  012737  000011  015300        MOV     #9.,XMCNT       ;SET UP TO XMIT 9 BYTES
1768  005734  052777  000100  173774        BIS     #TENA,@VXCSR    ;ALLOW XMIT INTERRUPTS
1769  005742  012746  000001                MOV     #XMDNE,-(SP)    ;;PUSH #XMDNE ON STACK
1770  005746  012746  000002                MOV     #2,-(SP)        ;;PUSH #2 ON STACK
1771  005752  004037  020620                JSR     R0,WTBGND       ;LOOK FOR XMIT DONE.
1772  005756  000455                        BR      CKEXT           ;TIME OUT - EXIT TEST.
1773  005760  005037  002220        CKSRC:  CLR     DLAY            ;SET UP TIME OUT DELAY
1774  005764  013702  031212                MOV     ABUFP,R2        ;RESET THE RECEIVER FLAG
1775  005770  023702  031212        1$:     CMP     ABUFP,R2        ;RECEIVED A CHAR?
1776  005774  001007                        BNE     2$              ;YES-GO CHECK IT.
1777  005776  005337  002220                DEC     DLAY            ;RUN TIME OUT DELAY.
1778  006002  001372                        BNE     1$
1779  006004  005237  002200                INC     FTLCNT          ;TIMED OUT-INCREMENT FATAL XMIT COUNT
1780  006010  104011                        ERROR   11              ;ISSUE HUNG MESSAGE AND EXIT.
1781  006012  000437                        BR      CKEXT
1782  006014  122777  000004  023170  2$:     CMPB    #EOM,@ABUFP     ;RECEIVED EOM CHAR?
1783  006022  001356                        BNE     CKSRC
1784  006024  042737  020000  002224        BIC     #REOM,VSTAT     ;CLEAR THE EOM FLAG
1785  006032  032737  020000  002224        BIT     #REOM,VSTAT     ;NOW WAIT FOR LAST EOM FLAG
1786  006040  001774                        BEQ     .-6             ;FROM XMIT TRANSMITTER CHECKSUM.
```

```
1787  006042  120477  006716                  CMPB    R4,@RBBUF       ;COMPARE 61 TO HOST CHECKSUM.
1788  006046  001421                          BEQ     CKEXT           ;EQUAL - EXIT TEST
1789  006050  004037  015660                  JSR     R0,CLREG
1790  006054  110437  001124                  MOVB    R4,$GDDAT       ;LOAD THE HOST CALCULATED CHECKSUM
1791  006060  117737  006700  001126          MOVB    @RBBUF,$BDDAT   ;LOAD THE VT61 TRANSMITTED CHECKSUM
1792  006066  104014                          ERROR   14              ;ISSUE VT61 XMIT CHECKSUM ERROR
1793  006070  012746  060001                  MOV     #60001,-(SP)    ;;PUSH #60001 ON STACK
1794  006074  004037  015516                  JSR     R0,CKSFT        ;CHECK FOR STATUS ERROR
1795  006100  000404                          BR      CKEXT
1796
1797  006102     033     117     103  ITSUMA: .BYTE   .ESC,.0,.DRECT,.ESC,.0,.CLRCK,0,0
      006105     033     117     133
      006110     000     000
1798
1799  006112  004037  016266          CKEXT:  JSR     R0,RESPTR
1800
1801                                  ;;**********************************************************************
1802                                          ;ROUTINE TO INSURE BASIC CURSOR COMMANDS
1803                                          ;RESULT IN CORRECT CURSOR MOVEMENT. COMMANDS
1804                                          ;ARE ISSUED IN THE SEQUENCE:  RESET, CURSOR RIGHT,
1805                                          ;CURSOR DOWN, CURSOR LEFT, AND CURSOR UP. THE READ
1806                                          ;CURSOR POSITION COMMAND IS ISSUED AFTER EVERY
1807                                          ;CURSOR COMMAND AND CURRENT IS COMPARED TO GOOD
1808                                          ;AND ANY ERRORS REPORTED.
1809                                  ;;**********************************************************************
1810
1811                                  ;;**********************************************************************
 (2)  006116  000004                  TST11:  SCOPE
 (1)  006120  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
 (1)  006126  012737  006134  001106          MOV     #CURS1A,$LPADR  ;;SET SCOPE LOOP ADDRESS
1812
1813  006134  013701  015272          CURS1A: MOV     TBBUF,R1        ;LOAD XMIT BUFFER ADDRESS
1814  006140  004037  015302                  JSR     R0,RESETV       ;RESET THE UNIT AND WAIT FOR XON.
1815  006144  004037  017714                  JSR     R0,LDBUF        ;LOAD THE BUFFER WITH:
1816  006150     033     103     033          .BYTE   .ESC,.CRT,.ESC,.0,.RDCUR,.ESC,.CDWN,.ESC
      006153     117     131     033
      006156     102     033
1817  006160     117     131     033          .BYTE   .0,.RDCUR,.ESC,.CLFT,.ESC,.0,.RDCUR
      006163     104     033     117
      006166     131
1818  006167     033     101     033          .BYTE   .ESC,.CUP,.ESC,.0,.RDCUR,.BEL,0
      006172     117     131     007
      006175     000
1819  006176  012737  000024  015300          MOV     #20.,XMCNT      ;SET TO XMIT 20 CHARACTERS
1820  006204  012737  000004  016150          MOV     #4,RECITT       ;SET RECEIVE ITERATION TO 4
1821  006212  012737  030612  016152          MOV     #TCRLB+100,WDSTOR;SET UP WORD STORAGE POINTER
1822  006220  004037  015702                  JSR     R0,XMREC        ;XMIT ,AND WAIT FOR REC.DONE
1823  006224  000402                          BR      11$             ;NORMAL EXIT
1824  006226  104011                          ERROR   11              ;LAST XMIT CAUSED VT61 TO HANG.
1825  006230  000436                          BR      CUR1XT          ;EXIT TEST
1826  006232  012701  006316          11$:    MOV     #GDCURP,R1      ;R1=GOOD POSITION TABLE
1827  006236  012702  030612                  MOV     #TCRLB+100,R2   ;R2=ACTUAL CURSOR POSITION
1828  006242  012703  001764                  MOV     #CRT,R3         ;R3=CURSOR COMMAND TABLE
1829
1830  006246  021112          12$:    CMP     (R1),(R2)       ;COMPARE GOOD TO ACTUAL
1831  006250  001415                  BEQ     2$              ;OK-GO UPDATE POINTERS.
```

```
1832   006252  113737  002132  001125          MOVB    ESCN,$GDDAT+1
1833   006260  111337  001124                  MOVB    (R3),$GDDAT     ;LOAD COMMAND IN ESC ERROR
1834   006264  005037  001126                  CLR     $BDDAT
1835   006270  104001                          ERROR   1               ;AND ISSUE IT
1836   006272  011237  030012                  MOV     (R2),RCRLB      ;LOAD BAD CURSOR POSITION
1837   006276  011146                          MOV     (R1),-(SP)      ;;PUSH (R1) ON STACK
1838   006300  004037  016346                  JSR     R0,CURER        ;LOAD AND ISSUE CURSRO ERROR MESSAGE
1839   006304  022122                  2$:     CMP     (R1)+,(R2)+     ;INCREMENT POSITION POINTERS.
1840   006306  022337  001772                  CMP     (R3)+,CUP       ;CHECK FOR COMMAND TERM.(CUP).
1841   006312  001355                          BNE     12$             ;NOT AT TERMINATOR-COMPARE AGAIN
1842   006314  000404                          BR      CUR1XT          ;EXIT TEST
1843
1844
1845   006316  020440                  GDCURP: .WORD   20440           ;ROW 0, COL. 1
1846   006320  020441                          .WORD   20441           ;ROW 1, COL. 1
1847   006322  020041                          .WORD   20041           ;ROW 1, COL. 0
1848   006324  020040                          .WORD   20040           ;ROW 0, COL. 0
1849   006326  000240                  CUR1XT: NOP
1850
1851                                    ;;**********************************************************************
1852                                    ;ROUTINE TO INSURE THAT READ CHARACTER AT CURSOR
1853                                    ;FUNCTIONS CORRECTLY. COMMAND SEQUENCE IS:  RESET, A, CURSOR
1854                                    ;LEFT, READ  CHARACTER AT CURSOR.
1855                                    ;AN ERROR IS REPORTED IF THE LAST READ IS NOT AN "A".
1856                                    ;;**********************************************************************
1857
1858                                    ;;**********************************************************************
 (2)   006330  000004                  TST12:  SCOPE
 (1)   006332  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
 (1)   006340  012737  006346  001106          MOV     #CURS1B,$LPADR  ;;SET SCOPE LOOP ADDRESS
1859
1860   006346  013701  015272          CURS1B: MOV     TBBUF,R1
1861   006352  004037  015302                  JSR     R0,RESETV       ;RESET THE UNIT AND WAIT FOR XON.
1862   006356  012721  000101                  MOV     #101,(R1)+      ;A
1863   006362  113721  002132                  MOVB    ESCN,(R1)+
1864   006366  113721  001770                  MOVB    CLFT,(R1)+      ;CURSOR LEFT
1865   006372  013721  002054                  MOV     ESCOI,(R1)+
1866   006376  013711  002064                  MOV     TCUCH,(R1)      ;TRANSMIT CH. AT CURSOR
1867   006402  012737  000006  015300          MOV     #6,XMCNT        ;SET UP TO XMIT 6 CHARACTERS
1868   006410  004037  015702                  JSR     R0,XMREC        ;XMIT STRING AND WAIT FOR EOM.
1869   006414  000402                          BR      10$             ;NORMAL EXIT
1870   006416  104011                          ERROR   11              ;LAST XMIT CAUSED VT61 TO HANG/FAIL
1871   006420  000430                          BR      2$              ;EXIT TEST
1872   006422  127727  006336  000101  10$:    CMPB    @RBBUF,#101     ;CHARACTER READ=A
1873   006430  001424                          BEQ     2$              ;YES-NEXT SUBTEST
1874   006432  013737  002054  001124          MOV     ESCOI,$GDDAT
1875   006440  000337  001124                  SWAB    $GDDAT          ;REASSEMBLE ESC DATA
1876   006444  005037  001126                  CLR     $BDDAT
1877   006450  113737  002064  001127          MOVB    TCUCH,$BDDAT+1  ;LOAD FAILING ESC SEQUENCE
1878   006456  104001                          ERROR   1               ;AND ISSUE IT
1879   006460  004037  015660                  JSR     R0,CLREG
1880   006464  112737  000101  001124          MOVB    #101,$GDDAT     ;LOAD GOOD CH. AND CH.
1881   006472  117737  006266  001126          MOVB    @RBBUF,$BDDAT
1882   006500  104004                          ERROR   4               ;READ AND ISSUE THEM.
1883
1884   006502  000240                  2$:     NOP                     ;END OF TEST
```

```
                                ;;*****************************************************************
                                ;ROUTINE TO VERIFY OPERATION OF REPLACE AND INSERT MODE.
                                ;INITIALLY ROW 0 IS WRITTEN TO 80 INCREMENTING CHAR.
                                ;ON THE FIRST PASS(REPLACE MODE) A CHARACTER IS REPLACED
                                ;AT HOME AND THE CHAR. AT ROW0,COL.0(172) AND ROW1,COL0(NULL)
                                ;ARE VERIFIED. ON THE SECOND PASS, INSERT MODE IS ENTERED
                                ;AND THE RESULTING INSERTION(AT HOME) IS VERIFIED.ROW0,COL0
                                ;SHOULD BE 172 AND ROW1,COL0 SHOULD BE 161.
                                ;;*****************************************************************

                                ;;*****************************************************************
(2)  006504  000004             TST13:  SCOPE
(1)  006506  012737  000005  001160        MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)  006514  012737  006522  001106        MOV     #INRPL,$LPADR   ;;SET SCOPE LOOP ADDRESS

006522  004037  015302          INRPL:  JSR     R0,RESETV       ;RESET THE UNIT
006526  013701  015272                  MOV     TBBUF,R1
006532  005201                          INC     R1              ;LEAVE ROOM IN BUFFER FOR SOM.
006534  012703  000120                  MOV     #80.,R3         ;CREATE A LINE OF 80 INCREMENTING
006540  004037  017226                  JSR     R0,BLDINC       ;CHAR. ON THE SCREEN.
006544  105011                          CLRB    (R1)
006546  013703  015272                  MOV     TBBUF,R3
006552  004037  016226                  JSR     R0,LDXMIT
006556  005005                          CLR     R5              ;USE R5 AS TEST INDEXER.
006560  012737  000002  016150  INAG:   MOV     #2,RECITT       ;SET UP TO RECEIVE 2 CHAR.
006566  012737  030712  016154          MOV     #TCRLB+200,BYSTOR       ;SET UP STORAGE AREA.
006574  013701  015272                  MOV     TBBUF,R1
006600  004037  017714                  JSR     R0,LDBUF        ;LOAD THE BUFFER WITH:
006604     033     110     172          .BYTE   .ESC,.CHOM,172,.ESC,.CHOM,.ESC,.0,.TCUCH
006607     033     110     033
006612     117     127
006614     033     102     033          .BYTE   .ESC,.CDWN,.ESC,.0,.TCUCH,0
006617     117     127     000
006622  012737  000015  015300          MOV     #13.,XMCNT      ;SET UP TO XMIT 13 CAHR.
006630  004037  015702                  JSR     R0,XMREC
006634  000402                          BR      1$              ;NORMAL EXIT
006636  104011                          ERROR   11              ;LAST XMIT CAUSED UNIT TO HANG.
006640  000433                          BR      INRXT           ;EXIT TEST.
006642  026537  006720  030712  1$:     CMP     TDATA(R5),TCRLB+200     ;COMPARE GOOD TO REC.DATA.
006650  001407                          BEQ     2$              ;GOOD-LOOP OR EXIT.
006652  016537  006712  001126          MOV     TFUNCT(R5),$BDDAT
006660  013737  002120  001124          MOV     ESCP,$GDDAT     ;LOAD ESCAPE SEQ. ERROR.
006666  104001                          ERROR   1
006670  005725                  2$:     TST     (R5)+           ;INCREMENT INDEXER.
006672  020527  000004                  CMP     R5,#4           ;THRU WITH TEST?
006676  001414                          BEQ     INRXT           ;YES-EXIT.
006700  012703  006724                  MOV     #ENSRT,R3       ;NO-SECOND PASS- ENTER
006704  004037  016226                  JSR     R0,LDXMIT       ;INSERT MODE AND DO AGAIN.
006710  000723                          BR      INAG

006712  000151  000111  177777  TFUNCT: .WORD   .ERPL,.EINST,-1
006720     172     000     172  TDATA:  .BYTE   172,0,172,160
006723     160
006724     033     120     111  ENSRT:  .BYTE   .ESC,.P,.EINST,0
006727     000
006730  000240                  INRXT:  NOP
```

```
                                ;;*****************************************************************
                                ;ROUTINE TO INSURE VT61 WILL SCROLL IF A LINE FEED
                                ;IS ISSUED FORM ROW 23 OR A CURSOR RIGHT FROM ROW23,COL. 79.
                                ;IN SUBTEST A, ROW 0 IS INTIIALLY 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
                                ;CHARACTER AT HOME IS VERIFIED TO BE A NULL.
                                ;;*****************************************************************

                                ;;*****************************************************************
(2)  006732  000004             TST14:  SCOPE
(1)  006734  012737  000005  001160        MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)  006742  012737  006750  001106        MOV     #CKSCRA,$LPADR  ;;SET SCOPE LOOP ADDRESS

1948 006750  004037  015302         CKSCRA: JSR     R0,RESETV       ;RESET THE UNIT.
1949 006754  013701  015272                 MOV     TBBUF,R1
1950 006760  004037  017714                 JSR     R0,LDBUF        ;LOAD THE XMIT BUFFER WITH:
1951 006764    060     033   102            .BYTE   60,.ESC,.CDWN,.ESC,.CLFT,61,.ESC,.Y,.R23,.C00
     006767    033     104   061
     006772    033     131   067
     006775    040
1952 006776    012     033   110            .BYTE   .LNFED,.ESC,.CHOM,.ESC,.0,.TCUCH,.BEL,0
     007001    033     117   127
     007004    007     000
1953 007006  012737  000020  015300         MOV     #16.,XMCNT      ;SET UP TO XMIT 16 BYTES.
1954 007014  004037  015702                 JSR     R0,XMREC
1955 007020  000402                         BR      1$              ;NORMAL EXIT
1956 007022  104011                         ERROR   11              ;LAST XMIT CAUSED UNIT TO HANG.
1957 007024  000452                         BR      GDSCRL          ;EXIT TEST.
1958 007026  127727  005732  000061  1$:    CMPB    @RBBUF,#61      ;CHARACTER AT HOME A 1?
1959 007034  001401                         BEQ     CKSCRB          ;YES-NEXT TEST
1960 007036  104023                         ERROR   23              ;NO-ISSUE NO SCROLL ERROR.
1961 007040  012737  000002  016150  CKSCRB: MOV     #2,RECITT       ;SET UP FOR TWO REC. LOOPS.
1962 007046  012737  030712  016152         MOV     #TCRLB+200,WDSTOR       ;SET UP CURSOR POSITION STROAGE.
1963 007054  013701  015272                 MOV     TBBUF,R1
1964 007060  004037  017714                 JSR     R0,LDBUF        ;LOAD XMIT BUFFER WITH:
1965 007064    033     131   067            .BYTE   .ESC,.Y,.R23,.C79,101,.ESC,.0,.RDCUR
     007067    157     101   033
     007072    117     131
1966 007074    033     110   033            .BYTE   .ESC,.CHOM,.ESC,.0,.TCUCH,0
     007077    117     127   000
1967 007102  012737  000015  015300         MOV     #13.,XMCNT      ;SET UP TO XMIT 13 BYTES.
1968 007110  004037  015702                 JSR     R0,XMREC        ;XMIT AND WAIT FOR RECEIVED DONE.
1969 007114  000402                         BR      1$
1970 007116  104011                         ERROR   11              ;LAST XMIT CAUSED VT61 TO HANG.
1971 007120  000414                         BR      GDSCRL          ;ERROR EXIT
1972 007122  127737  005636  002170  1$:    CMPB    @RBBUF,ZERO     ;NULL RECEIVED?
1973 007130  001410                         BEQ     GDSCRL          ;YES-EXIT TEST
1974 007132  104023                         ERROR   23              ;NO-ISSUE NO SCROLL ERROR.
1975 007134  013777  030712  005622         MOV     TCRLB+200,@RBBUF        ;LOAD RECEIVED CURSOR POSITION.
1976 007142  013746  002144                 MOV     R23C00,-(SP)    ;;PUSH R23C00 ON STACK
1977 007146  004037  016346                 JSR     R0,CURER        ;GO ISSUE CURSOR ERROR.
```

```
 1978  007152  000240                    GDSCRL: NOP
 1979
 1980
 1981                                    ;;*****************************************************************
 1982                                            ;THIS TEST INSURES THAT THE VT61 CURSOR CAN BE
 1983                                            ;POSITIONED TO VERY POSSIBLE ROW/COLUMN POSITON
 1984                                            ;ON THE SCREEN.  THIS IS TESTED BY FILLING THE
 1985                                            ;COMPLETE SCREEN WITH A CHARACTER(A) AND THEN
 1986                                            ;POSITONING THE CURSOR (VIA DCA) TO EVERY POSITION
 1987                                            ;AND THE "A" AT THAT POSITION IS REPLACED WITH A SPACE.
 1988                                            ;THE SCREEN IS THEN READ TO VERIFY THAT ONLY SPACES
 1989                                            ;EXIST ON THE SCREEN.  ALL POSITIONS CONTAINING
 1990                                            ;NON-SPACES ARE REPORTED.
 1991
 1992                                    ;;*****************************************************************
 1993
 1994                                    ;;*****************************************************************
  (2)  007154  000004                    TST15:  SCOPE
  (1)  007156  012737  000001  001160            MOV     #1,$TIMES       ;;DO 1 ITERATION
  (1)  007164  012737  007172  001106            MOV     #CURS2,$LPADR   ;;SET SCOPE LOOP ADDRESS
 1995
 1996  007172  042737  077577  002224  CURS2:   BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
 1997  007200  004037  015302                    JSR     R0,RESETV       ;RESET THE UNIT AND WAIT FOR XON.
 1998  007204  012702  003600                    MOV     #TOTCH,R2       ;LOAD A COUNT OF SCREEN(1920).
 1999  007210  112777  000002  006060            MOVB    #SOM,@TBUFP     ;ISSUE START OF MESSAGE.
 2000  007216  004037  016156                    JSR     R0,XMIT1
 2001  007222  005302                  1$:     DEC     R2              ;DECREMENT XMIT COUNT
 2002  007224  001413                          BEQ     10$             ;COUNT = ZERO?
 2003
 2004  007226  112777  000101  006042          MOVB    #101,@TBUFP     ;LOAD THE CHARACTER(A).
 2005  007234  004037  016156                    JSR     R0,XMIT1        ;NO-XMIT ANOTHER BYTE.
 2006  007240  023737  002200  002202          CMP     FTLCNT,ALWCNT   ;EXCEEDED FATAL ERROR COUNT?
 2007  007246  103765                          BLO     1$              ;NO-CHECK IF DONE NOW
 2008  007250  000137  007652                    JMP     C2XT            ;YES-ABORT TESTING THIS UNIT.
 2009  007254  112777  000004  006014  10$:     MOVB    #EOM,@TBUFP     ;ISSUE END OF MESSAGE.
 2010  007262  004037  016156                    JSR     R0,XMIT1
 2011  007266  004037  016266                    JSR     R0,RESPTR       ;RESET INTERRUPT POINTERS.
 2012  007272  013737  002160  016550            MOV     R23C78,LNRW     ;SET UP 1ST ADDRESS
 2013  007300  013701  015272                    MOV     TBBUF,R1        ;LOAD XMIT BUFFER WITH
 2014  007304  013721  002040                    MOV     DCRAD,(R1)+
 2015  007310  010102                          MOV     R1,R2           ;R2 POINTS TO CURSOR ADD. IN BUFFER
 2016  007312  013721  002160                    MOV     R23C78,(R1)+    ;CURSOR TO LOWER RIGHT -1.
 2017  007316  112721  000040                    MOVB    #40,(R1)+       ;SPACE
 2018  007322  012737  000005  015300  2$:      MOV     #5,XMCNT        ;SET UP TO XMIT 5 CHARACTERS
 2019  007330  042737  077577  002224          BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
 2020  007336  052777  000100  172372          BIS     #TENA,@VXCSR    ;XMIT INTERRUPTS.
 2021  007344  012746  000001                    MOV     #XMDNE,-(SP)    ;;PUSH #XMDNE ON STACK
 2022  007350  012746  000002                    MOV     #2,-(SP)        ;;PUSH #2 ON STACK
 2023  007354  004037  020620                    JSR     R0,WTBGND       ;LOOK FOR  XMIT DONE
 2024  007360  000534                          BR      C2XT            ;NOT FOUND-ERROR EXIT
 2025  007362  021237  002154                    CMP     (R2),CUHME      ;DELETED TO HOME?
 2026  007366  001405                          BEQ     3$              ;YES
 2027  007370  004037  016444                    JSR     R0,CMPOS        ;NO-GET NEXT POSITION TO BE DELETED
 2028  007374  013712  016550                    MOV     LNRW,(R2)       ;LOAD IT IN XMIT BUFFER
 2029  007400  000750                          BR      2$              ;AND DELETE IT.
 2030  007402  004037  016266          3$:     JSR     R0,RESPTR       ;RESET INTERRUPT POINTERS
```

```
2031  007406  013737  002154  016550          MOV     CUHME,LNRW      ;LOAD INITIAL CHECK POSITION(HOME)
2032  007414  012737  001001  002226          MOV     #1001,BLKM      ;SET UP TO XMIT A SOM -DATA- EOM.
2033  007422  013701  015272                  MOV     TBBUF,R1        ;LOAD XMIT BUFFER WITH
2034  007426  010102                          MOV     R1,R2           ;STORE ERRORS IN XMIT BUFFER
2035  007430  042737  077577  002224          BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2036  007436  013721  002132                  MOV     ESCN,(R1)+
2037  007442  013721  001762                  MOV     CHOM,(R1)+      ;CURSOR HOME
2038  007446  013721  002054                  MOV     ESCO,(R1)+
2039  007452  013721  002056                  MOV     XMTAL,(R1)+     ;TRANSMIT ALL
2040  007456  012737  000005  015300          MOV     #5,XMCNT
2041  007464  052777  000100  172244          BIS     #TENA,@VXCSR    ;SET XMIT ENABLE
2042  007472  012746  000001                  MOV     #XMDNE,-(SP)    ;;PUSH #XMDNE ON STACK
2043  007476  012746  000003                  MOV     #3,-(SP)        ;;PUSH #3 ON STACK
2044  007502  004037  020620                  JSR     R0,WTBGND       ;LOOK FOR SOM OR XMIT DONE
2045  007506  000461                          BR      C2XT            ;NOT FOUND-ERROR EXIT
2046  007510  013701  014770          4$:     MOV     RBUFP,R1        ;SET UP RECEIVE FLAG
2047  007514  005037  002220                  CLR     DLAY            ;SET UP TIME OUT DELAY
2048  007520  020137  014770          40$:    CMP     R1,RBUFP        ;CHARACTER RECEIVED?
2049  007524  103411                          BLO     41$             ;YES-GO CHECK IT.
2050  007526  032737  020000  002224          BIT     #REOM,VSTAT     ;LOOK FOR END OF MESSAGE
2051  007534  001025                          BNE     C2CK            ;FOUND IT, EXIT TEST
2052  007536  005337  002220                  DEC     DLAY            ;RUN TIME OUT DELAY.
2053  007542  001366                          BNE     40$             ;AND LOOK FOR RECEIVED CH.
2054  007544  104011                          ERROR   11              ;LAST XMIT CAUSED VT61 TO HANG.
2055  007546  000420                          BR      C2CK            ;GO SEE IF ANY ERRORS STORED.
2056  007550  013737  014764  014770  41$:    MOV     RBBUF,RBUFP     ;RESET RECEIVE POINTER
2057  007556  127727  005202  000040          CMPB    @RBBUF,#40      ;CHAR EQUAL A SPACE?
2058  007564  001003                          BNE     6$              ;NOT A SPACE-MUST BE ERROR-STORE IT
2059  007566  004037  016506          5$:     JSR     R0,CPPOS        ;UPDATE CURSOR POSITION
2060  007572  000746                          BR      4$
2061  007574  022702  030536          6$:     CMP     #TCRLB+20.,R2   ;STORED 10 ERRORS?
2062  007600  101772                          BLOS    5$              ;YES-IGNORE ANY FURTHER ERRORS.
2063  007602  013722  016550                  MOV     LNRW,(R2)+      ;STORE FAILING CURSOR POSITION
2064  007606  000767                          BR      5$
2065
2066  007610  020237  015272          C2CK:   CMP     R2,TBBUF        ;ANY ERRORS STORED?
2067  007614  001416                          BEQ     C2XT            ;NO EXIT TEST
2068  007616  013701  015272                  MOV     TBBUF,R1        ;USE R1 AS ERROR POINTER
2069  007622  021137  002042          1$:     CMP     (R1),R23C79     ;CURSOR TO LOWER RIGHT?
2070  007626  001411                          BEQ     C2XT            ;YES-NOT AN ERROR.
2071  007630  104012                          ERROR   12              ;NO-ISSUE ERROR MESSAGES
2072  007632  012746  020040                  MOV     #20040,-(SP)    ;;PUSH #20040 ON STACK
2073  007636  012177  005122                  MOV     (R1)+,@RBBUF    ;LOAD FAILING POS.
2074  007642  004037  016346                  JSR     R0,CURER        ;ISSUE CURSOR ERROR
2075  007646  020102                          CMP     R1,R2           ;DONE WITH ERRORS?
2076  007650  103764                          BLO     1$              ;NO, DUMP ANOTHER.
2077  007652  000240          C2XT:   NOP                             ;EXIT TEST
2078
2079                          ;;**************************************************************
2080                                  ;ROUTINE TO INSURE PROPER OPERATION OF CARRIAGE RETURN
2081                                  ;AND LINE FEED DURING NORMAL MODE. INITIALLY THE CURSOR IS
2082                                  ;SET(VIA D.C.A.) TO ROW0,COL 20 AND A LINE FEEL IS ISSUED
2083                                  ;THE CURSOR POSITION IS THEN READ AND MUST BE ROW1,COL20.
2084                                  ;A CARRIAGE RETURN IS THEN ISSUED AND CURSOR POSITION VERIFIED
2085                                  ;TO BE ROW1,COL0.
2086
```

