

VT61, VT61T

DL11 VT61 EXER
CZVTHCO

AH-9524C-MC
FICHE 1 OF 1

JUN 1980
COPYRIGHT © 76.80
MADE IN USA



1
2

.REM %

IDENTIFICATION

PRODUCT CODE: AC-9522C-MC
PRODUCT NAME: CZVTHCO DL11 VT61 EXER
PRODUCT DATE: APRIL 1980
MAINTAINER: DIAGNOSTIC ENGINEERING

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

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

COPYRIGHT (C) 1976, 1980 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

TABLE OF CONTENTS

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

1. ABSTRACT

THIS PROGRAM IS AN ACCEPTANCE TEST FOR THE ENTIRE VT61 FAMILY OF TERMINALS. THE FUNCTIONAL TESTING IS BASED UPON A SET OF TERMINAL FUNCTIONS WHICH ARE COMMON THROUGHOUT THE ENTIRE FAMILY OF VT61 TYPE TERMINALS. THE FUNCTIONS AND THEIR DERIVED TESTING IS DESIGNED TO COMPLETELY CHECK (AT THE FUNCTIONAL LEVEL) THE TERMINAL MICRO-PROCESSOR AND ASSOCIATED RAMS. ALL TRANSMISSIONS TO THE VT61 WILL BE PRECEDED BY A SOM AND TERMINATED 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 XMTS' WILL BE DISP_AYED AND THE TEST WILL EXIT TO THE INITIAL SETUP SEQUENCE OF THE REQUESTED MODE. IF THE TEST THEN LOCATES AN OPERATIONAL UNIT, IT WILL BEGIN TESTING IT.

6.3 COMMON ERROR MESSAGES

A. ESCAPE SEQUENCE ERROR (ERROR 1)

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

B. RECEIVE STATUS ERROR (ERROR 2)

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

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

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

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

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

D. VT61 HUNG ERROR (ERROR 11)

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

7. RESTRICTIONS

- A. IT IS IMPERATIVE THAT BOTH THE INTERFACE AND THE VT61 SHOULD BE PLACED IN FULL DUPLEX AND REMOTE (NOT LOCAL) MODE.
- B. UNIT TO BE TESTED CANNOT BE THE CONSOLE DEVICE.
- C. FOR THE AUTOMATIC TEST MAPPING OF THE 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.

8. MISCELLANEOUS

- A. EXECUTION TIME FOR THE AUTO SELECTION TESTS (TEST 1-20) WITH UNITS SET TO A BAUD RATE OF 9600 BAUD IS APPROXIMATELY 90 SECONDS.
- B. TO TEST A DEVICE (DL11 WITH VT61 ATTACHED) AT NON-STANDARD ADDRESSES THE LOCATION "SRTAB" CAN BE MODIFIED TO CONTAIN THE LOWEST OF THE NON-STANDARD ADDRESSES AND LOCATION "ENDTAB" MODIFIED TO CONTAIN THE HIGHEST NON-STANDARD ADDRESS. ALL INTERFACES WITHIN THE NEW ADDRESSES WILL BE MAPPED AND TESTED IF THE PROPER RESPONSES ARE OBTAINED.
- C. TO CHANGE THE NUMBER OF FATAL ERRORS ALLOWED BEFORE TESTING IS ABORTED, LOCATION "ALWCNT" (LOADED WITH 10) CAN BE MODIFIED TO THE DESIRED COUNT.
- D. ALL TESTS EXCEPT TEST 1 AND TEST 23 ARE RUN IN MAINTENANCE MODE, THEREFORE ALL TRANSMISSIONS FROM THE VT61 ARE EXPECTED TO BE PRECEDED BY A SOM AND TERMINATED WITH A EOM.