```
                                        ;;*****************************************************************
                                        ;;*****************************************************************
(2)  007654  000004                     TST16:  SCOPE
(1)  007656  012737  000005  001160             MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)  007664  012737  007672  001106             MOV     #NWLN,$LPADR    ;;SET SCOPE LOOP ADDRESS
                                        
2091 007672  004037  015302             NWLN:   JSR     R0,RESETV       ;RESET THE UNIT AND ENTER MAINT.MODE
2092 007676  013701  015272                     MOV     TBBUF,R1
2093 007702  004037  017714                     JSR     R0,LDBUF        ;LOAD XMIT BUFFER WITH-
2094 007706    033    131    040                .BYTE   .ESC,.Y,.R00,.C20
     007711    064
2095 007712    012    033    117                .BYTE   .LNFED,.ESC,.O,.RDCUR,.BEL,0
     007715    131    007    000
2096 007720  012737  000011  015300             MOV     #9.,XMCNT       ;SETUP TO XMIT 9 CHARACTERS
2097 007726  004037  015702                     JSR     R0,XMREC        ;GO DO IT
2098 007732  000402                            BR      30$             ;NORMAL EXIT.
2099 007734  104011                            ERROR   11              ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
2100 007736  000454                            BR      4$              ;EXIT TEST
2101 007740  023777  002136  005016  30$:     CMP     R01C20,@RBBUF   ;CHECK CURSOR POS. S/B ROW 1, COL 20.
2102 007746  001412                            BEQ     3$
2103 007750  005037  001124                     CLR     $GDDAT
2104 007754  013737  001754  001126             MOV     LNFED,$BDDAT
2105 007762  104001                            ERROR   1               ;ISSUE IT
2106 007764  013746  002136                     MOV     R01C20,-(SP)    ;;PUSH R01C20 ON STACK
2107 007770  004037  016346                     JSR     R0,CURER        ;SETUP AND ISSUE CURSOR ERROR
2108 007774  013701  015272             3$:     MOV     TBBUF,R1
2109 010000  013721  001752                     MOV     CARRT,(R1)+     ;LOAD XMIT BUFFER WITH
2110 010004  013721  002054                     MOV     ESCOI,(R1)+     ;CARRIAGE RETURN, READ CURSOR
2111 010010  013721  002050                     MOV     RDCUR,(R1)+     ;POSITION
2112 010014  012737  000004  015300             MOV     #4,XMCNT        ;SET UP TO TRANSMIT 4 CHARACTERS
2113 010022  004037  015702                     JSR     R0,XMREC        ;GO DO IT
2114 010026  000402                            BR      40$             ;NORMAL EXIT.
2115 010030  104011                            ERROR   11              ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
2116 010032  000416                            BR      4$              ;EXIT TEST
2117 010034  023777  002134  004722  40$:     CMP     R01C00,@RBBUF   ;CHECK CURSOR POS. S/B ROW1, COL 0.
2118 010042  001412                            BEQ     4$              ;EXIT TEST IF GOOD.
2119 010044  005037  001124                     CLR     $GDDAT
2120 010050  013737  001752  001126             MOV     CARRT,$BDDAT
2121 010056  104001                            ERROR   1               ;ISSUE IT
2122 010060  013746  002134                     MOV     R01C00,-(SP)    ;;PUSH R01C00 ON STACK
2123 010064  004037  016346                     JSR     R0,CURER        ;SET UP AND ISSUE CURSOR ERROR
2124 010070  000240                     4$:     NOP
2125
2126
2127                                    ;;*****************************************************************
2128
2129                                            ;ROUTINE TO VERIFY PROPER OPERATION OF ERASE TO END-OF-
2130                                            ;SCREEN. SCREEN IS WRITTEN TO 1920 INCREMENTING CHAR.
2131                                            ;ERASE TO END OF SCREEN IS THEN ISSUED AND THE
2132                                            ;ENTIRE SCREEN IS READ VERIFYING THAT IT IS ALL NULLS.
2133                                    ;;*****************************************************************
2134
2135                                    ;;*****************************************************************
(2)  010072  000004                     TST17:  SCOPE
(1)  010074  012737  000003  001160             MOV     #3,$TIMES       ;;DO 3 ITERATIONS
```

```
   (1)  010102  012737  010110  001106          MOV     #ERSE,$LPADR    ;;SET SCOPE LOOP ADDRESS
 2136
 2137
 2138  010110  004037  015302          ERSE:   JSR     R0,RESETV       ;RESET THE UNIT -SET MAINT. MODE.
 2139  010114  005077  004644                  CLR     @RBBUF          ;CLEAR THE CHECK LOCATION.
 2140  010120  004037  017254                  JSR     R0,DATSC        ;FILL THE SCREEN.
 2141  010124  013701  015272                  MOV     TBBUF,R1
 2142  010130  004037  017714                  JSR     R0,LDBUF        ;LOAD XMIT BUFFER WITH:
 2143  010134     033     110     033          .BYTE   .ESC,.CHOM,.ESC,.EOS,.ESC,.0,.XMTAL,0
       010137     112     033     117
       010142     126     000
 2144  010144  113737  002132  001125          MOVB    ESCN,$GDDAT+1
 2145  010152  113737  001774  001124          MOVB    EOS,$GDDAT      ;LOAD ERROR WITH ERASE TO EOS
 2146  010160  005037  001126                  CLR     $BDDAT
 2147  010164  005077  004574                  CLR     @RBBUF
 2148  010170  012737  000007  015300          MOV     #7,XMCNT        ;SET UP TO XMIT 7 BYTES
 2149  010176  004037  015702                  JSR     R0,XMREC        ;XMIT AND WAIT FOR REC. DONE
 2150  010202  000402                          BR      5$
 2151  010204  104011                          ERROR   11              ;ESC ERROR
 2152  010206  000413                          BR      ERSXT           ;EXIT TEST
 2153  010210  127737  004550  002170  5$:     CMPB    @RBBUF,ZERO     ;VT61 XMITTED SOM/EOM ONLY?
 2154  010216  001407                          BEQ     ERSXT           ;YES-EXIT TEST.
 2155  010220  104001                          ERROR   1               ;NO-ERASE TO END OF SCREEN
 2156  010222  004037  015660                  JSR     R0,CLREG        ;GO CLEAR ERROR STORAGE
 2157  010226  117737  004532  001126          MOVB    @RBBUF,$BDDAT
 2158  010234  104004                          ERROR   4               ;ISSUE DATA ERROR
 2159  010236  000240                  ERSXT:  NOP
 2160
 2161
 2162
 2163                                  ;;**********************************************************************
 2164                                  ;ROUTINE TO SET UP END OF PASS INDICATION.
 2165                                  ;SELF TEST(ESC P T) IS ISSUED TO THE UNIT UNDER TEST
 2166                                  ;AND AN ERROR IS ISSUED IF THE UNIT CANNOT RESPOND AFTER
 2167                                  ;SELF TEST IS COMPLETE, IF SELF TEST IS SUCCESSFUL THE
 2168                                  ;SCREEN IS WRITTEN TO 23 LINES OF INCREMENTING CHARACTERS
 2169                                  ;AND 23 LINES OF INCREMENTING CHAR. IN REVERSE VIDEO.
 2170                                  ;THE IDENT IS THEN CHECKED AND IF A COPIER IS PRESENT A
 2171                                  ;COPY SCREEN  COMMAND IS ISSUED(NOTE: THIS COMMAND WILL CAUSE
 2172                                  ;THE UNIT TO BE "BUSY" AND NOT RESPOND TO ANY FURTHER COMMANDS
 2173                                  ;UNTIL THE SCREEN HAS BEEN COMPLETELY COPIED.)
 2174                                  ;;**********************************************************************
 2175
 2176                                  ;;**********************************************************************
  (2)  010240  000004                  TST20:  SCOPE
  (1)  010242  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
  (1)  010250  012737  010256  001106          MOV     #LSTST,$LPADR   ;;SET SCOPE LOOP ADDRESS
 2177
 2178  010256                           LSTST:
  (2)  010256  013746  002170                  MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
 2179  010262  013746  002122                  MOV     TSTER,-(SP)     ;;PUSH TSTER ON STACK
 2180  010266  013746  002054                  MOV     ESCO,-(SP)      ;;PUSH ESCO ON STACK
 2181  010272  004037  013456                  JSR     R0,TESC         ;TRANSMIT IT.
 2182  010276  004037  015412                  JSR     R0,GETON        ;GO LOOK FOR A XON.
 2183  010302  000407                          BR      1$              ;VT61RESPONDED-NOT HUNG
 2184  010304  013737  001732  001124          MOV     VRCSR,$GDDAT    ;LOAD THE ADDRESS
```

```
2185  010312  013737  001742  001126          MOV     VECT,$BDDAT     ;LOAD THE VECTOR
2186  010320  104010                          ERROR   10              ;REPORT SELF TEST FAILURE
2187  010322  004037  015302          1$:     JSR     R0,RESETV       ;RESET AND SET MAINT. MODE.
2188  010326  005037  002210                  CLR     BUBCT           ;SET UP HALF-SCREEN FLAG.
2189  010332  042737  077577  002224  2$:     BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2190  010340  012737  001001  002226          MOV     #1001,BLKM      ;SET UP TO XMIT A SOM -DATA- EOM.
2191  010346  013701  015272                  MOV     TBBUF,R1        ;SET UP BEG. OF XMIT BUFFER
2192  010352  012703  000500                  MOV     #320.,R3        ;FILL BUFFER WITH INCREMENTING CHAR.
2193  010356  004037  017226                  JSR     R0,BLDINC
2194  010362  012737  001700  015300          MOV     #960.,XMCNT     ;SEND 12 LINE TO VT61
2195  010370  052777  000100  171340          BIS     #TENA,@VXCSR    ;ENABLE XMIT INTERRUPTS
2196  010376  012746  000001                  MOV     #XMDNE,-(SP)    ;;PUSH #XMDNE ON STACK
2197  010402  012746  000062                  MOV     #50.,-(SP)      ;;PUSH #50. ON STACK
2198  010406  004037  020620                  JSR     R0,WTBGND       ;LOOK FOR XMDNE.
2199  010412  000430                          BR      ENDSEL          ;NOT FOUND-EXIT.
2200  010414  005737  002210                  TST     BUBCT           ;DONE WITH SCREEN?
2201  010420  001007                          BNE     3$              ;YES-EXIT
2202  010422  012703  005430                  MOV     #SETREV,R3      ;NO-ISSUE ENTER REVERSE VIDEO
2203  010426  004037  016226                  JSR     R0,LDXMIT       ;ESCAPE SEQUENCE.
2204  010432  005237  002210                  INC     BUBCT           ;INCREMENT SCREEN HALF FLAG.
2205  010436  000735                          BR      2$              ;AND ISSUE SECOND HALF IN REV. VIDEO.
2206  010440  032737  000001  002124  3$:     BIT     #BIT00,IDENT    ;IDENT = COPIER?
2207  010446  001412                          BEQ     ENDSEL          ;NO
2208  010450  013746  002170                  MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
2209  010454  012746  000135                  MOV     #.CPYSC,-(SP)   ;;PUSH #.CPYSC ON STACK
2210  010460  013746  002132                  MOV     ESCN,-(SP)      ;;PUSH ESCN ON STACK
2211  010464  004037  013456                  JSR     R0,TESC
2212  010470  004037  017722                  JSR     R0,CKABRT       ;CHECK FOR A PERIPHERAL ABORT.
2213  010474  105737  002176          ENDSEL: TSTB    MODE            ;IF IN MAN MODE DO NOT ENTER EOP.
2214  010500  001402                          BEQ     ENDPS
2215  010502  000137  003074                  JMP     ASTRT
2216  010506  042777  000100  171216  ENDPS:  BIC     #RDENA,@VRCSR   ;CLEAR REC.INT. BEFORE NEXT UNIT SELECT.
2217                                  .SBTTL  END OF PASS ROUTINE
  (1)
  (2)                             ;;*****************************************************************
  (1)                             ;*INCREMENT THE PASS NUMBER ($PASS)
  (1)                             ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
  (1)                             ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
  (1)                             ;*IF THERES A MONITOR GO TO IT
  (1)                             ;*IF THERE ISN'T JUMP TO MODCA
  (1)
  (1)   010514                    $EOP:
  (1)   010514  000004                    SCOPE
  (1)   010516  005037  001102            CLR     $TSTNM          ;;ZERO THE TEST NUMBER
  (1)   010522  005037  001160            CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
  (1)   010526  005237  001100            INC     $PASS           ;;INCREMENT THE PASS NUMBER
  (1)   010532  042737  100000  001100    BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
  (1)   010540  005327                    DEC     (PC)+           ;;LOOP?
  (1)   010542  000001            $EOPCT: .WORD   1
  (1)   010544  003022                    BGT     $DOAGN          ;;YES
  (1)   010546  012737                    MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
  (1)   010550  000001            $ENDCT: .WORD   1
  (1)   010552  010542                    $EOPCT
  (1)   010554  104401  010621            TYPE    ,$ENDMG         ;;TYPE "END PASS #"
  (2)   010560  013746  001100            MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
  (2)   010564  104405                    TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
```

```
   (1)  010566  104401  010616                 TYPE    ,$ENULL          ;;TYPE A NULL CHARACTER
   (1)  010572  013700  000042        $GET42: MOV     @#42,R0          ;;GET MONITOR ADDRESS
   (1)  010576  001405                        BEQ     $DOAGN           ;;BRANCH IF NO MONITOR
   (1)  010600  000005                        RESET                    ;;CLEAR THE WORLD
   (1)  010602  004710                $ENDAD: JSR     PC,(R0)          ;;GO TO MONITOR
   (1)  010604  000240                        NOP                      ;;SAVE ROOM
   (1)  010606  000240                        NOP                      ;;FOR
   (1)  010610  000240                        NOP                      ;;ACT11
   (1)  010612                        $DOAGN:
   (1)  010612  000137                        JMP     @(PC)+           ;;RETURN
   (1)  010614  002530                $RTNAD: .WORD   MODCA
   (1)  010616     377     377     000 $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
   (1)  010621     015  042412  042116 $ENDMG: .ASCIZ  <15><12>/END PASS */
   (1)  010626  050040  051501  020123
   (1)  010634  000043
  2218                                 ;;*****************************************************************
  2219                                        ;ROUTINE TO ECHO THE KEYBOARD. KEYS FOR TAB,BELL,CARRIAGE
  2220                                        ;AND LINE FEED ECHO A MNEMONIC,NON-DISPLAY CHAR. ECHO OCTAL
  2221                                        ;EQUIVALENTS AND DISPLAY CHAR. ECHO THEMSELVES.
  2222                                        ;(EXAMPLES-CHAR.,SPACE, ESC, SPACE OR 037, SPACE.) A
  2223                                        ;CONTROL C (003) WILL CAUSE A TEST EXIT.
  2224
  2225                                 ;;*****************************************************************
  2226
  2227                                 ;;*****************************************************************
  (2)  010636  000004                 TST21:  SCOPE
  (1)  010640  012737  000001  001160        MOV     #1,$TIMES        ;;DO 1 ITERATION
  (1)  010646  012737  010654  001106        MOV     #KYBD,$LPADR     ;;SET SCOPE LOOP ADDRESS
  2228
  2229  010654  004037  016266        KYBD:   JSR     R0,RESPTR
  2230  010660  012702  026014                MOV     #DKYBD,R2        ;LOAD MESSAGE ADDRESS INR2
  2231  010664  004037  017322                JSR     R0,DSMES         ;DISPLAY KEYBOARD MESSAGE
  2232  010670  012703  026402                MOV     #DCNTZ,R3        ;ISSUE CONTROL C EXIT MESSAGE
  2233  010674  004037  016226                JSR     R0,LDXMIT
  2234  010700  012703  011136                MOV     #EXMAIN,R3
  2235  010704  004037  016226                JSR     R0,LDXMIT        ;ISSUE EXIT MAINTENANCE MODE.
  2236  010710  042737  077577  002224 KYSTRT: BIC     #77577,VSTAT     ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
  2237  010716  105777  020270                TSTB    @ABUFP           ;SEE IF A CHAR. RECEIVED
  2238  010722  001001                        BNE     11$              ;YES-GO PROCESS IT
  2239  010724  000001                        WAIT                     ;WAIT FOR A CH.
  2240  010726  117701  020260        11$:    MOVB    @ABUFP,R1        ;GET RAW RECEIVED DATA
  2241  010732  004037  020544                JSR     R0,EXTST         ;CHECK FOR EXIT CONDITIONS
  2242  010736  000402                        BR      10$              ;NO EXIT -CONTINUE.
  2243  010740  000137  003074                JMP     ASTRT            ;EXIT TEST 4
  2244  010744  105077  020242        10$:    CLRB    @ABUFP           ;CLEAR CHAR FROM BUFFER
  2245  010750  032737  000400  002224        BIT     #ESC,VSTAT       ;CHAR.=ESC(033)?
  2246  010756  001405                        BEQ     12$              ;NO
  2247  010760  005037  014772                CLR     ESAMB            ;YES - RESET ESC ASSEMBLY FLAG
  2248  010764  012703  026007                MOV     #DESC,R3         ;LOAD ESC MESSAGE ADDRESS
  2249  010770  000454                        BR      KYBXMT
  2250  010772  120127  000041        12$:    CMPB    R1,#41           ;CHAR. LESS THAN 41 OR
  2251  010776  103415                        BLO     2$               ;HIGHER THAN 176, GO ECHO
  2252  011000  120127  000176                CMPB    R1,#176          ;OCTAL EQUIVALENT
  2253  011004  101012                        BHI     2$
  2254  011006  110177  004264                MOVB    R1,@TBUFP        ;LOAD CHAR. IN XMIT BUFF.
  2255  011012  004037  016156                JSR     R0,XMIT1         ;GO XMIT IT
```

```
2256  011016  112777  000040  004252          MOVB    #40,@TBUFP      ;LOAD A SPACE
2257  011024  004037  016156                  JSR     R0,XMIT1        ;AND XMIT IT.
2258  011030  000727                          BR      KYSTRT
2259  011032  120137  001750          2$:     CMPB    R1,BEL          ;CHAR.=BELL?
2260  011036  001003                          BNE     3$
2261  011040  012703  026263                  MOV     #DBELL,R3       ;LOAD BELL MESSAGE ADDRESS
2262  011044  000426                          BR      KYBXMT
2263  011046  120137  001756          3$:     CMPB    R1,TAB          ;CHAR. =TAB?
2264  011052  001003                          BNE     4$
2265  011054  012703  026244                  MOV     #DTAB,R3        ;YES-ECHO 'TAB'
2266  011060  000420                          BR      KYBXMT
2267  011062  123701  001752          4$:     CMPB    CARRT,R1        ;CHAR.=CARRIAGE RETURN?
2268  011066  001003                          BNE     5$
2269  011070  012703  026251                  MOV     #DCR,R3         ;YES - ECHO 'C/R'.
2270  011074  000412                          BR      KYBXMT
2271  011076  120137  001754          5$:     CMPB    R1,LNFED        ;CHAR.=LINE FEED?
2272  011102  001003                          BNE     6$              ;NO CHECK FOR CONTROL Z
2273  011104  012703  026256                  MOV     #DLF,R3         ;YES - ECHO 'L/F'.
2274  011110  000404                          BR      KYBXMT
2275  011112  004037  017416          6$:     JSR     R0,BINOCT       ;CONVERT BINARY TO OCTAL
2276  011116  012703  002164                  MOV     #SVER1,R3
2277  011122  042737  077577  002224  KYBXMT: BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2278  011130  004037  016226                  JSR     R0,LDXMIT       ;GO XMIT BUFFER
2279  011134  000665                          BR      KYSTRT          ;WAIT FOR NEXT CHAR.
2280
2281                                          ;SEQUENCE TO EXIT MAINTENANCE MODE.
2282  011136      033      117      141  EXMAIN: .BYTE   .ESC,.O,.DMAIN,0
      011141      000
2283
2284                                  ;;**********************************************************************
2285                                  ;ROUTINE TO UTILIZE THE VT61 AS A PRINTER CONTROLLER.
2286                                  ;AFTER TEST MESSAGE IS DISPLAYED, THE TEST WAITS
2287                                  ;FOR A C/R BEFORE ACTUALLY ENTERING TEST. A PATTERN
2288                                  ;OF INCREMENTING, ROLLING CHAR. WILL BE OUTPUTTED UNTIL A
2289                                  ;CONTROL C(003) IS RECEIVED.
2290
2291                                  ;;**********************************************************************
2292
2293                                  ;;**********************************************************************
 (2)  011142  000004                  TST22:  SCOPE
 (1)  011144  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
 (1)  011152  012737  011160  001106          MOV     #TPRNT,$LPADR   ;;SET SCOPE LOOP ADDRESS
2294
2295  011160  012702  026446          TPRNT:  MOV     #DPRNT,R2       ;LOAD PRINTER MESSAGE ADDRESS
2296  011164  004037  017322                  JSR     R0,DSMES        ;AND ISSUE IT
2297  011170  012703  011136                  MOV     #EXMAIN,R3
2298  011174  004037  016226                  JSR     R0,LDXMIT       ;ISSUE EXIT MAINTENANCE MODE.
2299  011200  004037  017514                  JSR     R0,GTCR         ;GO SET CARRIAGE RETURN
2300  011204                          3$:
 (2)  011204  013746  002170                  MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
2301  011210  013746  002114                  MOV     EPNT,-(SP)      ;;PUSH EPNT ON STACK
2302  011214  013746  002132                  MOV     ESCN,-(SP)      ;;PUSH ESCN ON STACK
2303  011220  004037  013456                  JSR     R0,TESC
2304  011224  013701  015272                  MOV     TBBUF,R1        ;LOAD R1 WITH XMIT BUFFER
2305  011230  012705  000041          4$:     MOV     #41,R5          ;R5=1ST CHAR
2306  011234  042737  077577  002224  5$:     BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
```

```
2307  011242  013701  015272                MOV     TBBUF,R1
2308  011246  012703  000204                MOV     #132.,R3        ;R3= LINE WIDTH
2309  011252  004037  017232                JSR     R0,BLDINA       ;GO BUILD A SLIDING PATTERN.
2310  011256  013721  001752                MOV     CARRT,(R1)+     ;LOAD A C/R AND L/F
2311  011262  013721  001754                MOV     LNFED,(R1)+
2312  011266  012737  000206  015300        MOV     #134.,XMCNT     ;SET UP TO XMIT BY BYTES.
2313  011274  052777  000100  170434        BIS     #TENA,@VXCSR
2314  011302  032737  000001  002224        BIT     #XMDNE,VSTAT    ;WAIT FOR XMIT DONE
2315  011310  001774                        BEQ     .-6
2316  011312  004037  020544                JSR     R0,EXTST        ;CHECK FOR EXIT REQUEST.
2317  011316  000402                        BR      6$              ;NO-CONTINUE
2318  011320  000137  003074                JMP     ASTRT           ;YES-EXIT TEST!!
2319  011324  004037  017722        6$:     JSR     R0,CKABRT       ;CHECK FOR A PERIPHERAL ABORT.
2320  011330  122705  000177                CMPB    #177,R5         ;EXCEEDED PATT. LIMIT?
2321  011334  001337                        BNE     5$              ;NO
2322  011336  000734                        BR      4$              ;YES RESET IT
2323
2324                                  ;;**********************************************************************
2325
2326                                  ;ROUTINE TO LOOP DATA/COMMANDS FROM THE VT61 BACK TO
2327                                  ;THE VT61. DATA TRANSMISSIONS RESULTING FROM A ESC
2328                                  ;SEQUENCE WILL ALSO BE LOOPED AND WILL ENTER THE SCREEN
2329                                  ;AT THE CURSOR POSITION.THIS TEST CAN BE USED TO SIMULATE,
2330                                  ;OR CREATE, SPECIFIC SCREEN PATTERNS AND OPERATIONS.
2331                                  ;;**********************************************************************
2332
2333                                  ;;**********************************************************************
  (2)  011340  000004           TST23:  SCOPE
  (1)  011342  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
  (1)  011350  012737  011356  001106          MOV     #LPTST,$LPADR   ;;SET SCOPE LOOP ADDRESS
2334
2335  011356  004037  016266        LPTST:  JSR     R0,RESPTR       ;RESET POINTERS
2336  011362  012702  026271                MOV     #DLOOP,R2       ;LOAD LOOP MESSAGE ADDRESS
2337  011366  004037  017322                JSR     R0,DSMES        ;DISPLAY IT
2338  011372  012703  011136                MOV     #EXMAIN,R3
2339  011376  004037  016226                JSR     R0,LDXMIT       ;ISSUE EXIT MAINTENANCE MODE.
2340  011402  004037  020314                JSR     R0,LOOP         ;GO LOOP VT61
2341  011406  000137  003074                JMP     ASTRT           ;ENTER MAN MODE VIA SCOPE ROUTINE.
2342
2343                                  ;;**********************************************************************
2344
2345                                  ;PRODUCTION KEYBOARD TEST.  ALL KEYS MUST BE DEPRESSED
2346                                  ;IN THE SEQUENCE INDICATED ON THE SCREEN.  ALL ERRORS
2347                                  ;OR MISTAKES ARE DISPLAYED IN OCTAL POSITIONAL FORMAT AND THE
2348                                  ;CORRECT KEY POSITION IN THE ROW  IS DISPLAYED IN DECIMAL.
2349                                  ;THIS TEST IS RUN IN MAINTENANCE MODE, THEREFORE THE KEYS
2350                                  ;WILL ECHO THEIR POSITION ,NOT THEIR INDICATED MNEMONIC. 10
2351                                  ;ERRORS WILL CAUSE AN AUTOMATIC EXIT FROM TEST.
2352                                  ;;**********************************************************************
2353
2354                                  ;;**********************************************************************
  (2)  011412  000004           TST24:  SCOPE
  (1)  011414  012737  000001  001160          MOV     #1,$TIMES       ;;DO 1 ITERATION
  (1)  011422  012737  011430  001106          MOV     #PDKBD,$LPADR   ;;SET SCOPE LOOP ADDRESS
2355
2356
```

```
                                PDKBD:  MOV     #DKBD   ,R2
                                        JSR     R0      ,DSMES  ;DISPLAY KEYBOARD TEST MESSAGE.
                                        CLR     BUBCT           ;CLEAR ERROR COUNT LOCATION.
                                        CLR     R5

                                DOAROW: MOV     DTTBL(R5),R4    ;SET UP 'GOOD' CHAR. POINTER
                                        MOV     MSTBL(R5),R3
                                        BEQ     FEXIT           ;MESSAGE WAS ZERO-EXIT.
                                        BMI     CLMAIN          ;IF MESSAGE IS -1,CLEAR MAINT. MODE.
                                        JSR     R0,LDXMIT       ;ISSUE 'ROW OR FUNCTION' MESSAGE.
                                        JSR     R0,CKKBD        ;GO CHECK IT.
                                        CMPB    BUBCT,#10.      ;TEN ERROR EXIT?
                                        BLO     1$              ;NO-CONTINUE.
                                        BR      FEXIT           ;YES-EXIT TEST.
                                1$:     TST     (R5)+           ;INCREMENT OFFSET.
                                        BR      DOAROW          ;NO-DO NEXT ROW/FUNCTION.
                                FEXIT:  MOV     #DEXT,R2
                                        JSR     R0,DSMES        ;ISSUE EXIT MESSAGE
                                        JMP     ASTRT
                                CLMAIN: MOV     #RSMAIN,R3      ;SET UP TO EXIT MAINT. MODE.
                                        JSR     R0,LDXMIT
                                        TST     (R5)+           ;INCREMENT OFFSET.
                                        BR      DOAROW          ;NOW TEST CONTROL AND SHIFT FUNCTIONS.
011540  033     117     141     RSMAIN: .BYTE   .ESC,.O,.DMAIN,0
011543  000

                                                ;TABLE OF MESSAGE ADDRESSES.

011544  027067  027174  027231  MSTBL:  .WORD   DTOP,DSEC,DTHRD,DBOT
011552  027360
011554  027436  027462  177777          .WORD   DSPCE,DKPD,-1,DCONT,DLSHFT,DRSHFT,0
011562  027310  027015  027121
011570  000000

011572  027663  027704  027724  DTTBL:  .WORD   ROW1,ROW2,ROW3,ROW4,SPCB
011600  027742  027764
011604  027766  000000  027760          .WORD   KYPD,0,CNTRA,SHFTA,SHFTA
011612  027762  027762

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

011616                          SETA:
 (1)                            .SBTTL  INITIALIZE THE COMMON TAGS
 (1)                            ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
 (1)    011616  012706  001100          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 (1)    011622  005026                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 (1)    011624  022706  001140          CMP     #SWR,R6 ;;DONE?
 (1)    011630  001374                  BNE     .-6             ;;LOOP BACK IF NO
 (1)    011632  012706  001100          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 (1)                            ;;INITIALIZE A FEW VECTORS
 (1)    011636  012737  020734  000020  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 (1)    011644  012737  000340  000022  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
 (1)    011652  012737  021210  000030  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
```

```
  (1)  011660  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  011666  012737  022602  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  011674  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  011702  012737  022424  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  (1)  011710  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
  (1)  011716  013737  010550  010542          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
  (1)  011724  005037  001160                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
  (1)  011730  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
  (1)  011734  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
  (1)  011742  012737  011742  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  (1)  011750  012737  011750  001110          MOV     #.,$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)  011756  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
  (2)  011762  012737  012016  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
  (2)  011770  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)  011776  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  (2)  012004  022777  177777  167126          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  (2)  012012  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)  012014  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
  (2)  012016  012716  012024          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  (2)  012022  000002                          RTI
  (2)  012024  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
  (2)  012032  012737  000174  001142          MOV     #DISPREG,DISPLAY
  (2)  012040  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
  (1)
2396                                   .SBTTL  TYPE PROGRAM NAME
  (1)                                  ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
  (1)  012044  005227  177777                  INC     #-1             ;;FIRST TIME?
  (1)  012050  001017                          BNE     67$             ;;BRANCH IF NO
  (1)  012052  022737  010602  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
  (1)  012060  001413                          BEQ     67$             ;;BRANCH IF YES
  (1)  012062  104401  012070                  TYPE    ,68$            ;;TYPE ASCIZ STRING
  (1)  012066  000410                          BR      67$             ;;GET OVER THE ASCIZ
  (1)                                  ;;68$:  .ASCIZ  <CRLF>*MD-11-DZVTH-B*<CRLF>
  (1)  012110                          67$:
2397   012110  104401  022722                  TYPE    ,STUPM  ;ISSUE SET-UP MESSAGE.
2398   012114  012737  012134  000010          MOV     #TRPA,@#10      ;AND VECTOR
2399   012122  000230                          SPL     0               ;PROCESSOR IS 11/45?
2400   012124  012737  000004  017222          MOV     #4,PMULT        ;YES-DELAY MULTIPLIER = 4
2401   012132  000416                          BR      RTRP
2402
2403   012134  022626                  TRPA:   POP2SP                  ;NO
2404   012136  012737  012160  000010          MOV     #TRPB,@#10      ;RELOAD TRAP ADDRESS
2405   012144  006737  002162                  SXT     CHRD            ;PROCESSOR IS 11/40 OR 35?
2406   012150  012737  000002  017222          MOV     #2,PMULT        ;YES-DELAY MULTIPLIER=2
2407   012156  000404                          BR      RTRP
2408
2409   012160  022626                  TRPB:   POP2SP
2410   012162  012737  000001  017222          MOV     #1,PMULT        ;PROCESSOR MUST BE 11/05
2411   012170  012737  000012  000010  RTRP:   MOV     #12,@#10        ;RESTORE TRAP CATCHER
2412   012176  105737  002176                  TSTB    MODE            ;CHECK MODE FOR CORRECT EXIT.
2413   012202  001402                          BEQ     70$
2414   012204  000137  002276                  JMP     MANSA           ;EXIT TO MANUAL SELECT
2415   012210  000137  002242          70$:    JMP     AUTOA           ;EXIT TO AUTO MODE.
```

```
                                ;;*****************************************************************
                                ;SUBROUTINE TO VERIFY ADDRESS AS THEY ARE ENTERED IN
                                ;MANUAL SELECT MODE.
                                ;;*****************************************************************
012214  020227  160000  TMNAD:  CMP     R2,#160000      ;INSURE ADDRESS IS IN RANGE.
012220  103407          BLO     BDEXT           ;ITS NOT, TYPE A ? AND EXIT.
012222  012737  012236  000004  MOV     #BDEXTA,@#4     ;SET UP TIME OUT TRAP.
012230  005712          TST     @R2             ;CHECK THE ADDRESS.
012232  000240          NOP
012234  000405          BR      ALEXT           ;ITS OK, USE A NORMAL EXIT.
012236  022626  BDEXTA: POP2SP                  ;ADDRESS TRAPPED- PURGE IT, TYPE A
012240  104401  026662  BDEXT:  TYPE    ,QMRK    ;QUESTION MARK, RESTORE THE TRAP
012244  012700  002312          MOV     #BLDADD,R0      ;LOCATION, AND EXIT TO GET NEXT ENTRY.
012250  012737  000006  000004  ALEXT:  MOV     #6,@#4
012256  000200          RTS     R0

                                ;;*****************************************************************
                                ;THIS ROUTINE MAPS ALL POSSIBLE DL11 ADDRESSES AND STORES
                                ;THEM IN A TABLE (INTAB). ALL ADDRESSES WHICH DO NOT
                                ;RESULT IN TIMEOUTS ARE STORED.
                                ;;*****************************************************************

012260  012701  000300  TRPVEC: MOV     #300,R1         ;START AT BEG. OF FLOATING VECTORS
012264  012702  000302          MOV     #302,R2         ;
012270  012703  000004          MOV     #4,R3           ;R3 CONTAINS IOT TRAP INST.
012274  010221          1$:     MOV     R2,(R1)+        ;START LOADING ADDRESSES
012276  010321          MOV     R3,(R1)+        ;LOAD THE TRAP
012300  062702  000004          ADD     #4,R2           ;ASSUME 4 REGISTERS PER INTERFACE
012304  020127  001000          CMP     R1,#1000        ;DONE?
012310  002771          BLT     1$              ;NO CONTINUE LOADING TRAPS
012312  005037  000006          CLR     @#6
012316  012737  012356  000004  MOV     #TPENT,@#4      ;SET UP TIME-OUT TRAP ADDRESS
012324  005001          CLR     R1              ;CLEAR THE TABLE POINTER
012326  012705  001652          MOV     #INTAB,R5       ;R5=DESTINATION TABLE
012332  016102  001716  FADD:   MOV     STRTAB(R1),R2   ;PUT THE ADDRESS TO BE TESTED IN R2
012336  026102  001724  TRPE:   CMP     ENDTAB(R1),R2   ;HAVE WE EXCEEDED END OF TABLE ADDRESS?
012342  103407          BLO     TBLCK           ;YES GET NEXT BASE ADDRESS.
012344  005712          TST     @R2             ;ADDRESS THE DEVICE IF POSSIBLE
012346  010225          MOV     R2,(R5)+        ;IF WE GOT THIS FAR THERE IS A DEVICE THERE-SAVE IT
012350  062702  000010  FADD1:  ADD     #10,R2          ;INCREMENT TO THE NEXT POSSIBLE ADDRESS
012354  000770          BR      TRPE            ;GO TEST THE NEXT ADDRESS
012356  022626  TPENT:  POP2SP                  ;RESTORE THE STACK AND TEST
012360  000773          BR      FADD1           ;NEXT ADDRESS
012362  005721  TBLCK:  TST     (R1)+           ;BUMP AREA COUNTER BY 2.
012364  032701  000004          BIT     #BIT02,R1       ;SEE IF BOTH DL11 AREAS CHECKED.
012370  001760          BEQ     FADD            ;NO-GO CHECK THE OTHER AREA
012372  005015          CLR     (R5)            ;SET UP TABLE TERMINATOR OF  ZEROS.
012374  000200          RTS     R0
                                ;;*****************************************************************
                                ;THIS ROUTINE WILL INSURE THAT THE DEVICE(DL11)
                                ;WILL INTERRUPT WHEN XMIT INT. ENABLE BIT IS SET.
                                ;;*****************************************************************

012376  005046  CDEV:   CLR     -(SP)           ;CLEAR THE PSW,LSI11 STYLE.
```

```
2472  012400  012746  012406                  MOV     #100$,-(SP)
2473  012404  000002                          RTI
2474  012406  012737  000004  000004  100$:   MOV     #4,@#4          ;INSTALL IOT TRAP INST. AT LOCATION 4.
2475  012414  012737  012512  000020          MOV     #TDEV,@#IOTVEC  ;SET UP IOT TRAP EXIT ADDRESS
2476  012422  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;SET PSW TO 7-ALLOW NO OTHER INTERRUPTS
2477  012430  000005                          RESET                   ;INSURE ALL XMIT FLAGS HIGH.
2478  012432  012703  001552                  MOV     #VVECT,R3       ;VECTOR STORAGE ADDRESS SET
2479  012436  012702  001652                  MOV     #INTAB,R2       ;PRIMARY DEVICE TABLE ADDRESS SET
2480  012442  012705  001612                  MOV     #DLTBL,R5       ;FIN DEVICE TABLE ADDRESS SET.
2481  012446  012701  001732          CDEVA:  MOV     #VRCSR,R1       ;VT61 DEVICE ADDRESS SET.
2482  012452  005712                          TST     (R2)            ;CHECKED ALL DEVICES?
2483  012454  001506                          BEQ     AOUT            ;YES-EXIT
2484  012456  100403                          BMI     1$              ;INSURE ADDRESS IS IN PROPER RANGE(17XXXX)
2485
2486  012460  062702  000002                  ADD     #2,R2           ;ADDRESS IS DEFINITELY NOT GOOD -PURGE
2487  012464  000770                          BR      CDEVA           ;AND LOOK FOR ANOTHER.
2488  012466  004037  013174          1$:     JSR     R0,LDADD        ;LOAD NEXT ADDRESSES TO BE CHECKED
2489  012472  012701  001200                  MOV     #1200,R1        ;NOW USE R1 AS FAILSAFE COUNTER
2490  012476  052777  000100  167232          BIS     #TENA,@VXCSR    ;SET XMIT ENABLE
2491  012504  005301                          DEC     R1              ;IF DEVICE DOES NOT INTERRUPT WITHIN
2492  012506  001376                          BNE     .-2             ;APPROX. 200US IT IS NOT A DL11.
2493  012510  000756                          BR      CDEVA           ;THEREFORE, GO TRY ANOTHER DEVICE.
2494  012512  042777  000100  167216  TDEV:   BIC     #TENA,@VXCSR    ;CLEAR XMIT ENABLE.
2495  012520  162716  000010                  SUB     #10,(R6)        ;RESET TO RECEIVER VECTOR ADDRESS
2496  012524  012613                          MOV     (R6)+,(R3)      ;STORE IT IN VECTOR TABLE(VVECT).
2497  012526  005726                          TST     (R6)+           ;POP THE OLD PSW AND DISCARD
2498  012530  022626                          POP2SP                  ;POP THE ADD. AND PSW PRIOR TO INTERRUPT.
2499                          ;;**************************************************************
2500                          ;THIS ROUTINE IS A QUICK TEST OF ANY DL11 ENCOUNTERED
2501                          ;A DATA PATTERN WILL BE RUN ON ALL ENTRIES IN INTAB
2502                          ;;**************************************************************
2503  012532  005046                          CLR     -(SP)           ;CLEAR THE PSW,LSI11 STYLE.
2504  012534  012746  012542                  MOV     #100$,-(SP)
2505  012540  000002                          RTI
2506  012542  012301                  100$:   MOV     (R3)+,R1        ;GET THE RECEIVE VECTOR ADDRESS
2507  012544  012721  014004                  MOV     #RECAD,(R1)+    ;AND STORE SAME.
2508  012550  012721  000340                  MOV     #340,(R1)+      ;SET RECEIVE PSW TO 7.
2509  012554  012721  014066                  MOV     #TSMAD,(R1)+    ;STORE THE XMIT VECTOR ADDRESS
2510  012560  012711  000340                  MOV     #340,(R1)       ;SET XMIT PSW TO 7.
2511  012564  012704  000001                  MOV     #BIT00,R4       ;R4 IS NOW DATA PATTERN OF 1.
2512  012570  005001                          CLR     R1              ;SET UP FAILSAFE DELAY.
2513  012572  052777  000100  167132          BIS     #RDENA,@VRCSR   ;SET RECEIVE ENABLE.
2514  012600  052777  000104  167130          BIS     #TCOMB,@VXCSR   ;ENABLE XMIT INT. AND MAINTENACE .
2515  012606  105704                  1$:     TSTB    R4              ;XMIT PATTERN COMPLETE?
2516  012610  001423                          BEQ     GDAD            ;YES GO STORE THIS ADDRESS
2517  012612  005301                          DEC     R1              ;CYCLE TIMEOUT DELAY
2518  012614  001374                          BNE     1$              ;NOT YET 'TIMEOUT',KEEP CYCLING.
2519  012616  162703  000002                  SUB     #2,R3           ;RESET VECTOR POINTER
2520  012622  042777  000104  167106          BIC     #TCOMB,@VXCSR   ;CLEAR XMIT AND RECEIVE INT. ENABLES.
2521  012630  042777  000100  167074          BIC     #RDENA,@VRCSR
2522  012636  104401  023200                  TYPE    ,DLERR          ;ISSUE DL11 FAILURE MESSAGE.
2523  012642  013746  001732                  MOV     VRCSR,-(SP)     ;;SAVE VRCSR FOR TYPEOUT
  (1)                                                                 ;;TYPE BD. ADDRESS
  (1)   012646  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
  (1)   012650     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
  (1)   012651     001                          .BYTE   1               ;;TYPE LEADING ZEROS
```

```
2524  012652  104401  001171                    TYPE    $CRLF
2525  012656  000673                            BR      CDEVA           ;GO TRY ANOTHER SET OF ADDRESSES.
2526  012660  013725  001732            GDAD:   MOV     VRCSR,(R5)+     ;SAVE GOOD ADDRESS IN DL TABLE
2527  012664  005077  167044                    CLR     @VRBUF          ;CLEAR ANY RECEIVE FLAG STILL SET.
2528  012670  000666                            BR      CDEVA           ;;CHECK ANOTHER DL11
2529  012672  005015                    AOUT:   CLR     (R5)            ;SET A ZERO TABLE TERMINTATOR.
2530  012674  012737  000006  000004            MOV     #6,@#4          ;RESTORE LOCATION 4 TO HALT CONDITION
2531  012702  005037  000006                    CLR     @#6             ;TO CATCH ERRORS AND ILLEGAL INTERRUPTS.
2532  012706  012737  020734  000020            MOV     #$SCOPE,@#IOTVEC        ;RELOAD IOT VECTOR FOR SCOPE
2533  012714  012737  000340  000022            MOV     #340,@#IOTVEC+2 ;LOOP.
2534  012722  012701  000300                    MOV     #300,R1
2535  012726  012702  000302                    MOV     #302,R2
2536  012732  010221                    1$:     MOV     R2,(R1)+
2537  012734  005021                            CLR     (R1)+           ;RESTORE HALTS TO ALL LOCATIONS CONTAINING IOTS
2538  012736  062702  000004                    ADD     #4,R2
2539  012742  020127  001000                    CMP     R1,#1000        ;TO LOCATION 1000
2540  012746  103771                            BLO     1$
2541  012750  000005                            RESET                   ;CLEAR ALL FLAGS
2542  012752  000200                            RTS     R0
2543
2544                                    ;;**************************************************************
2545                                            ;INITIALIZATION ROUTINE FOR AUTO SELECTION. THIS ROUTINE
2546                                            ;WILL INSURE THAT ALL DL11S IN DLTBL HAVE A VT61 CONNECTED
2547                                            ;ALL UNITS WHICH CANNOT CORRECTLY RESPOND WILL BE PURGED.
2548                                    ;;**************************************************************
2549
2550  012754  012702  001612            INITA:  MOV     #DLTBL,R2       ;R2 POINTS TO DL11 ADDRESS TABLE
2551  012760  012703  001552                    MOV     #VVECT,R3       ;R3 POINTS TO DL11 VECTOR
2552  012764  012701  001732            11$:    MOV     #VRCSR,R1       ;POINTER TO VT61 DL11
2553  012770  005712                            TST     (R2)            ;SEE IF ALL CHECKED
2554  012772  001447                            BEQ     INTXT           ;YES-EXIT
2555  012774  004037  013174                    JSR     R0,LDADD        ;NO-GO LOAD THE ADDRESSES
2556  013000  062703  000002                    ADD     #2,R3           ;UPDATE VECTOR COUNT
2557  013004  004037  015456                    JSR     R0,ZFLAG        ;ISSUE ESCZ AND LOOK FOR RESPONSE.
2558  013010                            2$:
 (2)  013010  012637  002162                    MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
2559  013014  100414                            BMI     5$              ;TIMEOUT OCCURRED NO CHARACTER
2560  013016  123727  002162  000140            CMPB    CHRD,#140       ;CHECK IDENT FOR VT61 IDENTIFIERS
2561  013024  103410                            BLO     5$              ;NOT A VT61-SET UP TO PURGE ADDRESS
2562  013026  123727  002162  000172            CMPB    CHRD,#172       ;IDENTS ARE SMALL A THRU Z
2563  013034  101004                            BHI     5$              ;NOT A VT61-PURGE
2564  013036                            4$:
 (2)  013036  012637  002162                    MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
2565  013042  001375                            BNE     4$
2566  013044  000747                            BR      11$             ;TEST ANOTHER ADDRESS
2567  013046                            5$:
 (2)  013046  012637  002162                    MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
2568  013052  001375                            BNE     5$
2569  013054  162702  000002                    SUB     #2,R2           ;RESET ADDRESS AND VECTOR POINTERS
2570  013060  162703  000002                    SUB     #2,R3
2571  013064  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2572  013066  012746  000001                    MOV     #1,-(SP)        ;;PUSH #1 ON STACK
2573  013072  004037  013404                    JSR     R0,BBLUP
2574  013076  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2575  013100  012746  000001                    MOV     #1,-(SP)        ;;PUSH #1 ON STACK
2576  013104  004037  013404                    JSR     R0,BBLUP
```

```
2577  013110  000725                        BR      11$             ;TRY ANOTHER DL11 ADDRESS.
2578
2579  013112  005737  001612        INTXT:  TST     DLTBL           ;CHECK TO INSURE GOOD ADDRESSES
2580  013116  001012                BNE     EXINT           ;YES-GO TO NEXT TEST
2581  013120  104401  023107                TYPE    ,NOVT           ;NO-ISSUE NO VT61 MESSAGE.
2582  013124  012737  005670  017224        MOV     #3000.,DCOUNT   ;SET DELAY TO 30 SEC.
2583  013132  004037  017162                JSR     R0,DELAY        ;AND DO IT.
2584  013136  062700  000004                ADD     #4,R0           ;SET UP 'NO VT61 FOUND' EXIT
2585  013142  000200                        RTS     R0
2586  013144  012702  001612        EXINT:  MOV     #DLTBL,R2       ;LOAD AND ISSUE GOOD ADDRESSES
2587  013150  104401  023036                TYPE    ,DUNTST         ;OF RESPONSIVE VT61S.
2588  013154                        1$:
 (1)  013154  012246                        MOV     (R2)+,-(SP)     ;;SAVE (R2)+ FOR TYPEOUT
 (1)                                        ;;TYPE AN ADDRESS
 (1)  013156  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)  013160     006                        .BYTE   6               ;;TYPE 6 DIGIT(S)
 (1)  013161     001                        .BYTE   1               ;;TYPE LEADING ZEROS
2589  013162  104401  001171                TYPE    ,$CRLF
2590  013166  005712                        TST     (R2)            ;AT END OF GOOD UNITS?
2591  013170  001371                        BNE     1$              ;NO PRINT ANOTHER ADDRESS.
2592  013172  000200                        RTS     R0
2593                          ;;*************************************************************
2594                                  ;SUBOUTINE TO LOAD 4 ADRESSES FROM THE LOCATION AT (R2).
2595                                  ;TO 4 LOCATION POINTED TO BY R1(TO VXBUFF+2).ROUTINE USES R4 AS
2596                                  ;WORK REG AND EXITSD WITH R2 INCREMENTED BY 2.
2597                          ;;*************************************************************
2598
2599  013174  012204        LDADD:  MOV     (R2)+,R4        ;LOAD THE ADDRESS
2600  013176  010421        1$:     MOV     R4,(R1)+        ;STORE AN ADDRESS
2601  013200  062704  000002        ADD     #2,R4           ;INCREMENT ADDRESS
2602  013204  020127  001742        CMP     R1,#VXBUF+2     ;LOADED 4?
2603  013210  002772                BLT     1$              ;NO LOAD ANOTHER
2604  013212  000200                RTS     R0              ;YES-EXIT
2605
2606                          ;;*************************************************************
2607                                  ;ROUTINE TO RECEIVE CHARACTER(S).  ENTERED WITH
2608                                  ;NUMBER OF CHARACTERS TO RECEIVE ON THE STACK.
2609                                  ;ROUTINE EXITS WITH CARACTER(S) ON STACK.  IF A
2610                                  ;PROGRAM TIME-OUT (100 M.S.) OCCURS BEFORE A CHARACTER
2611                                  ;IS RECEIVED ROUTINE EXITS WITH -1 ON STACK. FORMAT
2612                                  ;FOR DATA IS (BYTE2, BYTE1) ETC.  A WORD OF ZEROS TERMINATES
2613                                  ;DATA STRING ON THE STACK. SOM/EOM, IF SENT, ARE RECEIVED
2614                                  ;BUT NOT STORED.
2615
2616                          ;;*************************************************************
2617
2618  013214                RECTM:
 (2)  013214  012637  002222                MOV     (SP)+,ROSVE     ;;POP STACK INTO ROSVE
2619  013220  012637  002210                MOV     (SP)+,@#BUBCT   ;;POP STACK INTO @#BUBCT
2620  013224  013746  002170                MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
2621  013230  005037  002162        1$:     CLR     CHRD            ;CLEAR CHARACTER STORAGE LOCATION.
2622  013234  005037  002220                CLR     DLAY            ;SET UP FAILSAFE DELAY
2623  013240  032777  000200  166464  3$:   BIT     #RECDN,@VRCSR   ;SEE IF DONE FLAG SET
2624  013246  001007                        BNE     4$
2625  013250  005337  002220                DEC     DLAY            ;DECREMENT FAILSAFE CNTR.
2626  013254  001371                        BNE     3$              ;NOT AT ZERO-CONTINUE WAITING.
```