9. PROGRAM DESCRIPTION

9.0 INITIALIZATION

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

9.1 TEST 1 CHECK ALL COMMON ESCAPE SEQUENCES.

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

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

9.2 TEST 2 CHECK MAINTENANCE MODE.

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

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

9.3 TEST 3 CHECK DIRECT CURSOR ADDRESSING

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

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

9.4 TEST 4 CHECK LINEAR ADDRESSING MODE.

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

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

9.5 TEST 5 CHECK XON/XOFF FROM VT61

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

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

9.6 TEST 6 CHECK XON/XOFF TO VT61

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

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

9.7 TEST 7 CHECK RAM AND COMMUNICATIONS PATHS

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

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

9.10 TEST 10 CHECK TRANSMIT AND RECEIVE CHECKSUMS.

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

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

9.11 TEST 11 CHECK BASIC CURSOR COMMANDS

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

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

9.12 TEST 12 CHECK READ CHARACTER AT CURSOR

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

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

9.13 TEST 13 CHECK REPLACE AND INSERT CHARACTER MODES

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

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

9.14 TEST 14 CHECK VT61 SCROLL CAPABILITIES.

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

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

9.15 TEST 15 CHECK ALL SCREEN ADDRESSES.

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

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

9.16 TEST 16 CHECK LINE FEED AND CARRIAGE RETURN

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

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

9.17 TEST 17 CHECK ERASE TO END OF SCREEN

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

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

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

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

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

END OF AUTO-ACCEPTANCE TESTS

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 EXPLICITLY DEFINED BY MESSAGES TO THE OPERATOR. 10 ERRORS WILL CAUSE AN AUTOMATIC EXIT FROM TEST.

10.0 CHANGE HISTORY

NOTE: HISTORY STARTS WITH REV. C0

C0: MODIFIED TO AUTOCK FOR CPU TYPE, AND INSERT APPROPRIATE DELAY VALUES FOR VARIOUS CPU EXECUTION TIMES.

```

654 .TITLE CZVTHCO DL11 VT61 EXER
655 ;*COPYRIGHT (C) 1976/1980
656 ;*DIGITAL EQUIPMENT CORP.
657 ;*MAYNARD, MASS. 01754
658 ;*
659 ;*PROGRAM BY DIAGNOSTIC ENGINEERING
660 ;*
661 ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAL
662 ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
663 ;*
664 .SBTTL OPERATIONAL SWITCH SETTINGS
665 ;*
666 ;*      SWITCH      USE
667 ;*      -----
668 ;*      15          HALT ON ERROR
669 ;*      14          LOOP ON TEST
670 ;*      13          INHIBIT ERROR TYPEOUTS
671 ;*      11          INHIBIT ITERATIONS
672 ;*      10          BELL ON ERROR
673 ;*      9           LOOP ON ERROR
674 ;*      8           LOOP ON TEST IN SWR<7:0>
675 .SBTTL BASIC DEFINITIONS
676 ;*
677 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
678 001100 STACK= 1100
679 .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
680 .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
681 ;*
682 ;*MISCELLANEOUS DEFINITIONS
683 000011 HT= 11          ;;CODE FOR HORIZONTAL TAB
684 000012 LF= 12          ;;CODE FOR LINE FEED
685 000015 CR= 15          ;;CODE FOR CARRIAGE RETURN
686 000200 CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
687 177776 PS= 177776     ;;PROCESSOR STATUS WORD
688 .EQUIV PS,PSW
689 177774 STKLMT= 177774  ;;STACK LIMIT REGISTER
690 177772 PIRQ= 177772    ;;PROGRAM INTERRUPT REQUEST REGISTER
691 177570 DSWR= 177570    ;;HARDWARE SWITCH REGISTER
692 177570 DDISP= 177570  ;;HARDWARE DISPLAY REGISTER
693 ;*
694 ;*GENERAL PURPOSE REGISTER DEFINITIONS
695 000000 R0= %0          ;;GENERAL REGISTER
696 000001 R1= %1          ;;GENERAL REGISTER
697 000002 R2= %2          ;;GENERAL REGISTER
698 000003 R3= %3          ;;GENERAL REGISTER
699 000004 R4= %4          ;;GENERAL REGISTER
700 000005 R5= %5          ;;GENERAL REGISTER
701 000006 R6= %6          ;;GENERAL REGISTER
702 000007 R7= %7          ;;GENERAL REGISTER
703 000006 SP= %6          ;;STACK POINTER
704 000007 PC= %7          ;;PROGRAM COUNTER
705 ;*
706 ;*PRIORITY LEVEL DEFINITIONS
707 000000 PR0= 0           ;;PRIORITY LEVEL 0
708 000040 PR1= 40          ;;PRIORITY LEVEL 1
709 000100 PR2= 100         ;;PRIORITY LEVEL 2