```
2627  013256  012737  177777  002162  31$:    MOV     #-1,CHRD        ;SET UP FOR FAILSAFE EXIT.
2628  013264  000442                          BR      RECEX           ;EXIT ROUTINE.
2629  013266  117737  166442  002162  4$:     MOVB    @VRBUF,CHRD     ;STORE THIS CHARACTER.
2630  013274  042737  000200  002162          BIC     #200,CHRD       ;STRIP PARITY BIT.
2631  013302  122737  000057  002162          CMPB    #SLSH,CHRD      ;RECEIVED A IDENT SLASH(57)?
2632  013310  001007                          BNE     41$             ;NO-STORE A CHARACTER.
2633  013312  105337  002211                  DECB    BUBCT+1         ;DECREMENT ALLOWABLE SLASH COUNT.
2634  013316  001757                          BEQ     31$             ;COUNT EQUAL ZERO-SET UP ERROR EXIT.
2635  013320  123727  002211  000213          CMPB    BUBCT+1,#139.   ;RECEIVED FIRST SLASH?
2636  013326  103740                          BLO     1$              ;YES-IGNORE THIS ONE.
2637  013330  122737  000002  002162  41$:    CMPB    #SOM,CHRD       ;IS CHAR. ACTUALLY SOM?
2638  013336  001003                          BNE     5$              ;NO
2639  013340  105237  002210                  INCB    BUBCT           ;YES -SET UP TO RECEIVE EOM ALSO
2640  013344  000731                          BR      1$              ;AND RECEIVE NEXT CHAR.
2641  013346  122737  000004  002162  5$:     CMPB    #EOM,CHRD       ;CHAR. = EOM?
2642  013354  001410                          BEQ     RECEXA          ;YES- DO NOT PUSH IT ON STACK
2643  013356  105337  002210                  DECB    BUBCT           ;DECREMENT CHARACTER COUNT.
2644  013362  001403                          BEQ     RECEX           ;COUNT=0. EXIT WERE DONE.
2645  013364  013746  002162                  MOV     CHRD,-(SP)      ;;PUSH CHRD ON STACK
2646  013370  000717                          BR      1$              ;GO READ AGAIN.
2647
2648  013372                          RECEX:
 (2)  013372  013746  002162                  MOV     CHRD,-(SP)      ;;PUSH CHRD ON STACK
2649  013376                          RECEXA:
 (2)  013376  013746  002222                  MOV     ROSVE,-(SP)     ;;PUSH ROSVE ON STACK
2650  013402  000200                          RTS     R0
2651
2652
2653                            ;;**************************************************************
2654                                    ;THIS ROUTINE WILL 'BUBBLE UP' XX WORDS TO
2655                                    ;ELIMINATE NON-RESPONSIVE ADDRESSES.  ENTERED
2656                                    ;WITH ADDRESS TO BE 'BUBBLED' TO ON THE STACK. LOCATIONS
2657                                    ;ELIMINATED WILL BE FILLED WITH ZEROS.  THE STACK MUST ALSO
2658                                    ;BE LOADED WITH THE NUMBER OF POSITIONS TO BUBBLE.
2659                            ;;**************************************************************
2660
2661  013404                          BBLUP:
 (2)  013404  012637  002222                  MOV     (SP)+,ROSVE     ;;POP STACK INTO ROSVE
2662  013410  012637  002210                  MOV     (SP)+,BUBCT     ;;POP STACK INTO BUBCT
2663  013414  012637  002206                  MOV     (SP)+,TOADD     ;;POP STACK INTO TOADD
2664  013420  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2665  013422  013704  002206          2$:     MOV     @#TOADD,R4      ;PUT LAST GOOD DL11 ADDRESS IN R4
2666  013426  012437  002162                  MOV     (R4)+,CHRD      ;MOVE NEXT WORD TO CHRD FOR STORAGE
2667  013432  012464  177774          1$:     MOV     (R4)+,-4(R4)    ;BUBBLE UP DATA.
2668  013436  001375                          BNE     1$              ;BUBBLE UNTIL ZERO BYTE MOVED.
2669  013440  005337  002210                  DEC     BUBCT           ;SUBTRACT ONE FROM BUBBLE COUNT.
2670  013444  001366                          BNE     2$              ;IF BUBBLE COUNT NOT ZERO - DO AGAIN.
2671  013446                          3$:
 (2)  013446  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
2672  013450  013746  002222                  MOV     ROSVE,-(SP)     ;;PUSH ROSVE ON STACK
2673  013454  000200                          RTS     R0              ;YES-EXIT
2674
2675                            ;;**************************************************************
2676                                    ;THIS ROUTINE OUTPUTS THE ESC SEQNENCE FOUND ON
2677                                    ;THE STACK. A WORD OF ZEROS MUST TERMINATE THE SEQUENCE.
2678                                    ;FORMAT FOR STACK WORD IS SEQ-ESC, IE-XXX033.
```

```
2679                                    ;;**********************************************************************
2680
2681  013456                            TESC:
 (2)  013456  012637  002222                    MOV     (SP)+,ROSVE     ;;POP STACK INTO ROSVE
2682  013462  010437  002210                    MOV     R4,BUBCT        ;SAVE R4.
2683  013466  112777  000002  166244            MOVB    #SOM,@VXBUF     ;SEND A START OF MESSAGE.
2684  013474  012705  177777            1$:     MOV     #-1,R5          ;ALL ONES THO CHECK LOCATION.
2685  013500  012604                            MOV     (R6)+,R4        ;GET COMMAND FROM STACK.
2686  013502  001415                            BEQ     3$              ;IF ZERO TERMINATOR FOUND-EXIT.
2687  013504  110405                            MOVB    R4,R5           ;LOAD CHECK BYTE.
2688  013506  105704                    2$:     TSTB    R4              ;CHECK BYTE FOR A ZERO.
2689  013510  001406                            BEQ     20$             ;IF ZERO-DO NOT XMIT IT.
2690  013512  032777  000200  166216            BIT     #TRDY,@VXCSR
2691  013520  001774                            BEQ     .-6             ;WAIT FOR XMIT READY BIT
2692  013522  110477  166212                    MOVB    R4,@VXBUF       ;XMMIT A BYTE.
2693  013526  000304                    20$:    SWAB    R4              ;GET THE OTHER BYTE .
2694  013530  120405                            CMPB    R4,R5           ;IF GOOD COMPARE WE HAVE CHECKED BOTH
2695  013532  001760                            BEQ     1$              ;BYTES SO POP ANOTHER WORD.
2696  013534  000764                            BR      2$              ;GO XMIT ANOTHER BYTE
2697  013536  032777  000200  166172    3$:     BIT     #TRDY,@VXCSR    ;SEE IF READY SET
2698  013544  001774                            BEQ     .-6
2699  013546  012777  000004  166164            MOV     #EOM,@VXBUF     ;SEND A EOM.
2700  013554  032777  000200  166154            BIT     #TRDY,@VXCSR    ;SEE IF READY SET
2701  013562  001774                            BEQ     .-6
2702  013564  013704  002210                    MOV     BUBCT,R4        ;RESTORE R4.
2703  013570  013746  002222                    MOV     ROSVE,-(SP)     ;;PUSH ROSVE ON STACK
2704  013574  000200                            RTS     R0
2705
2706                                    ;;**********************************************************************
2707                                            ;ROUTINE TO READ A CHARACTER FROM THE CONSOLE.
2708                                            ;EXITS WITH CHARACTER ON THE STACK.
2709                                    ;;**********************************************************************
2710
2711  013576                            CONRD:
 (2)  013576  012637  002222                    MOV     (SP)+,ROSVE     ;;POP STACK INTO ROSVE
2712  013602  032777  000200  165334            BIT     #RECDN,@$TKS    ;LOOK FOR DONE BIT
2713  013610  001774                            BEQ     .-6             ;WAIT FOR IT
2714  013612  117746  165330                    MOVB    @$TKB,-(R6)     ;PUSH CHARACTER TO STACK
2715  013616  042716  000200                    BIC     #200,(R6)       ;STRIP ANY PARITY BIT.
2716  013622  013746  002222                    MOV     ROSVE,-(SP)     ;;PUSH ROSVE ON STACK
2717  013626  000200                            RTS     R0
2718
2719                                    ;;**********************************************************************
2720                                            ;MANUAL TEST SELECT MONITOR
2721                                            ;SELECTS TESTS TO BE EXECUTED FROM THOSE ENTERED IN
2722                                            ;INITIAL DIALOGE. IF TEST 377 WAS REQUESTED THE TESTS WILL
2723                                            ;REPEAT INFINITELY.
2724                                    ;;**********************************************************************
2725
2726  013630  105737  002176            MONIT:  TSTB    MODE            ;TEST MODE SWITCH
2727  013634  001012                            BNE     1$              ;MANUAL MODE
2728  013636  023737  002200  002202            CMP     FTLCNT,ALWCNT   ;COMPARE FATAL XMITS WITH ALLOWED.
2729  013644  103405                            BLO     100$            ;FATALS LESS THAN ALLOWED-CONTINUE.
2730  013646  104401  024641                    TYPE    ,DABRT          ;ISSUE ABORT MESSAGE.
2731  013652  000005                    200$:   RESET                   ;CLEAR ALL INTERFACE FLAGS.
2732  013654  000137  011616                    JMP     SETA            ;SET UP TO RESTART TEST.
```

```
2733  013660  000200                  100$:   RTS     R0              ;AUTO MODE
2734  013662  005726                  1$:     TST     (R6)+           ;POP THE STACK
2735  013664  022626                          POP2SP                  ;POP SCOPE RETURN AND VECTOR
2736  013666  005037  002200                  CLR     FTLCNT          ;DO NOT INC. FATAL COUNT IN MANUAL MODE.
2737  013672  032777  000200  165244  10$:    BIT     #RECDN,@$TKS    ;CONSOLE ACTIVE?
2738  013700  001407                          BEQ     11$
2739  013702  117701  165240                  MOVB    @$TKB,R1        ;STORE INPUT BUFFER
2740  013706  042701  000200                  BIC     #200,R1         ;CLEAR THE PARITY BIT
2741  013712  122701  000003                  CMPB    #3,R1           ;CHAR. EQUAL ESC. C?
2742  013716  001755                          BEQ     200$            ;YES-EXIT
2743  013720  117701  166250          11$:    MOVB    @TSTPTR,R1      ;GET THE NEXT TEST #
2744  013724  003005                          BGT     2$              ;NOT AT END OF LIST
2745  013726  042777  000100  165776          BIC     #RDENA,@VRCSR   ;CLEAR REC. INTERRUPTS BEFORE NEXT UNIT SELECT.
2746  013734  000137  002530                  JMP     MODCA           ;END OF LIST-GO SET UP NEXT 61
2747  013740  005301                  2$:     DEC     R1              ;ADJUST OFFSET
2748  013742  006301                          ASL     R1              ;USE TEST # TO FORM ADDRESS OFFSET
2749  013744  016137  022652  014002          MOV     TSTADD(R1),JMPADD+2     ;LOAD NEW ADDRESS
2750  013752  062737  000002  014002          ADD     #2,JMPADD+2     ;BYPASS INITIAL SCOPE LOOP
2751  013760  005237  002174                  INC     TSTPTR          ;INCREMENT TEST OPINTER
2752  013764  005037  177776                  CLR     PSW             ;SET NON-INT. PRIORITY TO ZERO
2753  013770  005046                          CLR     -(SP)           ;CLEAR THE PSW,LSI 11 STYLE.
2754  013772  012746  014000                  MOV     #JMPADD,-(SP)
2755  013776  000002                          RTI
2756  014000  000137  014000          JMPADD: JMP     JMPADD          ;EXIT TO NEXT SELECTED TEST
2757                                  ;;**************************************************************
  (2)                               ;;**************************************************************
2758                                          ;FOLLOWING ROUTINES ARE INTERRUPT HANDLERS FOR THE
2759                                          ;DL11 QUICK-TEST.
2760                                  ;;**************************************************************
  (2)                               ;;**************************************************************
2761  014004  117737  165724  002162  RECAD:  MOVB    @VRBUF,CHRD     ;GET THE RECEIVED CHAR.
2762  014012  042737  000200  002162          BIC     #200,CHRD       ;CLEAR ANY PARITY.
2763  014020  120437  002162                  CMPB    R4,CHRD         ;COMPARE RECEIVED TO XMITTED
2764  014024  001407                          BEQ     UPD4            ;AND UPDATE PATTERN IF OK.
2765  014026  042777  000104  165702  TOFF:   BIC     #TCOMB,@VXCSR   ;DATA ERROR OCCURED OR WE ARE DONE
2766  014034  042777  000100  165670          BIC     #RDENA,@VRCSR   ;EITHER WAY-EXIT.
2767  014042  000002                  REEX:   RTI
2768  014044  052777  000100  165664  UPD4:   BIS     #TENA,@VXCSR    ;ENABLE XMIT INT.
2769  014052  106304                          ASLB    R4              ;UPDATE DATA PATTERN.
2770  014054  032704  000200                  BIT     #BIT07,R4       ;ROTATED TO PARITY BIT?
2771  014060  001770                          BEQ     REEX            ;NO-CONTINUE TESTING
2772  014062  005004                          CLR     R4              ;YES-SET UP COMPLETE FLAG
2773  014064  000760                          BR      TOFF            ;AND EXIT.
2774  014066  110477  165646          TSMAD:  MOVB    R4,@VXBUF       ;XMIT DATA
2775  014072  042777  000100  165636          BIC     #TENA,@VXCSR    ;CLEAR XMIT INT. UNTIL LAST BIT REC.
2776  014100  000002                          RTI
2777
2778                                  ;;**************************************************************
2779                                          ;RECEIVE INTERRUPT ROUTINE.AFTER EACH RECEIVE
2780                                          ;CYCLE BUFFER POINTER (RBUFP) WILL BE SET TO (RBBUF).
2781                                          ;MAX. EXECUTION TIME IS APPROX 200US, AVERAGE =100US.
2782                                          ;UPON RECEIPT OF XON, XMTKIL BIT IS CHECKED IN VSTAT
2783                                          ;AND IF SET, WILL BE CLEARED AND XMIT INT. ENABLE SET.
2784                                          ;LOCATION ESAMB IS USED FOR ESC ASSEMBLY FLAGS. IE. BIT
2785                                          ;00 SET MEANS A033 WAS RECEIVED, BIT 01 SET MEANS AN ESCP
2786                                          ;SEQUENCE IS BEING ASSEMBLED. BIT 03
```

```
                                        ;SET INDICATES AND ESCAPE O SEQUENCE IS BEING ASSEMBLED.
                                        ;LOCATIONS STRO AND STRP ARE USED TO STORE ESCAPE
                                        ;O AND ESCAPE P SEQUENCES DETECTED,BUT NOT UTILIZED IN TEST.
                                    ;;**********************************************************************

014102                              INTRC:
(2) 014102  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
014104  017701  165624                      MOV     @VRBUF,R1       ;USE R1 FOR  STORAGE OF STATUS AND CH.
014110  042701  000200                      BIC     #200,R1         ;STRIP PARITY BIT.
014114  032737  000100  002224              BIT     #TXSUM,VSTAT    ;CHECKSUM CALCULATION REQUESTED?
014122  001403                              BEQ     11$             ;NO
014124  010105                              MOV     R1,R5           ;YES-STORE CHAR. AND
014126  004037  017646                      JSR     R0,CALCK        ;CALCULATE THE CHECKSUM.
014132  005237  031212              11$:    INC     ABUFP           ;INCREMINT THE RAW DATA POINTER
014136  023727  031212  031276              CMP     ABUFP,#ABBUF+50.        ;AT THE END OF BUFFER?
014144  001003                              BNE     12$             ;NO
014146  012737  031214  031212              MOV     #ABBUF,ABUFP    ;YES-RESET IT
014154  110177  015032              12$:    MOVB    R1,@ABUFP       ;STORE THE RAW DATA
014160  001505                              BEQ     6$              ;IF CHAR. IS NULL-GO STORE IT
014162  032737  000013  014772              BIT     #BIT00+BIT01+BIT03,ESAMB        ;ESC OR ESC O?
014170  001150                              BNE     ASESC           ;YES-KEEP ASSEMBLING
014172  120137  002132                      CMPB    R1,ESCN         ;BYTE = ESCN?
014176  101076                              BHI     6$              ;NO-PROBABLY A DISPLAY CH.-STORE IT.
014200  001007                              BNE     1$              ;NO-DECODE FOR XON,XOFF,SOM,EOM
014202  012737  000001  014772              MOV     #1,ESAMB        ;YES SET ESC ASSEMBLY FLAG.
014210  052737  000400  002224              BIS     #ESC,VSTAT      ;SET ESC RECEIVED FLAG
014216  000515                              BR      RSTER           ;AND EXIT
014220  120127  000023              1$:     CMPB    R1,#XOFF        ;SEE IF RECEIVED BYTE WAS XOFF
014224  001004                              BNE     2$              ;NO
014226  052737  100000  002224              BIS     #RXOFF,VSTAT    ;YES, SET XOFF IN STATUS REG.
014234  000506                              BR      RSTER           ;EXIT
014236  120127  000021              2$:     CMPB    R1,#XON         ;SEE IF BYTE WAS XON
014242  001016                              BNE     3$              ;NO
014244  042737  100000  002224              BIC     #RXOFF,VSTAT    ;YES, CLEAR XOFF IN VSTAT.
014252  032737  000200  002224              BIT     #XMKIL,VSTAT    ;CHECK XMIT KILL BIT.
014260  001474                              BEQ     RSTER           ;NOT SET, EXIT
014262  052777  000100  165446              BIS     #TENA,@VXCSR    ;SET XMIT INT. ENABLE.
014270  042737  000200  002224              BIC     #XMKIL,VSTAT    ;CLEAR THE XMIT KILLED FLAG
014276  000465                              BR      RSTER           ;EXIT
014300  120127  000002              3$:     CMPB    R1,#SOM         ;SEE IF BYTE WAS SOM
014304  001004                              BNE     4$              ;NO
014306  052737  040000  002224      31$:    BIS     #RSOM,VSTAT     ;YES, SET SOM IN VSTAT.
014314  000456                              BR      RSTER           ;EXIT

014316  120127  000004              4$:     CMPB    R1,#EOM         ;WAS BYTE EOM?
014322  001012                              BNE     5$              ;NO
014324  052737  020000  002224              BIS     #REOM,VSTAT     ;NOW SET EOM IN VSTAT.
014332  013737  014764  014770              MOV     RBBUF,RBUFP     ;RESET THE BUFFER POINTER.
014340  042737  000100  002224              BIC     #TXSUM,VSTAT    ;CLEAR CHECKSUM REQUEST BIT.
014346  000441                              BR      RSTER           ;AND EXIT
014350  123701  001752              5$:     CMPB    CARRT,R1        ;CHAR. =CARRIAGE RETURN?
014354  001403                              BEQ     51$             ;YES-GO SET END OF LINE FLAG
014356  123701  001754                      CMPB    LNFED,R1        ;CHAR.= LINEFEED?
014362  001004                              BNE     6$              ;NO- GO STORE IT
014364  052737  001000  002224      51$:    BIS     #EPL,VSTAT      ;SET END OF LINE INDICATOR
014372  000427                              BR      RSTER
```

```
2842
2843  014374  023737  014770  014766  6$:     CMP     RBUFP,REBUF     ;IS CIRCULAR BUFFER FILLED?
2844  014402  001003                          BNE     61$             ;NO
2845  014404  013737  014764  014770          MOV     RBBUF,RBUFP     ;YES, RESET POINTER TO BEGINNING
2846  014412  032737  000020  002224  61$:    BIT     #COMGP,VSTAT    ;RECEIVING GRAPHICS CHAR.?
2847  014420  001402                          BEQ     7$              ;NO
2848  014422  162701  000137                  SUB     #137,R1         ;YES-SUBTRACT 137 FROM RECEIVED CHAR.
2849
2850  014426  032737  000040  002224  7$:     BIT     #REVID,VSTAT    ;REVERSE VIDEO MODE?
2851  014434  001402                          BEQ     70$             ;NO STORE RECEIVED BYTE.
2852  014436  052701  000200                  BIS     #200,R1         ;YES-FORCE BIT7 AS REV. VIDEO IND.
2853  014442  110177  000322          70$:    MOVB    R1,@RBUFP       ;STORE BYTE AND
2854  014446  005237  014770                  INC     RBUFP           ;INCREMENT POINTER.
2855
2856  014452  005701                  RSTER:  TST     R1              ;CHECK FOR STATUS ERROR
2857  014454  100014                          BPL     RECXT           ;NO, EXIT ROUTINE
2858
2859  014456  052737  004000  002224          BIS     #RSTT,VSTAT     ;SET STATUS ERROR FLAG IN VSTAT
2860  014464  027727  000326  177777          CMP     @STTEP,#-1      ;IS ERROR TABLE FULL?
2861  014472  001405                          BEQ     RECXT           ;YES, EXIT ROUTINE
2862  014474  010177  000316                  MOV     R1,@STTEP       ;NO, STORE STATUS ERR. AND CHECK
2863  014500  062737  000002  015016          ADD     #2,STTEP        ;INCREMENT STATUS ERR. POINTER
2864
2865  014506                  RECXT:
 (2)  014506  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2866  014510  000002                          RTI                     ;EXIT
2867  014512  032737  000002  014772  ASESC:  BIT     #2,ESAMB        ;ASSEMBLING ESC P?
2868  014520  001063                          BNE     AESCP           ;YES-GO GET LAST CH,
2869  014522  032737  000010  014772          BIT     #BIT03,ESAMB    ;ASSEMBLING ESC 0?
2870  014530  001062                          BNE     AESC0           ;YES
2871  014532  122701  000120                  CMPB    #120,R1         ;CH.= A P?
2872  014536  001004                          BNE     10$             ;NO KEEP CHECKING
2873  014540  052737  000002  014772          BIS     #BIT01,ESAMB    ;YES-SET ESCP ASSEMBLY FLAG
2874  014546  000741                          BR      RSTER           ;AND EXIT
2875  014550  122701  000077          10$:    CMPB    #77,R1          ;CHAR.IS AN ESC ? ?
2876  014554  001403                          BEQ     110$            ;YES-FAKE AN ESC 0.
2877  014556  122701  000117                  CMPB    #117,R1         ;CHAR = 0?
2878  014562  001004                          BNE     11$             ;NO
2879  014564  052737  000010  014772  110$:   BIS     #BIT03,ESAMB    ;YES SET ESC 0 ASSEMBLY FLAG
2880  014572  000727                          BR      RSTER           ;AND EXIT
2881  014574  123701  002050          11$:    CMPB    RDCUR,R1        ;BYTE= CURSOR POSITION?
2882  014600  001004                          BNE     1$              ;NO-
2883  014602  052737  000004  002224          BIS     #CURPOS,VSTAT   ;YES-SET RECEIVED CURSOR POSITION.
2884  014610  000424                          BR      CESAM
2885  014612  122701  000057          1$:     CMPB    #SLSH,R1        ;BYTE=TERMINAL ID ESC?
2886  014616  001004                          BNE     2$              ;NO-CHECK FOR GRAPHICS SEQUENCE.
2887  014620  052737  000002  002224          BIS     #TRMID,VSTAT    ;YES-SET TERM. IDENT FLAG IN VSTAT
2888  014626  000415                          BR      CESAM
2889  014630  122701  000106          2$:     CMPB    #CKGP,R1        ;RECEIVED GRAPHICS CHAR. SEQUENCE?
2890  014634  001004                          BNE     3$              ;NO
2891  014636  052737  000020  002224          BIS     #COMGP,VSTAT    ;YES-SET GRAPHICS DATA FLAG.
2892  014644  000406                          BR      CESAM
2893  014646  122701  000107          3$:     CMPB    #NCKGP,R1       ;RECEIVED RESET GRAPHICS SEQ.?
2894  014652  001003                          BNE     CESAM           ;NO
2895  014654  042737  000020  002224          BIC     #COMGP,VSTAT    ;YES-SET NORMAL CHAR. RECEIVE.
2896  014662  005037  014772          CESAM:  CLR     ESAMB           ;CLEAR ASSEMBLY FLAG.
```

```
                014666  000671                          BR      RSTER           ;AND EXIT.
                014670  110137  015022          AESCP:  MOVB    R1,STRP         ;STORE ANY UNCHECKED FOR  ESC. P
                014674  000772                          BR      CESAM
                014676  123701  002016          AESCO:  CMPB    EEMP,R1         ;BYTE=ESC 0  -REV. VIDEO- ?
                014702  001004                          BNE     1$              ;NO
                014704  052737  000040  002224          BIS     #REVID,VSTAT    ;YES-SET REVERSE VIDEO MODE IN VSTAT.
                014712  000763                          BR      CESAM
                014714  123701  002020          1$:     CMPB    DEMP,R1         ;BYTE=ESC 0 DISABLE REV. VIDEO MODE?
                014720  001004                          BNE     2$              ;NO
                014722  042737  000040  002224          BIC     #REVID,VSTAT    ;YES-CLEAR REVERSE VIDEO MODE IN VSTAT.
                014730  000754                          BR      CESAM
                014732  122701  000171          2$:     CMPB    #CPABRT,R1      ;COPIER ABORT?
                014736  001403                          BEQ     3$              ;YES-SET ABORT FLAG IN VSTAT
                014740  122701  000172                  CMPB    #PRABRT,R1      ;PRINTER ABORT?
                014744  001004                          BNE     4$              ;NO
                014746  052737  010000  002224  3$:     BIS     #PABRT,VSTAT    ;YES-SET THE ABORT FLAG.
                014754  000742                          BR      CESAM           ;AND EXIT.
                014756  110137  015020          4$:     MOVB    R1,STRO         ;STORE ESCAPE 0 COMMAND
                014762  000737                          BR      CESAM

                014764  000000          RBBUF:  .WORD                   ;ADDRESS OF STAT OF BUFFER
                014766  000000          REBUF:  .WORD                   ;ADDRESS OF END OF BUFFER.
                014770  000000          RBUFP:  .WORD                   ;READ BUFFER POINTER.
                014772  000000          ESAMB:  .WORD   0               ;ESCAPE SEQ.ASSEMBLY AREA

                014774  000010          STTER:
                014774  000000                  0
                014776  000000                  0
                015000  000000                  0
                015002  000000                  0
                015004  000000                  0
                015006  000000                  0
                015010  000000                  0
                015012  000000                  0
                015014  177777                  .WORD   -1              ;STATUS REGISTER DELIMITER.
                015016  000000          STTEP:  .WORD                   ;STATUS ERROR POINTER.
                015020  000000          STRO:   .WORD   0               ;ESCAPE 0 STORAGE
                015022  000000          STRP:   .WORD                   ;ESCAPE P STORAGE