```

710	000140	PR3=	140	:::PRIORITY LEVEL 3
711	000200	PR4=	200	:::PRIORITY LEVEL 4
712	000240	PR5=	240	:::PRIORITY LEVEL 5
713	000300	PR6=	300	:::PRIORITY LEVEL 6
714	000340	PR7=	340	:::PRIORITY LEVEL 7

715
716 ;*SWITCH REGISTER* SWITCH DEFINITIONS

717	100000	SW15=	100000
718	040000	SW14=	40000
719	020000	SW13=	20000
720	010000	SW12=	10000
721	004000	SW11=	4000
722	002000	SW10=	2000
723	001000	SW09=	1000
724	000400	SW08=	400
725	000200	SW07=	200
726	000100	SW06=	100
727	000040	SW05=	40
728	000020	SW04=	20
729	000010	SW03=	10
730	000004	SW02=	4
731	000002	SW01=	2
732	000001	SW00=	1
733		.EQUIV	SW09,SW9
734		.EQUIV	SW08,SW8
735		.EQUIV	SW07,SW7
736		.EQUIV	SW06,SW6
737		.EQUIV	SW05,SW5
738		.EQUIV	SW04,SW4
739		.EQUIV	SW03,SW3
740		.EQUIV	SW02,SW2
741		.EQUIV	SW01,SW1
742		.EQUIV	SW00,SW0

743
744 ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

745	100000	BIT15=	100000
746	040000	BIT14=	40000
747	020000	BIT13=	20000
748	010000	BIT12=	10000
749	004000	BIT11=	4000
750	002000	BIT10=	2000
751	001000	BIT09=	1000
752	000400	BIT08=	400
753	000200	BIT07=	200
754	000100	BIT06=	100
755	000040	BIT05=	40
756	000020	BIT04=	20
757	000010	BIT03=	10
758	000004	BIT02=	4
759	000002	BIT01=	2
760	000001	BIT00=	1
761		.EQUIV	BIT09,BIT9
762		.EQUIV	BIT08,BIT8
763		.EQUIV	BIT07,BIT7
764		.EQUIV	BIT06,BIT6
765		.EQUIV	BIT05,BIT5

```

766      .EQUIV BIT04,BIT4
767      .EQUIV BIT03,BIT3
768      .EQUIV BIT02,BIT2
769      .EQUIV BIT01,BIT1
770      .EQUIV BIT00,BIT0
771
772      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
773      ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
774      RESVEC= 10        ;; RESERVED AND ILLEGAL INSTRUCTIONS
775      TBITVEC=14        ;; "T" BIT
776      TRIVEC= 14        ;; TRACE TRAP
777      BPTVEC= 14        ;; BREAKPOINT TRAP (BPT)
778      IOTVEC= 20        ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
779      PWRVEC= 24        ;; POWER FAIL
780      EMTVEC= 30        ;; EMULATOR TRAP (EMT) **ERROR**
781      TRAPVEC=34        ;; "TRAP" TRAP
782      TKVEC= 60         ;; TTY KEYBOARD VECTOR
783      TPVEC= 64         ;; TTY PRINTER VECTOR
784      PIRQVEC=240       ;; PROGRAM INTERRUPT REQUEST VECTOR
785      .SBTTL TRAP CATCHER
786
787      .-0
788      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
789      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
790      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
791      .=174
792      000174 000000      DISPREG: .WORD 0          ;; SOFTWARE DISPLAY REGISTER
793      000176 000000      SWREG:   .WORD 0          ;; SOFTWARE SWITCH REGISTER
794      .SBTTL ACT11 HOOKS
795
796      ;*****
797      ;HOOKS REQUIRED BY ACT11
798      $SVPC=.            ;SAVE PC
799      .=46
800      000046 010776      $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
801      .=52
802      000052 000000      .WORD 0        ;;2)SET LOC.52 TO ZERO
803      .=$SVPC          ;; RESTORE PC
804      .=200
805      000200 000137 002236      START:  JMP      AUTO          ;USE AUTO SELECTION OF UNITS
806      000204 000137 002270      MSTRT:  JMP      MANS          ;ALLOW OPERATOR SELECTION OF UNITS/TESTS
  