                        ;;**************************************************************
                                ;TRANSMIT INTERRUPT ROUTINE.
                                ;  IF XOFF BIT IS SET IN VSTAT , TRANSMISSION WILL NOT OCCUR
                                ;AND XMIT INT. ENABLE BIT WILL BE CLEARED AND THE ROUTINE
                                ;WILL BE EXITED IMMEDIATELY. IF AFTER THE TRANSMISSION
                                ;OF THE CHARACTER DURING THIS INTERRUPT CYCLE, THE
                                ;XMIT COUNT (XMCNT) IS EQUAL TO ZERO,
                                ;THE XMIT DONE BIT WILL BE SET IN VSTAT AND XMIT
                                ;INT ENABLE BIT WILL CLEARED. TRANSMIT COUNT(XMCNT) MUST BE
                                ;SET TO THE NUMBER OF BYTE/CHARACTER TO TRANSMIT.
                                ;IF LOCATION BLKM IS SET TO 1001,A SOM WILL PRECEED THE
                                ;DATA AND A EOM WILL FOLLOW IT. IF XMZER IS SET TO NON-
                                ;ZERO, ALL DATA(INCLUDING ZEROS) WILL BE XMITTED.
                        ;;**************************************************************
```

```
2947  015024  005737  002224          INTXM:  TST     VSTAT           ;HAS 61 TRANSMITTED XOFF?
2948  015030  100004                          BPL     NOKIL           ;NO XMIT ANOTHER
2949  015032  052737  000200  002224          BIS     #XMKIL,VSTAT    ;SET XMIT KILLED BIT IN VSTAT
2950  015040  000510                          BR      KIENA           ;GO KILL XMIT ENABLE
2951
2952  015042  105737  002227          NOKIL:  TSTB    BLKM+1          ;SOM/EOM TRANSMIT?
2953  015046  001406                          BEQ     NOSOM           ;NO
2954  015050  112777  000002  164662          MOVB    #SOM,@VXBUF     ;YES-ISSUE START OF MESSAGE.
2955  015056  105037  002227                  CLRB    BLKM+1          ;AND CLEAR SOM FLAG.
2956  015062  000002                          RTI
2957  015064  005737  015300          NOSOM:  TST     XMCNT           ;XMITTED THE BUFFER?
2958  015070  001006                          BNE     100$            ;NO-XMIT A NORMAL CHAR.
2959  015072  112777  000004  164640          MOVB    #EOM,@VXBUF     ;YES SEND EOM AND EXIT
2960  015100  105037  002226                  CLRB    BLKM
2961  015104  000452                          BR      2$
2962  015106  105777  000164          100$:   TSTB    @TBUFP          ;CHECK FOR CH.= ZERO. IF SO DO NOT XMIT
2963  015112  001016                          BNE     1$              ;OR COUNT BYTE. OR ARE WE
2964  015114  005737  020616                  TST     XMZER           ;XMITTING ZEROS?
2965  015120  001023                          BNE     22$             ;YES-XMIT NEXT BYTE
2966  015122  023737  015276  015274          CMP     TBUFP,TEBUF     ;AT END OF BUFFER?
2967  015130  001004                          BNE     10$             ;NO
2968  015132  013737  015272  015276          MOV     TBBUF,TBUFP     ;YES-RESET BUFFER POINTER
2969  015140  000740                          BR      NOKIL
2970  015142  005237  015276          10$:    INC     TBUFP
2971  015146  000735                          BR      NOKIL           ;LOOK FOR NON-ZERO BYTE TO TRANSMIT.
2972
2973  015150  032737  002000  002224  1$:     BIT     #CKSUM,VSTAT    ;CHECKSUM REQUESTED?
2974  015156  001404                          BEQ     22$
2975  015160  117705  000112                  MOVB    @TBUFP,R5       ;YES,LOAD THE BYTE
2976  015164  004037  017646                  JSR     R0,CALCK        ;AND CALCULATE THE NEW CHECKSUM.
2977  015170  117777  000102  164542  22$:    MOVB    @TBUFP,@VXBUF   ;TRAMSMIT A CHARACTER
2978  015176  023737  015276  015274          CMP     TBUFP,TEBUF     ;AT END OF CIRCULAR BUFFER?
2979  015204  001004                          BNE     11$             ;NO
2980  015206  013737  015272  015276          MOV     TBBUF,TBUFP     ;YES, RESET IT TO START.
2981  015214  000402                          BR      12$             ;BY-PASS INCREMENT BUFF. POINTER
2982  015216  005237  015276          11$:    INC     TBUFP           ;INCREMENT BUFFER POINTER.
2983
2984  015222  005337  015300          12$:    DEC     XMCNT           ;DECREMENT THE TRANSMIT COUNT
2985  015226  001401                          BEQ     2$              ;YES,CLEANUP,REQUEST ERRORS AND EXIT.
2986  015230  000002                          RTI                     ;NO, CONTINUE
2987  015232  105737  002226          2$:     TSTB    BLKM            ;SOM/EOM XMIT?
2988  015236  001014                          BNE     TXEX            ;YES-DO NOT SET XMDNE UNTIL EOM SENT.
2989  015240  052737  000001  002224          BIS     #XMDNE,VSTAT    ;SET THE DONE BIT IN VSTAT.
2990  015246  042737  002000  002224          BIC     #CKSUM,VSTAT    ;CLEAR THE CHECKSUM FLAG WHEN DONE.
2991  015254  013737  015272  015276          MOV     TBBUF,TBUFP     ;RESET BUFFER POINTER.
2992  015262  042777  000100  164446  KIENA:  BIC     #TENA,@VXCSR    ;CLEAR XMIT. INT. ENABLE
2993  015270  000002                  TXEX:   RTI
2994
2995
2996  015272  000000                  TBBUF:  .WORD                   ;CONTAINS INITIAL ADDRESS
2997  015274  000000                  TEBUF:  .WORD                   ;CONTAIN LAST ADDRESS
2998  015276  000000                  TBUFP:  .WORD                   ;CONTAINS CURRENT LOCATION
2999
3000  015300  000000                  XMCNT:  .WORD   0               ;LOADED WITH NUMBER OF XMITS.
3001                                  ;;*************************************************************
3002
```

```
                                        ;SUBROUTINE TO ISSUE RESET TO THE VT61, ENTERS MAINTENANCE MODE
                                        ;AND FORCES LINEAR ADDRESSING.
                                ;;*****************************************************************
015302  113737  001102  002230  RESETV: MOVB    $TSTNM,TSTNM    ;LOAD THE TEST NUMBER IN ERROR PRINT AREA.
015310  013746  002170                  MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
015314  013746  002130                  MOV     RESET,-(SP)     ;;PUSH RESET ON STACK
015320  013746  002054                  MOV     ESCO,-(SP)      ;;PUSH ESCO ON STACK
015324  004037  013456                  JSR     R0,TESC         ;GO XIMT IT
015330  004037  015412                  JSR     R0,GETON        ;GO LOOK FOR XON.
015334  000405                          BR      1$              ;FOUND IT.
015336  005237  002200                  INC     FTLCNT          ;ADD 1 TO FATAL XMIT COUNT.
015342  010037  001120                  MOV     R0,$GDADR       ;NO XON ISSUE XON ERROR
015346  104017                          ERROR   17
015350                          1$:
(2)     015350  013746  002170          MOV     ZERO,-(SP)      ;;PUSH ZERO ON STACK
3019    015354  013746  002000          MOV     EMAIN,-(SP)     ;;PUSH EMAIN ON STACK
3020    015360  013746  002054          MOV     ESCO,-(SP)      ;;PUSH ESCO ON STACK
3021    015364  013746  00201C          MOV     DRECT,-(SP)     ;;PUSH DRECT ON STACK
3022    015370  013746  002054          MOV     ESCO,-(SP)      ;;PUSH ESCO ON STACK
3023    015374  004037  013456  2$:     JSR     R0,TESC
3024    015400  005037  002224          CLR     VSTAT           ;CLEAR INT. FLAGS AFTER TERMINAL RESET
3025    015404  005037  017076          CLR     HDFLG           ;CLEAR PRINT HEADER FLAG.
3026    015410  000200                  RTS     R0
3027
3028                                    ;;*****************************************************************
3029                                            ;SUBROUTINE TO WAIT FOR AN XON. NO XON EXIT IS PC +2.
3030                                    ;;*****************************************************************
3031
3032    015412  012737  000454  002210  GETON:  MOV     #300.,BUBCT     ;SET UP TO LOOK FOR 3 SEC.
3033    015420  127727  013566  000021  1$:     CMPB    @ABUFP,#XON     ;RECEIVED A XON?
3034    015426  001412                  BEQ     GOTON           ;YES-EXIT.
3035    015430  012737  000001  017224          MOV     #1,DCOUNT       ;NO-DELAY 10 M.S.
3036    015436  004037  017162                  JSR     R0,DELAY
3037    015442  005337  002210                  DEC     BUBCT           ;AT END OF DELAY?
3038    015446  001364                          BNE     1$              ;NO
3039    015450  062700  000002                  ADD     #2,R0           ;YES-SET UP ERROR EXIT.
3040    015454  000200          GOTON:  RTS     R0
3041
3042                                    ;;*****************************************************************
3043
3044                                            ;SUBROUTINE TO ISSUE ESCZ AND LOOK FOR A RESPONSE-EITHER
3045                                            ;A -1 OR THE RETURNED IDENT. THE -1 INDICATES NO
3046                                            ;RESPONSE FROM THE UNIT UNDER TEST.
3047                                    ;;*****************************************************************
3048
3049    015456                  ZFLAG:
(2)     015456  012637  015514          MOV     (SP)+,ROSV1     ;;POP STACK INTO ROSV1
3050    015462  013746  002170          MOV     @#ZERO,-(SP)    ;;PUSH @#ZERO ON STACK
(2)     015466  013746  002126          MOV     @#ESCZ,-(SP)    ;;PUSH @#ESCZ ON STACK
3051    015472  004037  013456          JSR     R0,TESC         ;GO ISSUE ESZ SEQUENCE
3052    015476  012746  106003          MOV     #106003,-(SP)   ;;PUSH #106003 ON STACK
3053    015502  004037  013214          JSR     R0,RECTM        ;GO READ THE CHARACTER
3054    015506  013746  015514          MOV     ROSV1,-(SP)     ;;PUSH ROSV1 ON STACK
3055    015512  000200                  RTS     R0
```

```
3056  015514  000000                  ROSV1:  .WORD   0
3057
3058                                  ;;*****************************************************************
3059                                          ;ROUTINE TO CHECK SOFTWARE STATUS REGISTER (VSTAT)
3060                                          ;RECEIVE FLAGS ONLY.  ENTERED WITH ANTICIPATED
3061                                          ;STATUS WORD ON THE STACK.
3062                                  ;;*****************************************************************
3063
3064  015516                          CKSFT:
 (2)  015516  012637  002222                  MOV     (SP)+,ROSVE     ;;POP STACK INTO ROSVE
3065  015522  010137  002164                  MOV     R1,SVER1        ;SAVE R1
3066  015526  010237  002166                  MOV     R2,SVER2        ;SAVE R2
3067
3068  015532  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3069  015534  013702  002224                  MOV     VSTAT,R2        ;SET R2 EQUAL TO VSTAT
3070
3071  015540  042702  003576                  BIC     #003576,R2      ;CLEAR NON-ERROR BITS
3072  015544  020102                          CMP     R1,R2           ;COMPARE ANTICIPATED TO ACTUAL.
3073  015546  001432                          BEQ     NOER            ;NO UNUSAL BITS EXIT
3074
3075  015550  010137  001124                  MOV     R1,$GDDAT       ;MOVE GOOD STATUS TO MESSAGE
3076  015554  013737  002224  001126          MOV     VSTAT,$BDDAT    ;MOVE BAD STATUS TO MESSAGE
3077  015562  104003                          ERROR   3               ;ISSUE ERROR MESSAGE.
3078
3079
3080                                  ;;*****************************************************************
3081                                          ;ROUTINE TO PRINT THE STATUS REGISTER IN THE FOLLOWING
3082                                          ;FORMAT: STATUS BITS (XXX 000), CHARACTER TRANSFERRED (000  X X)
3083                                  ;;*****************************************************************
3084
3085  015564  012701  014774                  MOV     #STTER,R1       ;SET R1 EQUAL TO FIRST ENTRY
3086  015570  013702  015016                  MOV     STTEP,R2        ;SET R2 EQUAL LAST ENTRY
3087  015574  020102                  1$:     CMP     R1,R2           ;ARE THEY EQUAL
3088  015576  001416                          BEQ     NOER            ;YES-RESET POINTERS AND EXIT.
3089  015600  004037  015660                  JSR     R0,CLREG        ;CLEAR ERROR PRINT LOC.
3090  015604  013737  001732  001120          MOV     VRCSR,$GDADR    ;LOAD ADDRESS
3091  015612  017737  164114  001124          MOV     @VRCSR,$GDDAT   ;LOAD CSR
3092  015620  112137  001126          2$:     MOVB    (R1)+,$BDDAT    ;MOVE CHARACTER AND
3093  015624  112137  001123                  MOVB    (R1)+,$BDADR+1  ;STATUS BITS TO ERROR REGISTERS.
3094  015630  104002                          ERROR   2               ;ISSUE ERROR MESSAGE
3095  015632  000760                          BR      1$              ;DO AGAIN
3096  015634  013701  002164          NOER:   MOV     SVER1,R1        ;RESTORE R1 AND
3097  015640  013702  002166                  MOV     SVER2,R2        ;R2.
3098  015644  012737  014774  015016          MOV     #STTER,STTEP    ;RESET STATUS ERROR POINTER.
3099  015652  013746  002222                  MOV     ROSVE,-(SP)     ;;PUSH ROSVE ON STACK
3100  015656  000200                          RTS     R0              ;EXIT
3101
3102                                  ;;*****************************************************************
3103                                          ;SUBROUTINE TO CLEAR ERROR/DATA OUTPUT LOCATIONS. NEEDED
3104                                          ;ONLY WHEN DISPLAYING BYTES IN WORD LOCATIONS.
3105                                  ;;*****************************************************************
3106
3107  015660  005037  001120          CLREG:  CLR     $GDADR
3108  015664  005037  001122                  CLR     $BDADR
3109  015670  005037  001124                  CLR     $GDDAT
3110  015674  005037  001126                  CLR     $BDDAT
```

```
                                        RTS     R0
                                ;;**************************************************************
                                        ;SUBROUTINE TO TRANSMIT THE BUFFER AND WAIT FOR XMIT DONE
                                        ;AND END OF RECEIVE MESSAGE. SUBROUTINE WILL LOOP IF LOCATION
                                        ;RECITT IS PRE-LOADED WITH A NUMBER HIGHER THAN1(IE. MULTIPLE
                                        ;RECEIVES CAN BE ACCOMPLISHED WITH ONLY ONE ENTRY TO SUB-
                                        ;ROUTINE).WDSTOR AND BYSTOR ARE THE WORD(CURSOR POS.) AND BYTE
                                        ;STORAGE LOCATIONS,RESPECTIVELY.DEFAULT STORAGE IS THE REC. BUFFER.

                                ;;**************************************************************
015702                          XMREC:
(2) 015702  010546                      MOV     R5,-(SP)        ;;PUSH R5 ON STACK
015704  012737  001001  002226          MOV     #1001,BLKM      ;SET UP FOR A SOM/EOM TRANSMIT.
015712  042737  077577  002224          BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
015720  013701  016154                  MOV     BYSTOR,R1       ;LOAD THE STORAGE POINTERS
015724  013702  016152                  MOV     WDSTOR,R2
015730  052777  000100  164000          BIS     #TENA,@VXCSR    ;SET INTERRUPT ENABLES
015736  042737  061466  002224  XMITT:  BIC     #61466,VSTAT    ;CLEAR SOM,EOM,EPL,ESC,REV.VID., PARA. DELIM.,IDENT,CUR.
015744  005037  002220          1$:     CLR     DLAY            ;SET UP TIME OUT DELAY.
015750  032737  000001  002224          BIT     #XMDNE,VSTAT    ;IS XMIT DONE?
015756  001015                          BNE     3$              ;YES-LOOK FOR RECEIVE DONE.
015760  032737  020000  002224  2$:     BIT     #REOM,VSTAT     ;RECEIVED AN EOM?
015766  001401                          BEQ     20$             ;NO
015770  000435                          BR      CKSTR           ;YES-GO HANDLE DATA
015772  032737  100000  002224  20$:    BIT     #RXOFF,VSTAT    ;NO- IS XOFF SET?
016000  001761                          BEQ     1$              ;NO-STILL TRANSMITTONG.
016002  005337  002220                  DEC     DLAY            ;YES- RUN DELAY
016006  001364                          BNE     2$              ;WAITING FOR XON
016010  000416                          BR      XMAD2           ;NO XON-REPORT VT61 FAILURE.

016012  013705  031212          3$:     MOV     ABUFP,R5        ;LOAD CH. RECEIVED FLAG.
016016  005037  002220                  CLR     DLAY            ;SET UP RECEIVE DELAY.
016022  032737  020000  002224  4$:     BIT     #REOM,VSTAT     ;RECEIVE END OF MESSAGE?
016030  001015                          BNE     CKSTR           ;YES-CHECK DATA STORAGE POINTERS
016032  020537  031212                  CMP     R5,ABUFP        ;RECEIVED ANOTHER CHARACTER?
016036  001365                          BNE     3$              ;YES-RESET CH. FLAG AND DELAY
016040  005337  002220          5$:     DEC     DLAY            ;RUN DELAY
016044  001366                          BNE     4$              ;AND KEEP LOOKING FOR EOM.
016046  062700  000002          XMAD2:  ADD     #2,R0           ;TIME OUT OCCURRED-SET UP ERROR EXIT.
016052  005237  002200                  INC     FTLCNT          ;INCREMENT FATAL XMIT COUNT.
016056  004037  016266                  JSR     R0,RESPTR       ;AND REST ALL INTERRUPT POINTERS.
016062  000422                          BR      CKVST
016064  020102                  CKSTR:  CMP     R1,R2           ;STORAGE POINTERS CLEARED?
016066  001413                          BEQ     CHKITT          ;YES--LEAVE DATA IN REC. BUFFER.
016070  032737  000004  002224          BIT     #CURPOS,VSTAT   ;RECEIVED A CURSOR POSITION?
016076  001403                          BEQ     STRBYT          ;NO-GO STORE A BYTE.
016100  017722  176660                  MOV     @RBBUF,(R2)+    ;YES,STORE IT.
016104  000404                          BR      CHKITT          ;AND CHECK ITERATION COUNT.
016106  005701                  STRBYT: TST     R1              ;STORING A CHAR?
016110  001402                          BEQ     CHKITT          ;NO
016112  117721  176646                  MOVB    @RBBUF,(R1)+    ;STORE A RECEIVED BYTE
016116  005337  016150          CHKITT: DEC     RECITT          ;DONE RECEIVING?
016122  001305                          BNE     XMITT           ;NO-LOOP SUBROUTINE
016124  004037  020710                  JSR     R0,CKOFF        ;SEE IS XOFF IS UP.
016130                          CKVST:
```

```
  (2)  016130  012746  060001                    MOV     #60001,-(SP)    ;;PUSH #60001 ON STACK
 3166  016134  004037  015516                    JSR     R0,CKSFT
 3167  016140  004037  016266                    JSR     R0,RESPTR       ;RESET INTERRUPT POINTERS.
 3168  016144  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
 3169  016146  000200                    XMXT:   RTS     R0              ;EXIT SUBROUTINE.
 3170  016150  000000                    RECITT: .WORD   0               ;RECEIVE ITERATION COUNT.
 3171  016152  000000                    WDSTOR: .WORD   0               ;WORD STORAGE POINTER
 3172  016154  000000                    BYSTOR: .WORD   0               ;BYTE STORAGE POINTER
 3173
 3174                                    ;;**************************************************************
 3175                                            ;SUBROUTINE TO  XMIT THE BYTE AT TBUFP.
 3176                                    ;;**************************************************************
 3177
 3178  016156  042737  000001  002224    XMIT1:  BIC     #1,VSTAT        ;CLEAR XMIT DONE FLAG
 3179  016164  012737  000001  015300            MOV     #1,XMCNT        ;SET UP TO XMIT 1 BYTE
 3180  016172  052777  000100  163536            BIS     #TENA,@VXCSR
 3181  016200                            1$:
  (2)  016200  012746  000001                    MOV     #XMDNE,-(SP)    ;;PUSH #XMDNE ON STACK
 3182  016204  012746  000001                    MOV     #1,-(SP)        ;;PUSH #1 ON STACK
 3183  016210  004037  020620                    JSR     R0,WTBGND       ;LOOK FOR XMIT DONE
 3184  016214  000401                            BR      FTLEXT          ;HUNG TRANSMIT-CLEAR FLAGS AND EXIT
 3185  016216  000402                            BR      NORXT           ;NORMAL EXIT.
 3186  016220  005037  002224    FTLEXT: CLR     VSTAT           ;CLEAR ANY FLAGS
 3187  016224  000200                    NORXT:  RTS     R0              ;AND EXIT
 3188
 3189                                    ;;**************************************************************
 3190                                            ;SUBROUTINE TO ISSUE A BYTE AT A TIME UNTIL A ZERO
 3191                                            ;BYTE IS ENCOUNTERED.
 3192                                    ;;**************************************************************
 3193
 3194  016226  112777  000002  177042    LDXMIT: MOVB    #SOM,@TBUFP     ;SEND THE START OF MESSAGE.
 3195  016234  000403                            BR      2$
 3196  016236  112377  177034            1$:     MOVB    (R3)+,@TBUFP    ;MOVE A BYTE TO XMIT BUFFER
 3197  016242  001403                            BEQ     LDOUT           ;IF A ZERO BYTE-EXIT
 3198  016244  004037  016156            2$:     JSR     R0,XMIT1        ;GO XMIT A BYTE
 3199  016250  000772                            BR      1$              ;XMIT AGAIN.
 3200  016252  112777  000004  177016    LDOUT:  MOVB    #EOM,@TBUFP     ;SEND THE END OF MESSAGE.
 3201  016260  004037  016156                    JSR     R0,XMIT1
 3202  016264  000200                            RTS     R0
 3203
 3204                                    ;;**************************************************************
 3205                                            ;ROUTINE TO RESET ALL INTERRUPT POINTERS.
 3206                                    ;;**************************************************************
 3207
 3208  016266  042777  000100  163442    RESPTR: BIC     #TENA,@VXCSR    ;CLEAR INTERRUPT ENABLES
 3209  016274  013737  014764  014770            MOV     RBBUF,RBUFP     ;RESET RECEIVE BUF POINTER
 3210  016302  013737  015272  015276            MOV     TBBUF,TBUFP     ;RESET XMMT BUF POINTER
 3211  016310  012737  014774  015016            MOV     #STTER,STTEP    ;RESET RECEIVE STATUS ERR POINTER
 3212  016316  005037  015300                    CLR     XMCNT           ;CLEAR TRANSMIT COUNT
 3213  016322  005037  014772                    CLR     ESAMB           ;CLEAR ESC ASSEMBLY FLAGS
 3214  016326  012737  000001  016150            MOV     #1,RECITT       ;RESET REC. ITERATION COUNT
 3215  016334  005037  016152                    CLR     WDSTOR          ;CLEAR STORAGE POINTERS
 3216  016340  005037  016154                    CLR     BYSTOR
 3217  016344  000200                            RTS     R0
 3218
 3219
```

```
                                        ;;*****************************************************************
                                        ;SUBROUTINE TO ISSUE CURSOR POSITION ERROR. GOOD
                                        ;LINE/COLUMN MUST BE A WORD ON STACK. ERROR
                                        ;POSITION IS EXPECTED TO BE @ RBBUF.
                                        ;;*****************************************************************

016346                                  CURER:
(2) 016346  012637  002222                  MOV     (SP)+,ROSVE     ;;POP STACK INTO ROSVE
3227 016352  012637  002162                 MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
3228 016356  162737  020040  002162         SUB     #20040,CHRD     ;EXTRACT MOD 40 FROM GOOD POSITION
3229 016364  004037  015660                 JSR     R0,CLREG
3230 016370  113737  002163  001124         MOVB    CHRD+1,$GDDAT   ;LOAD MESSAGE WITH GOOD
3231 016376  113737  002162  001120         MOVB    CHRD,$GDADR     ;LINE AND COLUMN
3232 016404  017737  176354  002162         MOV     @RBBUF,CHRD     ;LINE AND COLUMN.
3233 016412  162737  020040  002162         SUB     #20040,CHRD     ;EXTRACT MOD 40 FROM BAD POSITION.
3234 016420  113737  002163  001126         MOVB    CHRD+1,$BDDAT   ;LOAD MESSAGE WITH BAD
3235 016426  113737  002162  001122         MOVB    CHRD,$BDADR     ;LINE AND COLUMN.
3236 016434  104006                        ERROR   6               ;ISSUE ERROR
3237 016436  013746  002222                 MOV     ROSVE,-(SP)     ;;PUSH ROSVE ON STACK
3238 016442  000200                        RTS     R0
3239
3240                                    ;;*****************************************************************
3241
3242                                    ;;*****************************************************************
3243                                        ;SUBROUTINE TO DECREMENT CUSROR POSITION IN A
3244                                        ;LINEAR SEQUENCE. (IE. ROW 20, COL 1 ;ROW 20 COL0 ;ROW 17, COL 157).
3245                                    ;;*****************************************************************
3246
3247 016444  123727  016551  000040  CMPOS:  CMPB    LNRW+1,#40      ;AT LEFT EDGE OF ROW?
3248 016452  001403                        BEQ     1$              ;YES, GO ADJUST COL., ROW.
3249 016454  105337  016551                 DECB    LNRW+1          ;NO, DECREMENT COL. AND EXIT
3250 016460  000200                        RTS     R0
3251 016462  123727  016550  000040  1$:    CMPB    LNRW,#40        ;AT ROW 0?
3252 016470  001405                        BEQ     2$              ;YES, NO DECREMENT POSSIBLE-EXIT.
3253 016472  105337  016550                 DECB    LNRW            ;NO, DECREMENT ROW AND
3254 016476  112737  000157  016551         MOVB    #157,LNRW+1     ;SET COL. TO RIGHT EDGE.
3255 016504  000200                    2$:  RTS     R0
3256
3257                                    ;;*****************************************************************
3258                                        ;SUBROUTINE TO INCREMENT CURSOR POSITION IN A LINEAR
3259                                        ;SEQUENCE (IE. ROW 10, COL 78, ROW 10. COL 79, ROW 11 , COL 0).
3260                                    ;;*****************************************************************
3261
3262 016506  123727  016551  000157  CPPOS:  CMPB    LNRW+1,#157     ;AT RIGHT EDGE OF ROW
3263 016514  001403                        BEQ     1$              ;YES, ADJUST ROW AND COLUMN.
3264 016516  105237  016551                 INCB    LNRW+1          ;NO, INCREMENT COL. COUNT
3265 016522  000200                        RTS     R0              ;AND EXIT
3266 016524  123727  016550  000067  1$:    CMPB    LNRW,#67        ;AT BOTTOM ROW?
3267 016532  001405                        BEQ     2$              ;YES, NO INCREMENT POSSIBLE-EXIT.
3268 016534  105237  016550                 INCB    LNRW            ;NO, INCREMENT ROW COUNT AND
3269 016540  112737  000040  016551         MOVB    #40,LNRW+1      ;SET COL. TO LEFT EDGE.
3270 016546  000200                    2$:  RTS     R0
3271
3272 016550  000000                    LNRW:   .WORD   0               ;CONTAINS UPDATED CURSOR POSITION.
3273                                    ;;*****************************************************************
3274
```

```
                                        ;SUBROUTINE TO XMIT, RECEIVE AND COMPARE. DATA ERRORS
                                        ;ARE REPORTED FROM SUBROUTINE. IF THE TRANSMIT OR
                                        ;RECEIVE LOOPS 'TIME OUT', EXIT FROM SUBROUTINE WILL
                                        ;BE NORMAL EXIT +2. SUBROUTINE ENTERED WITH (R1)=
                                        ;GOOD DATA BUFFER, (R2)=RECEIVE DATA BUFFER AND
                                        ;R3=COMPARE COUNT. IF THE VT61 DOES NOT HANG,THE ROUTINE
                                        ;WILL WAIT FOR END OF REC. MESSAGE(EOM).