```

```

807      .SBTTL COMMON TAGS
808
809      ;*****
810      ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
811      ;*USED IN THE PROGRAM.
812
813      001100      .S1100
814      001100      $CMTAG:      ;:START OF COMMON TAGS
815      001100      $PASS:      .WORD 0      ;:CONTAINS PASS COUNT
816      001102      000      $TSTNM:      .BYTE 0      ;:CONTAINS THE TEST NUMBER
817      001103      000      $ERFLG:      .BYTE 0      ;:CONTAINS ERROR FLAG
818      001104      000000      $ICNT:      .WORD 0      ;:CONTAINS SUBTEST ITERATION COUNT
819      001106      000000      $LPADR:      .WORD 0      ;:CONTAINS SCOPE LOOP ADDRESS
820      001110      000000      $LPERR:      .WORD 0      ;:CONTAINS SCOPE RETURN FOR ERRORS
821      001112      000000      $ERTTL:      .WORD 0      ;:CONTAINS TOTAL ERRORS DETECTED
822      001114      000      $ITEMB:      .BYTE 0      ;:CONTAINS ITEM CONTROL BYTE
823      001115      001      $ERMAX:      .BYTE 1      ;:CONTAINS MAX. ERRORS PER TEST
824      001116      000000      $ERRPC:      .WORD 0      ;:CONTAINS PC OF LAST ERROR INSTRUCTION
825      001120      000000      $GDADR:      .WORD 0      ;:CONTAINS ADDRESS OF 'GOOD' DATA
826      001122      000000      $BDADR:      .WORD 0      ;:CONTAINS ADDRESS OF 'BAD' DATA
827      001124      000000      $GDDAT:      .WORD 0      ;:CONTAINS 'GOOD' DATA
828      001126      000000      $BDDAT:      .WORD 0      ;:CONTAINS 'BAD' DATA
829      001130      000000      .WORD 0      ;:RESERVED--NOT TO BE USED
830      001132      000000      .WORD 0
831      001134      000      $AUTOB:      .BYTE 0      ;:AUTOMATIC MODE INDICATOR
832      001135      000      $INTAG:      .BYTE 0      ;:INTERRUPT MODE INDICATOR
833      001136      000000      .WORD 0
834      001140      177570      SWR:      .WORD DSWR      ;:ADDRESS OF SWITCH REGISTER
835      001142      177570      DISPLAY:      .WORD DDISP      ;:ADDRESS OF DISPLAY REGISTER
836      001144      177560      $TKS:      177560      ;:TTY KBD STATUS
837      001146      177562      $TKB:      177562      ;:TTY KBD BUFFER
838      001150      177564      $TPS:      177564      ;:TTY PRINTER STATUS REG. ADDRESS
839      001152      177566      $TPB:      177566      ;:TTY PRINTER BUFFER REG. ADDRESS
840      001154      000      $NULL:      .BYTE 0      ;:CONTAINS NULL CHARACTER FOR FILLS
841      001155      002      $FILLS:      .BYTE 2      ;:CONTAINS # OF FILLER CHARACTERS REQUIRED
842      001156      012      $FILLC:      .BYTE 12      ;:INSERT FILL CHARS. AFTER A 'LINE FEED'
843      001157      000      $TPFLG:      .BYTE 0      ;:'TERMINAL AVAILABLE' FLAG (BIT<07> 0=YES)
844      001160      000000      $TIMES:      0      ;:MAX. NUMBER OF ITERATIONS
845      001162      000000      $ESCAPE:      0      ;:ESCAPE ON ERROR ADDRESS
846      001164      177607      $BELL:      .ASCIIZ <207><377><377>      ;:CODE FOR BELL
847      001170      077      $QUES:      .ASCII /?/      ;:QUESTION MARK
848      001171      015      $CRLF:      .ASCII <15>      ;:CARRIAGE RETURN
849      001172      000012      $LF:      .ASCIIZ <12>      ;:LINE FEED
850      ;*****
  