                                        ;;*****************************************************************

016552                                  XRCMP:
(2) 016552  010446                              MOV     R4,-(SP)        ;;PUSH R4 ON STACK
016554  005004                                  CLR     R4              ;USE R4 A RECEIVE COUNTER.
016556  012737  001001  002226                  MOV     #1001,BLKM      ;SET UP FOR A SOM/EOM TRANSMIT.
016564  042737  077577  002224                  BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
016572  052777  000100  163136                  BIS     #TENA,@VXCSR    ;SET INTERRUPT ENABLES.
016600  005037  017076                          CLR     HDFLG           ;CLEAR ERROR 13 PRINT FLAG
016604  012705  031162                          MOV     #TCRLB+450,R5   ;R5 IS ERROR STORAGE POINTER
016610  005037  002220              1$:         CLR     DLAY            ;SET UP TIME OUT DELAY
016614  032737  000001  002224                  BIT     #XMDNE,VSTAT    ;XMIT DONE?
016622  001014                                  BNE     XREC            ;YES-GO RECEIVE
016624  023737  014764  014770      2$:         CMP     RBBUF,RBUFP     ;HAS RECEIVE OPERATION BEGUN?
016632  103410                                  BLO     XREC            ;YES-GO RECEIVE
016634  032737  100000  002224                  BIT     #RXOFF,VSTAT    ;XMIT XOFF SET?
016642  001762                                  BEQ     1$              ;NO-KEEP LOOKING FOR XMIT DONE?
016644  005337  002220                          DEC     DLAY            ;YES RUN DELAY AND LOOK
016650  001365                                  BNE     2$              ;FOR XON OR RECEIVED CH.
016652  000432                                  BR      XRERR           ;TRANSMIT TIMEOUT-SET UP ERROR EXIT

016654  005037  002220              XREC:       CLR     DLAY            ;SET UP TIME OUT DELAY
016660  020237  014770              1$:         CMP     R2,RBUFP        ;INSURE COMPARE POINTER
016664  103410                                  BLO     2$              ;LESS THAN RECEIVE POINTER
016666  032737  020000  002224                  BIT     #REOM,VSTAT     ;RECEIVE EOM?
016674  001030                                  BNE     XREXT           ;YES-SET UP TO EXIT
016676  005337  002220                          DEC     DLAY            ;RUN TIMEOUT DELAY
016702  001416                                  BEQ     XRERR           ;TIME OUT OCCURRED-ERROR EXIT
016704  000765                                  BR      1$              ;RETURN TO CHECK RECEIVE COUNT
016706  005204                      2$:         INC     R4              ;ADD 1 TO RECEIVE COUNTER.
016710  122122                                  CMPB    (R1)+,(R2)+     ;COMPARE CHARACTERS
016712  001407                                  BEQ     4$              ;EQUAL-COMPARE AGAIN
016714  020527  031212                          CMP     R5,#TCRLB+500   ;ALLREADY STORED 50 ERRORS?
016720  103004                                  BHIS    4$              ;YES-BYPASS STORAGE
016722  114125                                  MOVB    -(R1),(R5)+     ;STORE GOOD DATA
016724  114225                                  MOVB    -(R2),(R5)+     ;STORE BAD DATA
016726  010425                                  MOV     R4,(R5)+        ;LOAD RECEIVE COUNT
016730  132122                                  BITB    (R1)+,(R2)+     ;RESET POINTERS AND
016732  005303                      4$:         DEC     R3              ;CHECK COMPARE COUNT
016734  001410                                  BEQ     XREXT           ;ALL DONE-EXIT
016736  000746                                  BR      XREC            ;COMPARE ANOTHER
016740  062700  000002              XRERR:      ADD     #2,R0           ;SET UP ERROR EXIT
016744  005237  002200                          INC     FTLCNT          ;INCREMENT FATAL XMIT COUNT.
016750  004037  016266                          JSR     R0,RESPTR       ;RESET INTERRUPT POINTERS.
016754  000440                                  BR      XROUT
016756                              XREXT:
(2) 016756  012746  020000                      MOV     #REOM,-(SP)     ;;PUSH #REOM ON STACK
```

```
3329  016762  012746  000004            MOV     #4,-(SP)        ;;PUSH #4 ON STACK
3330  016766  004037  020620            JSR     R0,WTBGND
3331  016772  000431                    BR      XROUT           ;NO EOM-ISSUE ERROR AND EXIT.
3332  016774  162705  031162            SUB     #TCRLB+450,R5   ;NOW EXTRACT ERROR COUNT-IF ANY.
3333  017000  010501                    MOV     R5,R1           ;AND STORE IT IN R1
3334  017002  012705  031162            MOV     #TCRLB+450,R5   ;RELOAD ERROR POINTER
3335  017006  005701                    TST     R1              ;TEST FOR ERRORS
3336  017010  001422                    BEQ     XROUT           ;NO-CHECK STATUS AND EXIT
3337  017012  005737  017076            TST     HDFLG           ;;DATA ERROR HEADER PRINTED?
3338  017016  001003                    BNE     1$              ;YES-BYPASS HEADER PRINT
3339  017020  104012                    ERROR   12              ;PRINT DATA ERROR HEADER
3340  017022  005237  017076            INC     HDFLG           ;SET HEADER PRINT FLAG
3341  017026  004037  015660    1$:     JSR     R0,CLREG        ;ERROR WAS LEGTIMATE. LOAD
3342  017032  112537  001124            MOVB    (R5)+,$GDDAT    ;ERROR MESSAGE AND ISSUE
3343  017036  112537  001126            MOVB    (R5)+,$BDDAT    ;IT.
3344  017042  012537  001120            MOV     (R5)+,$GDADR    ;LOAD RECEIVE COUNT
3345  017046  104004                    ERROR   4               ;ISSUE DATA COMPARE ERROR
3346  017050  162701  000004            SUB     #4,R1           ;DECREMENT ERROR COUNT
3347  017054  001364                    BNE     1$              ;PRINT ANOTHER IF NOT AT ZERO
3348  017056  004037  020710    XROUT:  JSR     R0,CKOFF        ;SEE IS XOFF IS UP.
3349  017062  012746  060001            MOV     #60001,-(SP)    ;;PUSH #60001 ON STACK
3350  017066  004037  015516            JSR     R0,CKSFT        ;CHECK FOR VSTAT /STATUS ERR.
3351  017072  012604                    MOV     (SP)+,R4        ;;POP STACK INTO R4
3352  017074  000200                    RTS     R0              ;EXIT SUBROUTINE
3353
3354  017076  000000    HDFLG:  0                               ;INHIBIT PRINT FLAG.
3355
3356                    ;;**************************************************************
3357
3358                            ;SUBROUTINE TO CREATE A 'RULER' IN LOCATIONS 200
3359                            ;TO 317.
3360
3361                    ;;**************************************************************
3362
3363  017100  012701  030712    CRRUL:  MOV     #TCRLB+200,R1   ;LOAD STARTING ADDRESS
3364  017104  012702  130461            MOV     #130461,R2      ;LOAD INITIAL RULER ASCII CODES.
3365  017110  110221            1$:     MOVB    R2,(R1)+        ;STORE A RULER BYTE IN XMIT BUF.
3366  017112  022701  031032            CMP     #TCRLB+320,R1   ;RULER COMPLETE?
3367  017116  103001                    BHIS    2$              ;NO
3368  017120  000200                    RTS     R0              ;AND EXIT.
3369  017122  105202            2$:     INCB    R2              ;INCREMENT ASCII BYTE
3370  017124  122702  000272            CMPB    #272,R2         ;END OF REVERSE VIDEO?
3371  017130  001003                    BNE     3$              ;NO-SEE IF END OF NORMAL.
3372  017132  012702  030660            MOV     #030660,R2      ;SET UP TO ISSUE REVERSE 0.
3373  017136  000405                    BR      5$
3374  017140  122702  000072    3$:     CMPB    #72,R2          ;END OF NORMAL VIDEO?
3375  017144  001361                    BNE     1$              ;NOT AT END OF A VIDEO STRING.
3376  017146  012702  130460            MOV     #130460,R2              ;YES-SET UP TO ISSUE NORMAL 0.
3377  017152  110221            5$:     MOVB    R2,(R1)+        ;DO IT
3378  017154  105202                    INCB    R2              ;SET BYTE TO NEXT ASCII CODE
3379  017156  000302                    SWAB    R2              ;REVERSE VIDEO MODE.
3380  017160  000753                    BR      1$              ;BEGIN NEXT STRING
3381
3382                    ;;**************************************************************
3383                            ;SUBROUTINE TO DELAY 10 M.S. TIME THE NUMBER INLOCATION
3384                            ;DCOUNT. THE PROCESSOR TYPE PRE-DETERMINES THE # OF LOOPS
```

```
3385                                    ;REQUIRED TO DELAY 10 M.S. FOR ONE ITERATION. LOCATION
3386                                    ;PMULT IS PRE-LOADED WITH : 11/45 = 4, 11/40 = 2
3387                                    ;AND 11/10 =1.
3388                                ;;**********************************************************************
3389
3390  017162                        DELAY:
 (2)  017162  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 (2)  017164  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3391  017166  013702  017222        1$:     MOV     PMULT,R2        ;LOAD PROCESSOR MULTIPLIER
3392  017172  012701  002570        2$:     MOV     #1400.,R1       ;LOAD 10 M.S. DELAY
3393  017176  005301                        DEC     R1              ;RUN BASIC DELAY
3394  017200  001376                        BNE     .-2
3395  017202  005302                        DEC     R2              ;RUN MULTIPLIER DELAY
3396  017204  001372                        BNE     2$
3397  017206  005337  017224              DEC     DCOUNT          ;RUN ITERATION COUNT
3398  017212  001365                        BNE     1$
3399  017214  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 (2)  017216  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
3400  017220  000200                        RTS     R0
3401
3402  017222  000000                PMULT:  0                       ;PROCESSOR MULTIPLIER
3403  017224  000000                DCOUNT: 0                       ;ITERATION COUNT
3404
3405                                ;;**********************************************************************
3406
3407                                        ;SUBROUTINE TO GENERATE A INCREMENTING PATTERN AT
3408                                        ;(R1)+.  ENTER WITH R3 EQUAL TO # OF CH. TO CREATE.
3409                                        ;R5 IS UTILIZED AS A WORK REGISTER.
3410
3411                                ;;**********************************************************************
3412
3413  017226  012705  000041        BLDINC: MOV     #41,R5          ;LOAD R5 WITH INITIAL CH.
3414  017232  110521                BLDINA: MOVB    R5,(R1)+        ;MOVE A CH. TO BUFFER
3415  017234  005303                        DEC     R3              ;DECREMENT BYTE COUNT
3416  017236  001001                        BNE     2$              ;NOT DONE-UPDATE PATTERN
3417  017240  000200                        RTS     R0              ;EXIT-DONE.
3418  017242  105205                2$:     INCB    R5              ;UPDATE CH. PATTERN
3419  017244  122705  000177              CMPB    #177,R5         ;PATTERN EXCEEDED MAX?
3420  017250  001766                        BEQ     BLDINC          ;YES-RESET IT.
3421  017252  000767                        BR      BLDINA          ;NO-ISSUE CURRENT PATTERN.
3422
3423
3424                                ;;**********************************************************************
3425
3426                                        ;SUBROUTINE TO FILL THE SCREEN WITH INCREMENTING DATA
3427                                ;;**********************************************************************
3428
3429  017254  042737  077577  002224 DATSC:  BIC     #77577,VSTAT            ;CLEAR INTERRUPT FLAGS.
3430  017262  013701  015272                MOV     TBBUF,R1
3431  017266  012703  000500                MOV     #320.,R3        ;FILL XMIT BUFFER WITH INCRE-
3432  017272  004037  017226                JSR     R0,BLDINC       ;MENTING PATTERN
3433  017276  012737  003600  015300 10$:    MOV     #TOTCH,XMCNT    ;SET UP TO XMIT 1920 BYTES
3434  017304  052777  000100  162424        BIS     #TENA,@VXCSR
3435
3436  017312  032737  000001  002224 1$:     BIT     #XMDNE,VSTAT    ;XMIT DONE?
3437  017320  001774                        BEQ     .-6             ;NO
```

```
3438
3439                                    ;;**********************************************************************
3440
3441                                            ;SUBROUTINE TO RESET VT61 AND DISPLAY MESSAGE
3442                                            ;POINTED TO BY R2.
3443
3444                                    ;;**********************************************************************
3445
3446  017322  004037  015302       DSMES:  JSR     R0,RESETV       ;RESET THE UNIT AND WAIT FOR XON.
3447  017326  042737  077577  002224       BIC     #77577,VSTAT    ;CLEAR ALL FLAGS EXCEPT XOFF AND XMKIL.
3448  017334  012737  000005  015300       MOV     #5,XMCNT        ;PRE-LOAD XMIT COUNT.
3449  017342  013701  015272               MOV     TBBUF,R1        ;LOAD XMIT BUFFER WITH:
3450  017346  012721  000002               MOV     #SOM,(R1)+      ;START OF MESSAGE
3451  017352  013721  002054               MOV     ESCO,(R1)+
3452  017356  013721  002010               MOV     DRECT,(R1)+     ;DISABLE RECTANGULAR MODE
3453  017362  005237  015300       1$:     INC     XMCNT           ;INCREMENT TRANSMIT COUNT
3454  017366  112221                       MOVB    (R2)+,(R1)+     ;DISPLAY MESSAGE
3455  017370  001374                       BNE     1$
3456  017372  112711  000004               MOVB    #EOM,(R1)       ;TERMINATE WITH END OF MESSAGE.
3457  017376  052777  000100  162332       BIS     #TENA,@VXCSR    ;XMIT IT AND WAIT FOR
3458  017404  032737  000001  002224  2$:     BIT     #XMDNE,VSTAT    ;DONE
3459  017412  001774                       BEQ     2$
3460  017414  000200                       RTS     R0
3461
3462                                    ;;**********************************************************************
3463
3464                                            ;SUBROUTINE TO CONVERT A BINARY CHARACTER
3465                                            ;TO 3 OCTAL CHARACTERS. R1 CONTAINS BINARY
3466                                            ;NUMBER. RESULT IS STORED IN LOCATIONS SVER1,
3467                                            ;SVER2
3468
3469                                    ;;**********************************************************************
3470  017416                        BINOCT:
 (2)  017416  010546                       MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3471  017420  012705  000002               MOV     #2,R5           ;LOAD ITERATION COUNT
3472  017424  000403                       BR      2$              ;BYPASSS SHIFTS FOR 1ST CONVERSION
3473  017426  106201               1$:     ASRB    R1
3474  017430  106201                       ASRB    R1              ;SHIFT A CHAR INTO POSITION
3475  017432  106201                       ASRB    R1
3476  017434  110165  002164       2$:     MOVB    R1,SVER1(R5)    ;STORE THE BINARY OFFSET
3477  017440  142765  000370  002164       BICB    #370,SVER1(R5)  ;CLEAR NON ESSENTIAL BITS
3478  017446  152765  000060  002164       BISB    #60,SVER1(R5)   ;CONVERT OFFSET TO OCTAL
3479  017454  005305                       DEC     R5              ;DECREMENT CONVERSION COUNT
3480  017456  100363                       BPL     1$              ;NOT DONE CONVERT ANOTHER
3481  017460  112737  000040  002167       MOVB    #40,SVER2+1     ;LOAD A SPACE
3482  017466  012605                       MOV     (SP)+,R5        ;;POP STACK INTO R5
3483  017470  000200                       RTS     R0
3484
3485                                    ;;**********************************************************************
3486                                            ;SUBROUTINE TO CONVERT AN OCTAL CHAR. TO BINARY.  REG
3487                                            ;R1 CONTAINS OCTAL AND REG R2 IS BINARY ASSEMBLY AREA.
3488                                    ;;**********************************************************************
3489
3490  017472  042701  177770       OCTBIN: BIC     #177770,R1      ;EXTRACT OCTAL COMPONENT
3491  017476  005702                       TST     R2              ;FIRST CONVERSION?
3492  017500  001403                       BEQ     NOSHFT          ;YES - DO NOT SHIFT
```

```
3493  017502  006302                    ASL     R2              ;NO - SHIFT PREVIOUS CHAR.
3494  017504  006302                    ASL     R2
3495  017506  006302                    ASL     R2
3496  017510  060102            NOSHFT: ADD     R1,R2           ;ADD CURRENT CHAR.
3497  017512  000200                    RTS     R0
3498
3499                            ;;*****************************************************************
3500                                    ;ROUTINE TO WAIT FOR C/R FROM VT61 UNDER TEST
3501                            ;;*****************************************************************
3502
3503  017514  032777  000200  162210    GTCR:   BIT     #RECDN,@VRCSR   ;WAIT FOR REVEIVE DONE
3504  017522  001774                    BEQ     .-6
3505  017524  127737  162204  001752            CMPB    @VRBUF,CARRT    ;CHAR = CARRIAGE RETURN?
3506  017532  001370                    BNE     GTCR            ;NO-KEEP LOOKING
3507  017534  000200                    RTS     R0              ;YES-EXIT
3508
3509
3510                            ;;*****************************************************************
3511                                    ;SUBROUTINE TO GET A CHARACTER (NUMERIC) FROM THE
3512                                    ;CONSOLE.  IF OTHER THAN A NUMERIC IS TYPED A
3513                                    ;"?" WILL BE ECHOED.
3514                            ;;*****************************************************************
3515
3516  017536  004037  013576    GTNUM:  JSR     R0,CONRD        ;GET A CHAR
3517  017542  012601                    MOV     (SP)+,R1        ;POP STACK INTO R1
3518  017544  122701  000054            CMPB    #54,R1          ;CHAR. =COMMA?
3519  017550  001411                    BEQ     1$              ;YES-GO PRINT IT
3520  017552  123701  001752            CMPB    CARRT,R1        ;CHAR. = CARRIAGE RETURN?
3521  017556  001406                    BEQ     1$
3522  017560  120127  000060            CMPB    R1,#60
3523  017564  103421                    BLO     QUST            ;IF CHAR. IS LESS THAN 60
3524  017566  120127  000067            CMPB    R1,#67          ;OR MORE THAN 67, TYPE
3525  017572  101016                    BHI     QUST            ;A QUESTION MARK
3526  017574  110137  017644    1$:     MOVB    R1,TYPNUM
3527  017600  104401  017644            TYPE    ,TYPNUM
3528  017604  123701  001754            CMPB    LNFED,R1
3529  017610  001406                    BEQ     GTEXT
3530  017612  123701  001752            CMPB    CARRT,R1        ;IF CHAR. - C/R SET UP TO ISSUE
3531  017616  001003                    BNE     GTEXT           ;LINE FEED BEFORE EXITING.
3532  017620  113701  001754            MOVB    LNFED,R1
3533  017624  000763                    BR      1$
3534  017626  000200            GTEXT:  RTS     R0              ;GOOD CHAR., EXIT
3535  017630  112737  000077  017644  QUST:   MOVB    #77,TYPNUM
3536  017636  104401  017644            TYPE    ,TYPNUM         ;TYPE QUESTION MARK AND
3537  017642  000735                    BR      GTNUM           ;KEEP LOOKING.
3538  017644     000            TYPNUM: .BYTE   0
3539  017645     000                    .BYTE   0
3540
3541
3542                            ;;*****************************************************************
3543                                    ;SUBROUTINE TO CALCULATE CHECKSUM ON THE LOWER
3544                                    ;BYTE OF R5.  R4 IS STORAGE FOR THE CHECKSUM
3545                                    ;CHARACTER. ALGORITHM FOR CHECKSUM IS ROTATE
3546                                    ;CURRENT ONE PLACE LEFT AND XOR NEW CHAR. CHECKSUM
3547                                    ;IS THE LOWER 7 BITS OF R4
3548
```

```
                                ;;*****************************************************************
                                CALCK:  BIC     #177400,R5      ;CLEAR UPPER BYTE OF R5
                                        CMPB    R5,#XON         ;CHAR. =XON?
                                        BEQ     NOCALC          ;YES DO NOT CALCULATE CHECKSUM
                                        CMPB    R5,#XOFF        ;CHAR =XOFF?
                                        BEQ     NOCALC          ;YES DO NOT CALCULATE CHECKSUM

                                        CLC                     ;INSURE CARRY BIT INITIALLLY CLEAR
                                        TSTB    R4              ;SET UP TO ROTATE R4
                                        BPL     1$              ;A FULL 8 BYTES

                                        SEC                     ;R4 WAS NEG. SO ROTATE A ONE
                                1$:     ROLB    R4              ;INTO LOW ORDER BIT.
                                        MOV     R4,R3
                                        BIC     R5,R3           ;NOT A AND B
                                        BIC     R4,R5           ;NOT B AND A
                                        BIS     R3,R5           ;ORED
                                        MOV     R5,R4           ;EQUAL NEW CHECKSUM
                                NOCALC: RTS     R0
                                ;;*****************************************************************

                                        ;SUBROUTINE TO LOAD XMIT BUFFER FROM R0 THRU R1
                                ;;*****************************************************************

                                LDBUF:  MOVB    (R0)+,(R1)+     ;LOAD A BYTE
                                        BNE     .-2             ; UNTIL ZERO BYTE FOUND.
                                        RTS     R0
                                ;;*****************************************************************
                                        ;SUBROUTINE TO CHECK THE VT61 FOR A PERIPHERAL ABORT.
                                ;;*****************************************************************

                                CKABRT: BIT     #PABRT,VSTAT    ;ABORT FLAG RECEIVED?
                                        BEQ     2$              ;NO-EXIT
                                        MOV     R0,$GDDAT
                                        SUB     #4,$GDDAT       ;POINT ERR PC TO MAIN ROUTINE.
                                        MOV     VSTAT,$BDDAT
                                        ERROR   20              ;ISSUE PERIPHERAL ABORT ERROR

                                        MOV     TBBUF,R1
                                        JSR     R0,LDBUF        ;LOAD THE XMIT BUFFER WITH:
                                        .BYTE   .ESC,.0,.IABT,.ESC,.0,.RABT

                                        .BYTE   .ESC,.0,.UNLKKB,0

                                        MOV     #9.,XMCNT       ;SET UP TO XMIT 9 BYTES.
                                        JSR     R0,XMREC        ;XMIT AND RECEIVE.
                                        NOP
                                        CMPB    STR0,#NABRT     ;ABORT FLAG CLEARED?
                                        BEQ     2$              ;YES-EXIT
                                        MOV     R0,$GDDAT       ;NO-SET UP AND ISSUE A CANT
                                        SUB     #4,$GDDAT       ;CLEAR ABORT FLAG ERROR MESSAGE.
                                        MOV     VSTAT,$BDDAT
                                        ERROR   21
                                2$:     RTS     R0
```

```
                        ;;*****************************************************************
                        ;SUBROUTINE TO COMPARE RECEIVED KEYBOARD POSITION WITH
                        ;EXPECTED KEYBOARD POSITION. ERRORS ARE REPORTED
                        ;AS POSITIONAL ERRORS AND NOT DATA COMPARE ERRORS.
                        ;;*****************************************************************

020046  105077  011140        CKKBD:  CLRB    @ABUFP          ;CLEAR RECEIVE BYTE
020052  005037  002162                CLR     CHRD            ;CLEAR INPUT STORAGE.
020056  105777  011130        KBDLP:  TSTB    @ABUFP          ;WAIT FOR A INPUT.
020062  001775                        BEQ     .-4

020064  117737  011122  002162        MOVB    @ABUFP  ,CHRD   ;STORE IT AND
020072  105077  011114                CLRB    @ABUFP          ;CLEAR THE INPUT AREA.
020076  123714  002162        1$:     CMPB    CHRD    ,(R4)   ;RECEIVED EQUAL EXPECTED?
020102  001500                        BEQ     GDSTRK          ;NO-UPDATE POINTERS.
020104  005237  002210                INC     BUBCT           ;INCREMENT ERROR COUNT.
020110  023727  002210  000012        CMP     BUBCT,#10.      ;COUNT = 10?
020116  103075                        BHIS    CNTF            ;YES-EXIT SUBROUTINE.
020120  010401                        MOV     R4,R1
020122  166501  011572                SUB     DT↑BL(R5),R1    ;EXTRACT KEY POSITION FROM ROW LOC.
020126  005201                        INC     R1              ;CONVERT LOGICAL POS. TO ACTUAL.
020130  004037  017416                JSR     R0,BINOCT       ;GET KEY POSITION IN OCTAL.
020134  113737  002166  002164        MOVB    SVER2,SVER1     ;RE-ASSEMBLE OCTAL BYTES.
020142  123727  002165  000060        CMPB    SVER1+1,#60     ;POSITION LESS THAN 8?
020150  001413                        BEQ     LDPOS           ;YES-GO LOAD IT.
020152  123727  002164  000062        CMPB    SVER1,#62       ;POSITION GREATER THAN 8 AND LESS THAN12?
020160  103404                        BLO     BOROW           ;YES-SET UP TO BORROW.
020162  162737  000002  002164        SUB     #2,SVER1        ;NO-JUST SUBTRACT 2.
020170  000403                        BR      LDPOS
020172  162737  000370  002164  BOROW:  SUB     #370,SVER1      ;SUBTRACT AND BORROW.
020200  113737  002164  027623  LDPOS:  MOVB    SVER1,KYSTRK+1  ;LOAD THE CONVERTED DECIMAL #.
020206  113737  002165  027622        MOVB    SVER1+1,KYSTRK
020214  012703  027545        DMPOCT: MOV     #DKBERR ,R3
020220  004037  016226                JSR     R0      ,LDXMIT ;ISSUE BODY OF KEYBOARD ERROR.
020224  111401                        MOVB    (R4)    ,R1
020226  004037  017416                JSR     R0      ,BINOCT
020232  012703  002164                MOV     #SVER1  ,R3
020236  004037  016226                JSR     R0      ,LDXMIT ;CONVERT AND ISSUE GOOD CHAR.
020242  012703  027654                MOV     #DSPC6,R3
020246  004037  016226                JSR     R0,LDXMIT       ;INSERT 6 SPACES IN MESSAGE.
020252  113701  002162                MOVB    CHRD    ,R1
020256  004037  017416                JSR     R0      ,BINOCT
020262  012703  002164                MOV     #SVER1  ,R3
020266  004037  016226                JSR     R0      ,LDXMIT ;CONVERT AND ISSUE RECEIVED CHAR.
020272  012703  001171                MOV     #$CRLF  ,R3
020276  004037  016226                JSR     R0      ,LDXMIT ;ISSUE C/R AND L/F.
020302  000665                        BR      KBDLP           ;LOOK FOR SAME KEY AGAIN.

020304  005204        GDSTRK: INC     R4              ;INCREMENT KEYBOARD ROW COUNTER.
020306  105714                TSTB    (R4)            ;REACHED END OF ROW?
020310  001262                BNE     KBDLP           ;NO-LOOK FOR NEXT INPUT
020312  000200        CNTF:   RTS     R0              ;YES-EXIT.

                        ;;*****************************************************************
```

```
                                        ;SUBROUTINE TO LOOP DATA THROUGH HOST COMPUTER. ALL
                                        ;FUNCTIONS ARE ALLOWED, BUT BLOCK TRANSMITS WHICH
                                        ;EXCEED 552 BYTES WILL RESULT IN THE TERMINATION
                                        ;OF THE OPERATION AFTER 552 RECEIVED BYTES.

                                ;;**************************************************************

020314  005237  020616        LOOP:   INC     XMZER           ;SET UP TO XMIT NULLS.
020320  012737  031212  014766        MOV     #TCRLB+500,REBUF        ;RESET BUFFER POINTERS
020326  012737  030012  015272        MOV     #RCRLB,TBBUF
020334  004037  016266        JSR     R0,RESPTR       ;RELOAD ALL INTERRUPT POINTERS
020340  042737  077577  002224        BIC     #77577,VSTAT    ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
020346  013704  014770        LOOPT:  MOV     RBUFP,R4        ;SET UP RECEIVE FLAG
020352  032737  000001  002224  LOOPTA: BIT     #XMDNE,VSTAT    ;XMIT COMPLETE?
020360  001407                        BEQ     LOOPR           ;NO
020362  042737  000001  002224        BIC     #XMDNE,VSTAT    ;YES RESET FLAG
020370  013737  014764  014770        MOV     RBBUF,RBUFP     ;RESET THE REC. BUFFER POINTER
020376  000763                        BR      LOOPT
020400  032737  001400  002224  LOOPR:  BIT     #EPL+ESC,VSTAT  ;RECEIVED AN ESC OR EPL?
020406  001004                        BNE     LPSTR           ;YES-GO CHECK IT
020410  023704  014770                CMP     RBUFP,R4        ;RECEIVED A DISPLAY CHAR?
020414  001756                        BEQ     LOOPTA          ;NO-LOOP
020416  000426                        BR      BUMPCT
020420  117777  010566  174342  LPSTR:  MOVB    @ABUFP,@RBUFP   ;YES LOAD IT IN THE BUFFER
020426  005237  014770                INC     RBUFP           ;AND INCREMENT BUFFER POINTER
020432  005037  014772                CLR     ESAMB           ;CLEAR ESC ASSEMBLY WORD
020436  042737  001400  002224        BIC     #EPL+ESC,VSTAT  ;CLEAR THE FLAGS
020444  005237  015300                INC     XMCNT           ;INCREMENT XMIT COUNT
020450  123777  002132  010534        CMPB    ESCN,@ABUFP     ;CHAR. A ESC(033)?
020456  001733                        BEQ     LOOPT           ;YES WAIT FOR NEXT PART OF FUNCTION
020460  113777  001754  174302        MOVB    LNFED,@RBUFP    ;CHAR. WAS EPL ADD A LINE FEED.
020466  005237  014770                INC     RBUFP
020472  000407                        BR      FRCECT          ;AND ISSUE THEM.
020474  023727  015300  000764  BUMPCT: CMP     XMCNT,#500.     ;BUFFER ABOUT FILLED?
020502  103403                        BLO     FRCECT          ;NO
020504  005337  014770          1$:   DEC     RBUFP           ;YES-RESET THE RECEIVE POINTER
020510  000716                        BR      LOOPT
020512  005237  015300        FRCECT: INC     XMCNT           ;INCREMENT THE XMIT COUNT
020516  023727  015300  000002        CMP     XMCNT,#2        ;FIRST CHAR TO XMIT?
020524  101003                        BHI     XMWT            ;NO
020526  052777  000100  161202        BIS     #TENA,@VXCSR    ;YES-SET THE XMIT ENABLE
020534  004037  020544          XMWT: JSR     R0,EXTST        ;LOOK FOR END OF TEST COMMAND.
020540  000702                        BR      LOOPT           ;NONE FOUND.
020542  000200                        RTS     R0              ;AND EXIT

                                ;;**************************************************************
                                        ;SUBROUTINE TO CHECK FOR END OF TEST COMMAND. THE CONTROL
                                        ;C KEY EXITS ALL TESTS EXCEPT THE BLOCK MODE TEST
                                        ;WHICH IS EXITED ON A @ KEY.
                                ;;**************************************************************

020544  127727  010442  000003  EXTST:  CMPB    @ABUFP,#3       ;LOOK FOR CONTROL C.
020552  001020                        BNE     NOROUT

020554  012737  030511  014766  ABSXT:  MOV     #RCRLB+477,REBUF        ;RESET THE BUFFERS
020562  012737  030512  015272        MOV     #TCRLB,TBBUF
```

```
3715  020570  004037  016266                  JSR     R0,RESPTR       ;RESET ALL POINTERS
3716  020574  012702  026433                  MOV     #DEXT,R2
3717  020600  004037  017322                  JSR     R0,DSMES        ;ISSUE EXIT MESSAGE
3718  020604  005037  020616                  CLR     XMZER           ;CLEAR THE ZERO TRANSMIT FLAG.
3719  020610  062700  000002                  ADD     #2,R0           ;SET UP TEST EXIT.
3720  020614  000200                  NOROUT: RTS     R0              ;EXIT SUBROUTINE.
3721
3722  020616  000000                  XMZER:  .WORD   0
3723                                  ;;**************************************************************
3724                                          ;SUB-ROUTINE TO LOOK FOR VSTAT BIT ON THE STACK
3725                                          ;DELAY FACTOR IS FIRST WORD ON THE STACK AND VSTAT BIT
3726                                          ;IS THE SECOND. MIN. DELAY IS 4 U.S FOR A MOS 11/45.
3727                                  ;;**************************************************************
3728
3729  020620                          WTBGND:
  (2) 020620  012637  002222                  MOV     (SP)+,ROSVE     ;;POP STACK INTO ROSVE
3730  020624  012637  020706                  MOV     (SP)+,VDLAY     ;;POP STACK INTO VDLAY
3731  020630  012637  020704                  MOV     (SP)+,VBIT      ;;POP STACK INTO VBIT
3732  020634  005037  002220          1$:     CLR     DLAY
3733  020640  033737  020704  002224  2$:     BIT     VBIT,VSTAT      ;SENSED THE CONDITION?
3734  020646  001012                          BNE     FNDBT           ;YES-EXIT.
3735  020650  005337  002220                  DEC     DLAY            ;NO-RUN DELAY.
3736  020654  001371                          BNE     2$
3737  020656  005337  020706                  DEC     VDLAY           ;DELAY FACTOR EXPIRED?
3738  020662  001364                          BNE     1$              ;NO-LOOP
3739  020664  104011                          ERROR   11              ;DELAY EXPIRED-ISSUE HUNG NIT
3740  020666  005237  002200                  INC     FTLCNT          ;INCREMENT FATAL XMIT COUNT.
3741  020672  000401                          BR      TIMEXT
3742  020674  005720                  FNDBT:  TST     (R0)+           ;SET UP FOR NORMAL EXIT
3743  020676                          TIMEXT:
  (2) 020676  013746  002222                  MOV     ROSVE,-(SP)     ;;PUSH ROSVE ON STACK
3744  020702  000200                          RTS     R0
3745  020704  000000                  VBIT:   0
3746  020706  000000                  VDLAY:  0
3747                                  ;;**************************************************************
3748                                          ;SUBROUTINE TO LOOK FOR XOFF BEFORE EXITING A RECEIVE ROUTINE.
3749                                  ;;**************************************************************
3750
3751  020710  005037  002220          CKOFF:  CLR     DLAY
3752  020714  032737  100000  002224  1$:     BIT     #RXOFF,VSTAT    ;IS XOFF SET?
3753  020722  001403                          BEQ     2$              ;NO-EXIT
3754  020724  005337  002220                  DEC     DLAY            ;RUN DELAY.
3755  020730  001371                          BNE     1$
3756  020732  000200                  2$:     RTS     R0
3757
3758                                  .SBTTL  SCOPE HANDLER ROUTINE
  (1)
  (2)                             ;;**************************************************************
  (1)                             ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
  (1)                             ;*AND LOAD THE TEST NUMBER($TSTNM) 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)  020734                             $SCOPE:
(2)  020734  004037  013630             1$:     JSR     R0,MONIT
(1)  020740  032777  040000  160172             BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(1)  020746  001111                             BNE     $OVER           ;;YES IF SW14=1
(1)                                     ;*****START OF CODE FOR THE XOR TESTER*****
(1)  020750  000416                     $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                     ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)  020752  013746  000004                     MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  020756  012737  020776  000004             MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  020764  005737  177060                     TST     @#177060        ;;TIME OUT ON XOR?
(1)  020770  012637  000004                     MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  020774  000463                             BR      $SVLAD          ;;GO TO THE NEXT TEST
(1)  020776  022626                     5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  021000  012637  000004                     MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  021004  000423                             BR      7$              ;;LOOP ON THE PRESENT TEST
(1)  021006                             6$:;*****END OF CODE FOR THE XOR TESTER*****
(1)  021006  032777  000400  160124             BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
(1)  021014  001404                             BEQ     2$              ;;BR IF NO
(1)  021016  127737  160116  001102             CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
(1)  021024  001462                             BEQ     $OVER           ;;BR IF YES
(1)  021026  105737  001103             2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(1)  021032  001421                             BEQ     3$              ;;BR IF NO
(1)  021034  123737  001115  001103             CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
(1)  021042  101015                             BHI     3$              ;;BR IF NO
(1)  021044  032777  001000  160066             BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
(1)  021052  001404                             BEQ     4$              ;;BR IF NO
(1)  021054  013737  001110  001106     7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
(1)  021062  000443                             BR      $OVER
(1)  021064  105037  001103             4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  021070  005037  001160                     CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  021074  000415                             BR      1$              ;;ESCAPE TO THE NEXT TEST
(1)  021076  032777  004000  160034     3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(1)  021104  001011                             BNE     1$              ;;BR IF YES
(1)  021106  005737  001100                     TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  021112  001406                             BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  021114  005237  001104                     INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  021120  023737  001160  001104             CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  021126  002021                             BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  021130  012737  000001  001104     1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  021136  013737  021206  001160             MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  021144  105237  001102             $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  021150  011637  001106                     MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  021154  011637  001110                     MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  021160  005037  001162                     CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  021164  112737  000001  001115             MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  021172  013777  001102  157742     $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  021200  013716  001106                     MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  021204  000002                             RTI                     ;;FIXES PS
(1)  021206  000005                     $MXCNT: 5                       ;;MAX. NUMBER OF ITERATIONS
3759                                    .SBTTL  ERROR HANDLER ROUTINE
(1)
(2)                                     ;;**************************************************************
(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)  021210                             $ERROR:
(1)  021210  105237  001103             7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  021214  001775                             BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  021216  013777  001102  157716             MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  021224  032777  002000  157706             BIT     #BIT10,@SWR     ;;BELL ON ERROR?
(1)  021232  001402                             BEQ     1$              ;;NO - SKIP
(1)  021234  104401  001164                     TYPE    ,$BELL          ;;RING BELL
(1)  021240  005237  001112             1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
(1)  021244  011637  001116                     MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  021250  162737  000002  001116             SUB     #2,$ERRPC
(1)  021256  117737  157634  001114             MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  021264  032777  020000  157646             BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(1)  021272  001004                             BNE     20$             ;;SKIP TYPEOUTS
(1)  021274  004737  021576                     JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  021300  104401  001171                     TYPE    ,$CRLF
(1)  021304                             20$:
(1)  021304  005777  157630             2$:     TST     @SWR            ;;HALT ON ERROR
(1)  021310  100001                             BPL     3$              ;;SKIP IF CONTINUE
(1)  021312  000000                             HALT                    ;;HALT ON ERROR!
(1)  021314  032777  001000  157616     3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(1)  021322  001402                             BEQ     4$              ;;BR IF NO
(1)  021324  013716  001110                     MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(1)  021330  005737  001162             4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(1)  021334  001402                             BEQ     5$              ;;BR IF NONE
(1)  021336  013716  001162                     MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  021342                             5$:
(1)  021342  022737  010602  000042             CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
(1)  021350  001001                             BNE     6$              ;;BRANCH IF NO
(1)  021352  000000                             HALT                    ;;YES
(1)  021354                             6$:
(1)  021354  000002                             RTI                     ;;RETURN
3760                                    .SBTTL  TYPE ROUTINE
(1)
(2)                                     ;;**************************************************************
(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)  021356  105737  001157    $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)  021362  100002                    BPL     1$              ;;BR IF YES
(1)  021364  000000                    HALT                    ;;HALT HERE IF NO TERMINAL
(1)  021366  000407                    BR      3$              ;;LEAVE
(1)  021370  010046            1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)  021372  017600  000002            MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  021376  112046            2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  021400  001005                    BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  021402  005726                    TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  021404  012600            60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  021406  062716  000002    3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  021412  000002                    RTI                     ;;RETURN
(1)  021414  122716  000011    4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  021420  001430                    BEQ     8$
(1)  021422  122716  000200            CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  021426  001006                    BNE     5$
(1)  021430  005726                    TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  021432  104401                    TYPE                    ;;TYPE A CR AND LF
(1)  021434  001171                    $CRLF
(1)  021436  105037  021572            CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  021442  000755                    BR      2$              ;;GET NEXT CHARACTER
(1)  021444  004737  021526    5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  021450  123726  001156    6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  021454  001350                    BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  021456  013746  001154            MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                            ;;AND THE NULL CHAR.
(1)  021462  105366  000001    7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  021466  002770                    BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  021470  004737  021526            JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  021474  105337  021572            DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  021500  000770                    BR      7$              ;;LOOP
(1)
(1)                                     ;HORIZONTAL TAB PROCESSOR
(1)
(1)  021502  112716  000040    8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  021506  004737  021526    9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  021512  132737  000007  021572    BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  021520  001372                    BNE     9$              ;;TAB STOP
(1)  021522  005726                    TST     (SP)+           ;;POP SPACE OFF STACK
(1)  021524  000724                    BR      2$              ;;GET NEXT CHARACTER
(1)  021526  105777  157416    $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  021532  100375                    BPL     $TYPEC
(1)  021534  116677  000002  157410    MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  021542  122766  000015  000002    CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  021550  001003                    BNE     1$              ;;BRANCH IF NO
(1)  021552  105037  021572            CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  021556  000406                    BR      $TYPEX          ;;EXIT
(1)  021560  122766  000012  000002    CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  021566  001402                    BEQ     $TYPEX          ;;BRANCH IF YES
(1)  021570  105227            1$:     INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  021572  000000            $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(1)  021574  000207            $TYPEX: RTS     PC
(1)
3761                                    .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
```

```
(1)
(2)                                     ;;**********************************************************************
(1)                                     ;*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)
(1)  021576                             $ERRTYP:
(1)  021576  104401  001171                     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021602  010046                             MOV     R0,-(SP)        ;;SAVE R0
(1)  021604  005000                             CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  021606  153700  001114                     BISB    @#$ITEMB,R0
(1)  021612  001004                             BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                     ;;TYPE THE PC OF THE ERROR
(2)  021614  013746  001116                     MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                     ;;ERROR ADDRESS
(2)  021620  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  021622  000445                             BR      10$             ;;GET OUT
(1)  021624  005300                     1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  021626  006300                             ASL     R0              ;;      WORK FOR THE ERROR TABLE
(1)  021630  006300                             ASL     R0
(1)  021632  006300                             ASL     R0
(1)  021634  062700  001174                     ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  021640  012037  021650                     MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(1)  021644  001404                             BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  021646  104401                             TYPE                    ;;TYPE THE "ERROR MESSAGE"
(1)  021650  000000                     2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  021652  104401  001171                     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021656  012037  021666             3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
(1)  021662  001404                             BEQ     5$              ;;SKIP TYPEOUT IF 0
(1)  021664  104401                             TYPE                    ;;TYPE THE "DATA HEADER"
(1)  021666  000000                     4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
(1)  021670  104401  001171                     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021674  010146                     5$:     MOV     R1,-(SP)        ;;SAVE R1
(1)  021676  012001                             MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
(1)  021700  001415                             BEQ     9$              ;;BR IF NO DATA TO BE TYPED
(1)  021702  012000                             MOV     (R0)+,R0        ;;PICKUP "DATA FORMAT" POINTER
(1)  021704  105720                     6$:     TSTB    (R0)+           ;;"OCTAL" OR "DECIMAL"
(1)  021706  001003                             BNE     7$              ;;BR IF DECIMAL
(2)  021710  013146                             MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)  021712  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  021714  000402                             BR      8$
(1)  021716                             7$:
(2)  021716  013146                             MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)  021720  104405                             TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  021722  005711                     8$:     TST     (R1)            ;;IS THERE ANOTHER NUMBER?
(1)  021724  001403                             BEQ     9$              ;;BR IF NO
(1)  021726  104401  021746                     TYPE    ,11$            ;;TYPE TWO(2) SPACES
(1)  021732  000764                             BR      6$              ;;LOOP
(1)
(1)  021734  012601                     9$:     MOV     (SP)+,R1        ;;RESTORE R1
(1)  021736  012600                     10$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  021740  104401  001171                     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  021744  000207                             RTS     PC              ;;RETURN
(1)  021746  020040      000           11$:    .ASCIZ  /  /            ;;TWO(2) SPACES
(1)          021752                             .EVEN
3762                                    .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
```