```

000377

```

851 .SBTTL ERROR POINTER TABLE
852
853 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
854 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
855 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
856 ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
857 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
858
859 ;* EM ;:POINTS TO THE ERROR MESSAGE
860 ;* DH ;:POINTS TO THE DATA HEADER
861 ;* DT ;:POINTS TO THE DATA
862 ;* DF ;:POINTS TO THE DATA FORMAT
863
864
865 $ERRTB:
866 ;GENERAL ESCAPE SEQUENCE ERROR MESSAGE
867
868 EM1 ;AN ESCAPE SEQUENCE TO VT61 FAILED.
869 DH1 ;TEST#,ERROR PC,2 SEQUENCE BYTES,2 SEQUENCE BYTES.
870 DT0
871 DF0
872
873 ;RECEIVE STATUS ERROR MESSAGE
874
875 EM2 ;RECEIVE STATUS ERROR
876 DH2 ;ADDRESS,STATUS ,ERR. BITS,CHAR.
877 DT2
878 DF0
879
880 ;RECIEVE SOFTWARE STATUS ERROR MESSAGE.
881
882 EM3 ;SFTWARE ($IAT) STATUS ERROR
883 DH3 ;PASS#,TEST#,GOOD STATUS,RECEIVED STATUS
884 DT4
885 DF6
886
887 ;DATA ERROR
888
889 EM4 ;DATA EXPECTED DOES NOT MATCH RECEIVE DATA.
890 DH4 ;TEST#,REC.CNT.,EXPECTED DATA, RECEIVE DATA
891 DT5
892 DF0
893
894 ;RECEIVE BYTE COUNT ERROR
895
896 EM5 ;BYTES EXPECTED DOES NOT EQUAL BYTES RECEIVED.
897 DH5 ;BYTES EXPECTED, BYTES RECEIVED
898 DT1
899 DF2
900
901 ;GENERAL DIRECT CURSOR ADDRESS FAILURE
902
903 EM6 ;CURSOR POSITION ERROR
904 DH6 ;GD LINE, GD COL., BD LINE, BAD COL.
905 DT2
906

```

907	001252	001476	DF3	
908				
909				
910				:DIRECT CURSOR ADDRESS ERROR
911	001254	024514	EM7	:DIRECT CURSOR ADDRESS ERROR
912	001256	024614	DH10	:PASS#,TEST#,BD. ROW,BD. COL.
913	001260	001502	DT4	
914	001262	001545	DF6	
915				
916				
917				:LAST TEST-SELF TEST FAILED
918				
919	001264	025016	EM10	:VT61 FAILED SELF-TEST FUNCTION
920	001266	025365	DH11	:CSR, VECTOR
921	001270	001436	DT1	
922	001272	001450	DF1	
923				
924				:VT61 FAIL/HUNG ERROR MESSAGE
925	001274	024653	EM11	:LAST TRANSMISSION TO VT61 CAUSED VT61 TO FAIL/HANG
926	001276	024557	DH7	:PASS#,TEST#,ERROR PC
927	001300	001466	DT3	
928	001302	001536	DF4	
929				
930				:GENERAL TEST FAILURE-PRECEEDS DATA/POSITION ERROR
931				
932	001304	024741	EM12	:VT61 UNDERR TEST FAILED-ERROR DATA FOLLOWS
933	001306	024557	DH7	:PASS#,TEST#,ERROR PC.
934	001310	001466	DT3	
935	001312	001536	DF4	
936				
937				:RECEIVE CHECKSUM ERROR
938				
939	001314	025172	EM13	:VT61 RECEIVER CHECKSUM ERROR
940	001316	025057	DH12	:PASS#,TEST#,GD.CKSUM,BD CKSUM
941	001320	001502	DT4	
942	001322	001545	DF6	
943				
944				:TRANSMITTER CHECKSUM ERROR
945				
946	001324	025241	EM14	:VT61 TRANSMITTER CHECKSUM ERROR
947	001326	025057	DH12	
948	001330	001502	DT4	
949	001332	001545	DF6	
950				
951				
952				
953				:XOFF FAILED TO HALT BLOCK XMIT
954				
955	001334	025460	EM15	:XOFF TO VT61 FAILED TO HALT BLOCK XMIT
956	001336	026175	DH13	:PASS,TEST,VSTAT
957	001340	001526	DT6	
958	001342	001536	DF4	
959				
960				:XON FAILED TO RESTART BLOCK XMIT
961				
962	001344	025531	EM16	:XON TO VT61 FAILED TO RESTART BLOCK XMIT

963	001346	026175				DM13	
964	001350	001526				DT6	
965	001352	001536				DF4	
966							
967							:NO XOM AFTER UNIT WAS RESET
968							
969	001354	025604				EM17	:NO XOM AFTER UNIT WAS RESET.
970	001356	024557				DM7	:PASS#,TEST#,ERROR PC
971	001360	001526				DT6	
972	001362	001536				DF4	
973							
974							:PERIPHERAL ABORT ERROR
975							
976	001364	025662				EM20	:LAST PERIPHERAL OPERATION ABORTED.
977	001366	026227				DM14	:PASS,TEST,ERROR PC, VSTAT
978	001370	001502				DT4	
979	001372	001545				DF6	
980							
981							:CANT CLEAR PERIPHERAL ABORT FLAG.
982							
983	001374	025726				EM21	:COULD NOT CLEAR LAST ABORT FLAG.
984	001376	026227				DM14	
985	001400	001502				DT4	
986	001402	001545				DF6	
987							
988							:MAINTENANCE MODE DID NOT FORCE A SOM/EOM.
989							
990	001404	025771				EM22	:SOM OR EOM NOT REC. IN MAINT. MODE.
991	001406	024122				DM3	:PASS#,TEST#,EXP.STAT, ACT.STAT
992	001410	001502				DT4	
993	001412	001545				DF6	
994							
995							:LINE FEED OR CURSOR RIGHT AT ROW 23 DID NOT CAUSE A SCROLL.
996							
997	001414	026057				EM23	:NO SCROLL FROM LINE FEED OR CURSOR RIGHT.
998	001416	024557				DM7	
999	001420	001526				DT6	
1000	001422	001536				DF4	
1001							
1002	001424	001102	001116	001124	DT0:	.WORD	\$TSTNM,\$ERRPC,\$GDDAT,\$BDDAT,0 ;(REV. CO)
1003	001432	001126	000000				
1004	001436	001124	001126	000000	DT1:	.WORD	\$GDDAT,\$BDDAT,0
1005	001444	000	000	000	DF0:	.BYTE	0,0,0,0
1006	001447	000					
1007	001450	000	000		DF1:	.BYTE	0,0
1008							
1009	001452	001	001		DF2:	.BYTE	1,1 ;DECIMAL TYPE
1010							
1011	001454	001120	001124	001122	DT2:	.WORD	\$GDADR,\$GDDAT,\$BDADR,\$BDDAT,0
1012	001462	001126	000000				
1013	001466	001100	001102	001116	DT3:	.WORD	\$PASS,\$TSTNM,\$ERRPC,0 ;(REV. CO)
1014	001474	000000					
1015	001476	001	001	001	DF3:	.BYTE	1,1,1,1
1016	001501	001					
1017	001502	001100	001102	001124	DT4:	.WORD	\$PASS,\$TSTNM,\$GDDAT,\$BDDAT,0 ;(REV. CO)
1018	001510	001126	000000				

```

1019 001514 001102 001120 001124 DT5: .WORD $STNM,$GDADR,$GDDAT,$BDDAT,0 ;(REV. C0)
1020 001522 001126 000000
1021 001526 001100 001102 001120 DT6: .WORD $PASS,$STNM,$GDADR,0 ;(REV. C0)
1022 001534 000000
1023 001536 001 000 000 DF4: .BYTE 1,0,0
1024 001541 000 000 001 DF5: .BYTE 0,0,1,1
1025 001544 001
1026 001545 001 000 000 DF6: .BYTE 1,0,0,0