```
(1)
(2)                             ;;*****************************************************************
(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)  021752  017646  000000     $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  021756  116637  000001  022175     MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  021764  112637  022177             MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  021770  062716  000002             ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  021774  000406                     BR      $TYPON
(1)  021776  112737  000001  022175 $TYPOC: MOVB  #1,$OFILL      ;;SET THE ZERO FILL SWITCH
(1)  022004  112737  000006  022177     MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  022012  112737  000005  022174 $TYPON: MOVB  #5,$OCNT       ;;SET THE ITERATION COUNT
(1)  022020  010346                     MOV     R3,-(SP)        ;;SAVE R3
(1)  022022  010446                     MOV     R4,-(SP)        ;;SAVE R4
(1)  022024  010546                     MOV     R5,-(SP)        ;;SAVE R5
(1)  022026  113704  022177             MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  022032  005404                     NEG     R4
(1)  022034  062704  000006             ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  022040  110437  022176             MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  022044  113704  022175             MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  022050  016605  000012             MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  022054  005003                     CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  022056  006105                 1$: ROL     R5              ;;ROTATE MSB INTO "C"
(1)  022060  000404                     BR      3$              ;;GO DO MSB
(1)  022062  006105                 2$: ROL     R5              ;;FORM THIS DIGIT
(1)  022064  006105                     ROL     R5
(1)  022066  006105                     ROL     R5
(1)  022070  010503                     MOV     R5,R3
(1)  022072  006103                 3$: ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  022074  105337  022176             DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  022100  100016                     BPL     7$              ;;BR IF NO
(1)  022102  042703  177770             BIC     #177770,R3      ;;GET RID OF JUNK
(1)  022106  001002                     BNE     4$              ;;TEST FOR 0
(1)  022110  005704                     TST     R4              ;;SUPPRESS THIS 0?
(1)  022112  001403                     BEQ     5$              ;;BR IF YES
(1)  022114  005204                 4$: INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

```
(1)  022116  052703  000060            BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  022122  052703  000040     5$:    BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  022126  110337  022172            MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  022132  104401  022172            TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  022136  105337  022174     7$:    DECB    $OCNT           ;;COUNT BY 1
(1)  022142  003347                    BGT     2$              ;;BR IF MORE TO DO
(1)  022144  002402                    BLT     6$              ;;BR IF DONE
(1)  022146  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  022150  000744                    BR      2$              ;;GO DO THE LAST DIGIT
(1)  022152  012605     6$:    MOV     (SP)+,R5                ;;RESTORE R5
(1)  022154  012604                    MOV     (SP)+,R4        ;;RESTORE R4
(1)  022156  012603                    MOV     (SP)+,R3        ;;RESTORE R3
(1)  022160  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  022166  012616                    MOV     (SP)+,(SP)
(1)  022170  000002                    RTI                     ;;RETURN
(1)  022172     000     8$:    .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
(1)  022173     000            .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
(1)  022174     000     $OCNT: .BYTE   0                       ;;OCTAL DIGIT COUNTER
(1)  022175     000     $OFILL:.BYTE   0                       ;;ZERO FILL SWITCH
(1)  022176  000000     $OMODE:.WORD   0                       ;;NUMBER OF DIGITS TO TYPE
3763                    .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
(1)
(2)                     ;;**************************************************************
(1)                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
(1)                     ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
(1)                     ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
(1)                     ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
(1)                     ;*REPLACED WITH SPACES.
(1)                     ;*CALL:
(1)                     ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
(1)                     ;*      TYPDS                   ;;GO TO THE ROUTINE
(1)
(1)  022200             $TYPDS:
(3)  022200  010046             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  022202  010146             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  022204  010246             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  022206  010346             MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  022210  010546             MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  022212  012746  020200     MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  022216  016605  000020     MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  022222  100004             BPL     1$              ;;BR IF INPUT IS POS.
(1)  022224  005405             NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  022226  112766  000055  000001  MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  022234  005000     1$:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  022236  012703  022414     MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  022242  112723  000040     MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  022246  005002     2$:    CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  022250  016001  022404     MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  022254  160105     3$:    SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  022256  002402             BLT     4$              ;;BR IF DONE
(1)  022260  005202             INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  022262  000774             BR      3$
(1)  022264  060105     4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  022266  005702             TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  022270  001002             BNE     5$              ;;FALL THROUGH IF 0
(1)  022272  105716             TSTB    (SP)            ;;STILL DOING LEADING 0'S?
```

```
(1)  022274  100407                        BMI     7$              ;;BR IF YES
(1)  022276  106316                5$:     ASLB    (SP)            ;;MSD?
(1)  022300  103003                        BCC     6$              ;;BR IF NO
(1)  022302  116663  000001  177777        MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)  022310  052702  000060        6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  022314  052702  000040        7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  022320  110223                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  022322  005720                        TST     (R0)+           ;;JUST INCREMENTING
(1)  022324  020027  000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)  022330  002746                        BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  022332  003002                        BGT     8$              ;;GO TO EXIT
(1)  022334  010502                        MOV     R5,R2           ;;GET THE LSD
(1)  022336  000764                        BR      6$              ;;GO CHANGE TO ASCII
(1)  022340  105726                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  022342  100003                        BPL     9$              ;;BR IF NO
(1)  022344  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  022352  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
(3)  022354  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  022356  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  022360  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  022362  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  022364  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  022366  104401  022414                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)  022372  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)  022400  012616                        MOV     (SP)+,(SP)
(1)  022402  000002                        RTI                     ;;RETURN TO USER
(1)  022404  023420                $DTBL:  10000.
(1)  022406  001750                        1000.
(1)  022410  000144                        100.
(1)  022412  000012                        10.
(1)  022414  000004                $DBLK:  .BLKW   4
3764                                       .SBTTL  POWER DOWN AND UP ROUTINES
  (1)
  (2)                                      ;;*****************************************************************
  (1)                                      ;POWER DOWN ROUTINE
  (1)  022424  012737  022564  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
  (1)  022432  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
  (3)  022440  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  022442  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)  022444  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (3)  022446  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (3)  022450  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (3)  022452  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  (3)  022454  017746  156460                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
  (1)  022460  010637  022570                MOV     SP,$SAVR6       ;;SAVE SP
  (1)  022464  012737  022476  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
  (1)  022472  000000                        HALT
  (1)  022474  000776                        BR      .-2             ;;HANG UP
  (1)
  (2)                                      ;;*****************************************************************
  (1)                                      ;POWER UP ROUTINE
  (1)  022476  012737  022564  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
  (1)  022504  013706  022570                MOV     $SAVR6,SP       ;;GET SP
  (1)  022510  005037  022570                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
  (1)  022514  005237  022570        1$:     INC     $SAVR6          ;;WAIT FOR THE INC
  (1)  022520  001375                        BNE     1$              ;;OF  WORD
```

```
(3)  022522  012677  156412            MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
(3)  022526  012605                    MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  022530  012604                    MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  022532  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  022534  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  022536  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  022540  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  022542  012737  022424  000024    MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1)  022550  012737  000340  000026    MOV     #340,@#PWRVEC+2 ;;PRIO:7
(1)  022556  104401                    TYPE                    ;REPORT THE POWER FAILURE
(1)  022560  022572            $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
(1)  022562  000002                    RTI
(1)  022564  000000            $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
(1)  022566  000776                    BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
(1)  022570  000000            $SAVR6: 0                       ;;PUT THE SP HERE
(1)  022572  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
(1)  022600  000122
(1)                                    .EVEN
3765                           .SBTTL  TRAP DECODER
(1)
(2)                            ;;**************************************************************
(1)                            ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
(1)                            ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1)                            ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)                            ;*GO TO THAT ROUTINE.
(1)
(1)  022602  010046            $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
(1)  022604  016600  000002            MOV     2(SP),R0        ;;GET TRAP ADDRESS
(1)  022610  005740                    TST     -(R0)           ;;BACKUP BY 2
(1)  022612  111000                    MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1)  022614  006300                    ASL     R0              ;;POSITION FOR INDEXING
(1)  022616  016000  022636            MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(1)  022622  000200                    RTS     R0              ;;GO TO ROUTINE
(1)
(1)
(1)                            ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
(1)
(1)  022624  011646            $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
(1)  022626  016666  000004  000002    MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
(1)  022634  000002                    RTI                     ;;RESTORE THE PSW
(1)
(3)                            .SBTTL  TRAP TABLE
(3)
(3)                            ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)                            ;*BY THE "TRAP" INSTRUCTION.
(3)
(3)                            ;       ROUTINE
(3)                            ;       -------
(3)  022636  022624            $TRPAD: .WORD   $TRAP2
(3)  022640  021356                    $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
(3)  022642  021776                    $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)  022644  021752                    $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)  022646  022012                    $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)  022650  022200                    $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
(1)
(1)
```

```
 3766
 3767  022652  003074  003462  003670  TSTADD: TST1,TST2,TST3
 3768
 3769  022660  004064  004216  004432          TST4,TST5,TST6
 3770
 3771  022666  004662  005436  006116          TST7,TST10,TST11
 3772
 3773  022674  006330  006504  006732          TST12,TST13,TST14
 3774
 3775  022702  007154  007654  010072          TST15,TST16,TST17
 3776
 3777  022710  010240  010636  011142          TST20,TST21,TST22
 3778
 3779  022716  011340  011412                  TST23,TST24
 3780
 3781
 3782
 3783  022722  042523  020124  052126  STUPM:  .ASCII  /SET VT61S TO  FULL DUPLEX, /<15><12>
       022730  030466  020123  047524
       022736  020040  052506  046114
       022744  042040  050125  042514
       022752  026130  006440     012
 3784  022757     071  030066  041060          .ASCIZ  /9600BAUD, REMOTE,PARITY MATCHED TO INTERFACE/<15><12>
       022764  052501  026104  051040
       022772  046505  052117  026105
       023000  040520  044522  054524
       023006  046440  052101  044103
       023014  042105  052040  020117
       023022  047111  042524  043122
       023030  041501  006505  000012
 3785
 3786
 3787
 3788  023036  005015  042101  051104  DUNTST: .ASCIZ  <15><12>/ADDRESSES WITH RESPONSIVE VT61S ARE:/<15><12>
       023044  051505  042523  020123
       023052  044527  044124  051040
       023060  051505  047520  051516
       023066  053111  020105  052126
       023074  030466  020123  051101
       023102  035105  005015     000
 3789  023107     015  047012  020117  NOVT:   .ASCIZ  <15><12>/NO VT61 RESPONDED TO ESCZ SEQ. AUTO RETRY IN 30 SEC./<15><12>
       023114  052126  030466  051040
       023122  051505  047520  042116
       023130  042105  052040  020117
       023136  051505  055103  051440
       023144  050505  020056  052501
       023152  047524  051040  052105
       023160  054522  044440  020116
       023166  030063  051440  041505
       023174  006456  000012
 3790
 3791
 3792  023200  005015  046104  030461  DLERR:  .ASCIZ  <15><12>/DL11 FAILED AT ADDRESS/
       023206  043040  044501  042514
       023214  020104  052101  040440
       023222  042104  042522  051523
```

```
                023230      000
3793
3794            023231      115  047101  040525  DMANA:  .ASCII  /MANUAL TEST SELECTED -/<15><12>
                023236   020114  042524  052123
                023244   051440  046105  041505
                023252   042524  020104  006455
                023260      012
3795            023261      105  052116  051105          .ASCIZ  /ENTER ADDRESSES OF VT61S TO BE TESTED/<15><12>
                023266   040440  042104  042522
                023274   051523  051505  047440
                023302   020106  052126  030466
                023310   020123  047524  041040
                023316   020105  042524  052123
                023324   042105  005015      000
3796
3797            023331      105  052116  051105  DMANB:  .ASCIZ  /ENTER TESTS TO BE RUN/<15><12>
                023336   052040  051505  051524
                023344   052040  020117  042502
                023352   051040  047125  005015
                023360      000
3798
3799            023361      101  020116  051505  EM1:    .ASCIZ  /AN ESC SEQ. TO THE VT61  FAILED - OCTAL EQUIV. IS:/<15><12>
                023366   020103  042523  027121
                023374   052040  020117  044124
                023402   020105  052126  030466
                023410   020040  040506  046111
                023416   042105  026440  047440
                023424   052103  046101  042440
                023432   052521  053111  020056
                023440   051511  006472  000012
3800            023446   042524  052123  020043  DH1:    .ASCIZ  /TEST#  ERR PC  BYTE 1+2 BYTE 3+4/<15><12>
                023454   042440  051122  050040
                023462   020103  041040  052131
                023470   020105  025461  020062
                023476   054502  042524  031440
                023504   032053  005015      000
3801
3802            023511      122  041505  044505  EM2:    .ASCIZ  /RECEIVE STATUS ERROR./<15><12>
                023516   042526  051440  040524
                023524   052524  020123  051105
                023532   047522  027122  005015
                023540      000
3803            023541      101  042104  020056  DH2:    .ASCIZ  /ADD.  STAT.  ERR.BITS  CHAR./<15><12>
                023546   051440  040524  027124
                023554   020040  051105  027122
                023562   044502  051524  020040
                023570   044103  051101  006456
                023576   000012
3804
3805            023600   047523  052106  040527  EM3:    .ASCIZ  /SOFTWARE (VSTAT) STATUS ERROR./<15><12>
                023606   042522  024040  051526
                023614   040524  024524  051440
                023622   040524  052524  020123
                023630   051105  047522  027122
                023636   005015      000
3806            023641      040  040520  051523  DH3:    .ASCIZ  / PASS#,  TEST#,  EXP.STAT,  ACT.STAT/<15><12>
```

```
                023646  026043  020040  042524
                023654  052123  026043  020040
                023662  054105  027120  052123
                023670  052101  020054  040440
                023676  052103  051456  040524
                023704  006524  000012
 3807
 3808   023710  042107  020056  040504  EM4:    .ASCIZ  /GD. DATA DOES NOT MATCH REC. DATA/<15><12>
        023716  040524  042040  042517
        023724  020123  047516  020124
        023732  040515  041524  020110
        023740  042522  027103  042040
        023746  052101  006501  000012
 3809   023754  042524  052123  020043  DH4:    .ASCIZ  /TEST# ,REC.CNT.,GD. DATA, REC. DATA/<15><12>
        023762  051054  041505  041456
        023770  052116  026056  042107
        023776  020056  040504  040524
        024004  020054  042522  027103
        024012  042040  052101  006501
        024020  000012
 3810                                    .EVEN
 3811
 3812   024022  054502  042524  020123  EM5:    .ASCIZ  /BYTES EXPECTED DOES NOT EQUAL BYTES RECEIVED/<15><12>
        024030  054105  042520  052103
        024036  042105  042040  042517
        024044  020123  047516  020124
        024052  050505  040525  020114
        024060  054502  042524  020123
        024066  042522  042503  053111
        024074  042105  005015     000
 3813   024101     102  052131  051505  DH5:    .ASCIZ  /BYTES EXP., BYTES REC./<15><12>
        024106  042440  050130  026056
        024114  041040  052131  051505
        024122  051040  041505  006456
        024130  000012
 3814
 3815   024132  052503  051522  051117  EM6:    .ASCIZ  /CURSOR POSITIONING ERROR/<15><12>
        024140  050040  051517  052111
        024146  047511  044516  043516
        024154  042440  051122  051117
        024162  005015     000
 3816   024165     107  020104  044514  DH6:    .ASCIZ  /GD LINE  GD COL.   BD LINE  BD COL/<15><12>
        024172  042516  020040  042107
        024200  041440  046117  020056
        024206  020040  042102  046040
        024214  047111  020105  041040
        024222  020104  047503  006514
        024230  000012
 3817
 3818   024232  044504  042522  052103  EM7:    .ASCIZ  /DIRECT CURSOR ADDRESSING FAILURE/<15><12>
        024240  041440  051125  047523
        024246  020122  042101  051104
        024254  051505  044523  043516
        024262  043040  044501  052514
        024270  042522  005015     000
 3819   024275     120  051501  021523  DH7:    .ASCIZ  /PASS#  TEST #  ERROR PC   /<15><12>
```

```
                024302  020040  042524  052123
                024310  021440  020040  051105
                024316  047522  020122  041520
                024324  020040  006440  000012
3820            024332  040520  051523  020043  DH10:   .ASCIZ  /PASS#  TEST#  BD.ROW  BD.COL/<15><12>
                024340  052040  051505  021524
                024346  020040  042102  051056
                024354  053517  020040  042102
                024362  041456  046117  005015
                024370     000
3821
3822            024371     114  051501  020124  EM11:   .ASCIZ  /LAST TRANSMISSION TO VT61 CAUSED UNIT TO FAIL-HANG./<15><12>
                024376  051124  047101  046523
                024404  051511  044523  047117
                024412  052040  020117  052126
                024420  030466  041440  052501
                024426  042523  020104  047125
                024434  052111  052040  020117
                024442  040506  046111  044055
                024450  047101  027107  005015
                024456     000
3823
3824            024457     126  033124  020061  EM12:   .ASCIZ  /VT61 UNDER TEST FAILED- ERROR DATA FOLLOWS/<15><12>
                024464  047125  042504  020122
                024472  042524  052123  043040
                024500  044501  042514  026504
                024506  042440  051122  051117
                024514  042040  052101  020101
                024522  047506  046114  053517
                024530  006523  000012
3825
3826            024534  052126  030466  043040  EM10:   .ASCIZ  /VT61 FAILED SELF TEST FUNCTION/<15><12>
                024542  044501  042514  020104
                024550  042523  043114  052040
                024556  051505  020124  052506
                024564  041516  044524  047117
                024572  005015     000
3827
3828
3829            024575     120  051501  021523  DH12:   .ASCIZ  /PASS#,  TEST#, GD.CKSUM, BD.CKSUM/<15><12>
                024602  020054  052040  051505
                024610  021524  020054  042107
                024616  041456  051513  046525
                024624  020054  042102  041456
                024632  051513  046525  005015
                024640     000
3830
3831            024641     124  051505  044524  DABRT:  .ASCIZ  /TESTING ABORTED-TOO MANY FATAL XMITS/<15><12>
                024646  043516  040440  047502
                024654  052122  042105  052055
                024662  047517  046440  047101
                024670  020131  040506  040524
                024676  020114  046530  052111
                024704  006523  000012
3832
3833            024710  052126  030466  051040  EM13:   .ASCIZ  /VT61 RECEIVER CHECKSUM COMPARE ERROR/<15><12>
```

```
                024716  041505  044505  042526
                024724  020122  044103  041505
                024732  051513  046525  041440
                024740  046517  040520  042522
                024746  042440  051122  051117
                024754  005015     000
3834
3835    024757     126  033124  020061  EM14:   .ASCIZ  /VT61 TRANSMITTER CHECKSUM COMPARE ERROR/<15><12>
                024764  051124  047101  046523
                024772  052111  042524  020122
                025000  044103  041505  051513
                025006  046525  041440  046517
                025014  040520  042522  042440
                025022  051122  051117  005015
                025030     000
3836
3837            025032                           .EVEN
3838    025032  047125  052111  052440  DVUNIT: .ASCII  /UNIT UNDER TEST /<15><12>
                025040  042116  051105  052040
                025046  051505  020124  005015
3839    025054  041522  051123  020040          .ASCIZ  /RCSR   VECT.   IDENT/<15><12>
                025062  053040  041505  027124
                025070  020040  044440  042504
                025076  052116  005015     000
3840    025103     040  041522  051123  DH11:   .ASCIZ  / RCSR   VECT./<15><12>
                025110  020040  053040  041505
                025116  027124  005015     000
3841    025123     120  044522  052116  DPRTR:  .ASCIZ  /PRINTER IS ATTACHED/<15><12>
                025130  051105  044440  020123
                025136  052101  040524  044103
                025144  042105  005015     000
3842    025151     103  050117  042511  DCOPYR: .ASCIZ  /COPIER IS ATTACHED/<15><12>
                025156  020122  051511  040440
                025164  052124  041501  042510
                025172  006504  000012
3843    025176  047530  043106  052040  EM15:   .ASCIZ  /XOFF TO VT61 FAILED TO HALT BLOCK XMIT/<15><12>
                025204  020117  052126  030466
                025212  043040  044501  042514
                025220  020104  047524  044040
                025226  046101  020124  046102
                025234  041517  020113  046530
                025242  052111  005015     000
3844    025247     130  047117  052040  EM16:   .ASCIZ  /XON TO VT61 FAILED TO RESTART BLOCK XMIT/<15><12>
                025254  020117  052126  030466
                025262  043040  044501  042514
                025270  020104  047524  051040
                025276  051505  040524  052122
                025304  041040  047514  045503
                025312  054040  044515  006524
                025320  000012
3845    025322  047516  054040  047117  EM17:   .ASCIZ  /NO XON RECEIVED WITHIN 3 SEC. AFTER A RESET/<15><12>
                025330  051040  041505  044505
                025336  042526  020104  044527
                025344  044124  047111  031440
                025352  051440  041505  020056
                025360  043101  042524  020122
```

```
                025366  020101  042522  042523
                025374  006524  000012
3846            025400  040514  052123  050040  EM20:   .ASCIZ  /LAST PERIPHERAL OPERATION ABORTED/<15><12>
                025406  051105  050111  042510
                025414  040522  020114  050117
                025422  051105  052101  047511
                025430  020116  041101  051117
                025436  042524  006504  000012
3847            025444  047503  046125  020104  EM21:   .ASCIZ  /COULD NOT CLEAR LAST ABORT FLAG./<15><12>
                025452  047516  020124  046103
                025460  040505  020122  040514
                025466  052123  040440  047502
                025474  052122  043040  040514
                025502  027107  005015     000
3848            025507     123  046517  047440  EM22:   .ASCIZ  /SOM OR EOM NOT RECEIVED DURING MAINT. MODE TRANSMIT/<15><12>
                025514  020122  047505  020115
                025522  047516  020124  042522
                025530  042503  053111  042105
                025536  042040  051125  047111
                025544  020107  040515  047111
                025552  027124  046440  042117
                025560  020105  051124  047101
                025566  046523  052111  005015
                025574     000
3849            025575     114  047111  020105  EM23:   .ASCIZ  /LINE FEED OR CURSOR RIGHT ISSUED FROM ROW 23 DID NOT CAUSE SCREEN TO SC
                025602  042506  042105  047440
                025610  020122  052503  051522
                025616  051117  051040  043511
                025624  052110  044440  051523
                025632  042525  020104  051106
                025640  046517  051040  053517
                025646  031040  020063  044504
                025654  020104  047516  020124
                025662  040503  051525  020105
                025670  041523  042522  047105
                025676  052040  020117  041523
                025704  047522  046114  005015
                025712     000
3850            025713     120  051501  020123  DH13:   .ASCIZ  /PASS ,   TEST ,   VSTAT/<15><12>
                025720  020054  020040  042524
                025726  052123  026040  020040
                025734  053040  052123  052101
                025742  005015     000
3851            025745     120  051501  026123  DH14:   .ASCIZ  /PASS,   TEST,   ERR PC,   VSTAT/<15><12>
                025752  020040  052040  051505
                025760  026124  020040  042440
                025766  051122  050040  026103
                025774  020040  053040  052123
                026002  052101  005015     000
3852
3853            026007     105  041523  000040  DESC:   .ASCIZ  /ESC /
3854
3855
3856
3857            026014  042513  041131  040517  DKYBD:  .ASCII  /KEYBOARD TEST/<15><12>
                026022  042122  052040  051505
```

```
                026030  006524  012
3858            026033  113     054505  052123          .ASCII  /KEYSTROKES ECHO:/<15><12>
                026040  047522  042513  020123
                026046  041505  047510  006472
                026054  012
3859            026055  101     042040  051511          .ASCII  /A DISPLAY CHAR. = A DISPLAY CHAR./<15><12>
                026062  046120  054501  041440
                026070  040510  027122  036440
                026076  040440  042040  051511
                026104  046120  054501  041440
                026112  040510  027122  005015
3860            026120  031463  036440  042440          .ASCII  /33 = ESC/<15><12>
                026126  041523  005015
3861            026132  032461  036440  041440          .ASCII  /15 = C-R/<15><12>
                026140  051055  005015
3862            026144  031061  036440  046040          .ASCII  /12 = L-F/<15><12>
                026152  043055  005015
3863            026156  033460  036440  041040          .ASCII  /07 = BELL/<15><12>
                026164  046105  006514  012
3864            026171  061     020060  020075          .ASCII  /10 = TAB/<15><12>
                026176  040524  006502  012
3865            026203  116     047117  042055          .ASCIZ  /NON-DISPLAY CHAR.= OCTAL EQUIV/<15><12>
                026210  051511  046120  054501
                026216  041440  040510  027122
                026224  020075  041517  040524
                026232  020114  050505  044525
                026240  006526  000012
3866
3867            026244  040524  020102  000     DTAB:   .ASCIZ  /TAB /
3868            026251  103     051055  000040  DCR:    .ASCIZ  /C-R /
3869            026256  026514  020106  000     DLF:    .ASCIZ  /L-F /
3870            026263  102     046105  020114  DBELL:  .ASCIZ  /BELL /
                026270  000
3871
3872            026271  114     047517  020120  DLOOP:  .ASCII  /LOOP TEST - LOOP COMMANDS AND DATA THRU/<15><12>
                026276  042524  052123  026440
                026304  046040  047517  020120
                026312  047503  046515  047101
                026320  051504  040440  042116
                026326  042040  052101  020101
                026334  044124  052522  005015
3873            026342  047510  052123  041040          .ASCII  /HOST BACK TO VT61 UNDER TEST. /<15><12>
                026350  041501  020113  047524
                026356  053040  033124  020061
                026364  047125  042504  020122
                026372  042524  052123  020056
                026400  005015
3874            026402  047503  052116  047522  DCNTZ:  .ASCIZ  /CONTROL C  EXITS TEST./<15><12>
                026410  020114  020103  042440
                026416  044530  051524  052040
                026424  051505  027124  005015
                026432  000
3875
3876            026433  105     044530  020124  DEXT:   .ASCIZ  /EXIT TEST./
                026440  042524  052123  000056
3877
```

```
3878  026446  051120  047111  042524  DPRNT:  .ASCII  /PRINTER TEST -/<15><12>
      026454  020122  042524  052123
      026462  026440  005015
3879  026466  031461  020062  047503          .ASCII  /132 COLUMNS OF A SLIDING PATTERN WILL BE/
      026474  052514  047115  020123
      026502  043117  040440  051440
      026510  044514  044504  043516
      026516  050040  052101  042524
      026524  047122  053440  046111
      026532  020114  042502
3880  026536  047503  052116  047111          .ASCII  /CONTINOUSLY OUTPUTTED TO PRINTER/<15><12>
      026544  052517  046123  020131
      026552  052517  050124  052125
      026560  042524  020104  047524
      026566  050040  044522  052116
      026574  051105  005015
3881  026600  040503  027122  051040  DCRST:  .ASCIZ  /CAR. RET. TO START/<15><12>
      026606  052105  020056  047524
      026614  051440  040524  052122
      026622  005015     000
3882
3883  026625     114  051501  020124  DEVERR: .ASCIZ  /LAST XMIT CAUSED VT61 HANG/<15><12>
      026632  046530  052111  041440
      026640  052501  042523  020104
      026646  052126  030466  044040
      026654  047101  006507  000012
3884  026662  000077                  QMRK:   .ASCIZ  /?/
3885  026664  051120  042117  041525  DKBD:   .ASCII  /PRODUCTION KEYBOARD TEST. 10 ERRORS CAUSES TEST EXIT./<15><12>
      026672  044524  047117  045440
      026700  054505  047502  051101
      026706  020104  042524  052123
      026714  020056  030061  042440
      026722  051122  051117  020123
      026730  040503  051525  051505
      026736  052040  051505  020124
      026744  054105  052111  006456
      026752     012
3886  026753     104  050105  042522          .ASCIZ  /DEPRESS KEYS FROM LEFT TO RIGHT/<15><12>
      026760  051523  045440  054505
      026766  020123  051106  046517
      026774  046040  043105  020124
      027002  047524  051040  043511
      027010  052110  005015     000
3887  027015     104  050105  042522  DLSHFT: .ASCIZ  /DEPRESS LEFT SHIFT KEY AND THE "A" KEY /<15><12>
      027022  051523  046040  043105
      027030  020124  044123  043111
      027036  020124  042513  020131
      027044  047101  020104  044124
      027052  020105  040442  020042
      027060  042513  020131  005015
      027066     000
3888  027067     104  050105  042522  DTOP:   .ASCIZ  /DEPRESS KEYS IN TOP ROW/<15><12>
      027074  051523  045440  054505
      027102  020123  047111  052040
      027110  050117  051040  053517
      027116  005015     000
```

```
 3889
 3890  027121  000104  050105  042522  DRSHFT: .ASCIZ  /DEPRESS RIGHT SHIFT KEY AND THE "A" KEY /<15><12>
       027126  051523  051040  043511
       027134  052110  051440  044510
       027142  052106  045440  054505
       027150  040440  042116  052040
       027156  042510  021040  021101
       027164  045440  054505  006440
       027172  000012
 3891  027174  042504  051120  051505  DSEC:   .ASCIZ  /DEPRESS KEYS IN SECOND ROW/<15><12>
       027202  020123  042513  051531
       027210  044440  020116  042523
       027216  047503  042116  051040
       027224  053517  005015  000000
 3892
 3893  027231  000104  050105  042522  DTHRD:  .ASCIZ  /DEPRESS KEYS IN THIRD ROW BEGINNING WITH 'A'/<15><12>
       027236  051523  045440  054505
       027244  020123  047111  052040
       027252  044510  042122  051040
       027260  053517  041040  043505
       027266  047111  044516  043516
       027274  053440  052111  020110
       027302  040447  006447  000012
 3894  027310  042504  051120  051505  DCONT:  .ASCIZ  /DEPRESS CONTROL KEY ,AND THE "A" KEY /<15><12>
       027316  020123  047503  052116
       027324  047522  020114  042513
       027332  020131  040454  042116
       027340  052040  042510  021040
       027346  021101  045440  054505
       027354  006440  000012
 3895  027360  042504  051120  051505  DBOT:   .ASCIZ  /DEPRESS KEYS IN FORTH ROW EXCEPT SHIFT KEYS/<15><12>
       027366  020123  042513  051531
       027374  044440  020116  047506
       027402  052122  020110  047522
       027410  020127  054105  042503
       027416  052120  051440  044510
       027424  052106  045440  054505
       027432  006523  000012
 3896  027436  042504  051120  051505  DSPCE:  .ASCIZ  /DEPRESS SPACE BAR/<15><12>
       027444  020123  050123  041501
       027452  020105  040502  006522
       027460  000012
 3897
 3898  027462  042504  051120  051505  DKPD:   .ASCIZ  /DEPRESS KEYPAD KEYS,LEFT TO RIGHT, TOP TO BOTTOM/<15><12>
       027470  020123  042513  050131
       027476  042101  045440  054505
       027504  026123  042514  052106
       027512  052040  020117  044522
       027520  044107  026124  052040
       027526  050117  052040  020117
       027534  047502  052124  046517
       027542  005015  000000
 3899
 3900  027545  000113  054505  047502  DKBERR: .ASCII  /KEYBOARD ERROR,KEY POSITION IN ROW SHOULD BE /
       027552  051101  020104  051105
       027560  047522  026122  042513
```

```
                027566  020131  047520  044523
                027574  044524  047117  044440
                027602  020116  047522  020127
                027610  044123  052517  042114
                027616  041040  020105
3901            027622  020040  005015          KYSTRK: .ASCII  /  /<15><12>
3902            027626  041517  040524  020114          .ASCIZ  /OCTAL GD, OCTAL BAD/<15><12>
                027634  042107  020054  041517
                027642  040524  020114  040502
                027650  006504  000012
3903            027654  020040  020040  020040  DSPC6:  .ASCIZ  /      /
                027662      000
3904
3905            027663      036      076      020  ROW1:   .BYTE   36,76,20,13,32,12,54,44,14,41,71,57,63,64,3,114,0
                027666      013      032      012
                027671      054      044      014
                027674      041      071      057
                027677      063      064      003
                027702      114      000
3906
3907            027704      026      056      030  ROW2:   .BYTE   26,56,30,73,52,22,55,34,24,31,51,77,62,61,2,0
                027707      073      052      022
                027712      055      034      024
                027715      031      051      077
                027720      062      061      002
                027723      000
3908
3909            027724      046      040      053  ROW3:   .BYTE   46,40,53,23,72,42,45,74,11,21,47,27,66,0
                027727      023      072      042
                027732      045      074      011
                027735      021      047      027
                027740      066      000
3910
3911            027742      016      070      060  ROW4:   .BYTE   16,70,60,50,33,43,25,35,75,65,37,115,67,0
                027745      050      033      043
                027750      025      035      075
                027753      065      037      115
                027756      067      000
3912
3913            027760      001      000          CNTRA:  .BYTE   01,0
3914
3915            027762      101      000          SHFTA:  .BYTE   101,0
3916
3917            027764      015      000          SPCB:   .BYTE   15,0
3918
3919            027766      113      004      103 KYPD:   .BYTE   113,04,103,104,1,112,101,102,6,7,106,100,5
                027771      104      001      112
                027774      101      102      006
                027777      007      106      100
                030002      005
3920            030003      010      105      107          .BYTE   10,105,107,110,17,111,0
                030006      110      017      111
                030011      000
3921
3922                                             .EVEN
3923            030012  000500                   RCRLB:  .BLKB   500     ;RECEIVE CIRCULAR BUFFER
```

```
                        
3924                    
3925  030512  000500    TCRLB:  .BLKB   500             ;TRANSMIT CIRCULAR BUFFER
3926  031212  000000    ABUFP:  .WORD   0
3927  031214  000062    ABBUF:  .BLKB   50.
3928  031276  000000            0
3929          000001            .END
```