1027 001550 000
1028 001552 .EVEN
1029 ;INSTRUCTION DEFINITIONS
1030 POP2SP =22626
1031 PUSH2SP =24646
1032
1033 ;*****
1034 ;DEFINITION SOFTWARE STATUS(VSTAT) REGISTER BITS
1035 ;*****
1036
1037 100000 RXOFF =100000 ;SET FOR XOFF, CLEARED FOR XON
1038 040000 RSOM =040000 ;SET FOR SOM (START OF MESSAGE).
1039 020000 REOM =020000 ;SET FOR EOM (END OF MESSAGE).
1040 010000 PABRT =010000 ;SET FOR A PERIPHERAL ABORT.
1041 004000 RSTT =004000 ;SET FOR RECEIVE STATUS ERROR.
1042 002000 CKSUM =002000 ;SET TO CALCULATE 61 REC. CHECKSUM
1043 001000 EPL =001000 ;SET WHEN END OF LINE DETECTED
1044 000400 ESC =000400 ;SET WHEN OCTAL 33 RECEIVED.
1045 000200 XMKIL =000200 ;SET WHEN TRANSMIT KILLED.
1046 000100 TXSUM =000100 ;SET TO CALCULATE 61 XMIT CHECKSUM
1047 000040 REVID =000040 ;SET WHEN REVERSE VIDEO MODE RECEIVED.
1048 000020 COMGP =000020 ;SET TO CONVERT REC. CHAR. BY -137.
1049 000004 CURPOS =000004 ;SET WHEN CURSOR POS. RECEIVED
1050 000002 TRMID =000002 ;SET WHEN TERMINAL I.D. RECEIVED.
1051 000001 XMDNE =000001 ;SET UPON TRANSMIT COMPLETE
1052
1053 ;*****
1054 ;DEFINITION OF DL11 CONTROL BITS
1055 ;*****
1056
1057 000200 RECDN =200
1058 100000 DSCHNG =100000
1059 000100 RDENA =000100
1060 100000 RERR =100000
1061 040000 RORUN =40000
1062 020000 RFMER =20000
1063 010000 RPAR =10000
1064 000200 TRDY =00200
1065 000100 TENA =00100
1066 000004 MAINT =00004
1067 000104 TCOMB =00104 ;COMBINATION INTERRUPT ENABLE AND MAINT.
1068
1069 003600 TOTCH =1920. ;TOTAL CHARACTERS ON SCREEN
1070 003601 TOTCI =1921. ;TOTAL SCREEN +1
1071 ;*****
1072 ;FOLLOW ARE DL11 ADDRESS AND VECTOR STORAGE TABLES
1073 ;*****
1074 001552 000020 VVECT: .BLKW 20 ;GOOD DL11 VECTOR TABLE

```

```

1075 001612 000020 DLTBL: .BLKW 20 ;GOOD DL11 ADDRESS TABLE
1076 001652 000020 INTAB: .BLKW 20 ;TABLE OF POSSIBLE DL11 ADDRESSES
1077
1078
1079 ;CURRENT POINTERS FOR ADDRESSES AND VECTORS
1080 001712 000000 VECPT: .WORD ;VECTOR INDEX
1081 001714 000000 DLTPT: .WORD ;ADDRESS INDEX
1082 ;ADDRESS TABLES FOR DL11 INTERFACES
1083 001716 176500 STRTAB: .WORD 176500 ;DL11A/B
1084 001720 175610 .WORD 175610 ;DL11 C/D/E
1085 001722 000000 .WORD 0
1086 001724 176676 ENDTAB: .WORD 176676 ;DL11 A/B
1087 001726 176170 .WORD 176170 ;DL11 C/D/E
1088 001730 000000 .WORD 0 ;END OF LIST MARKER
1089 ;*****
1090 ;VT61 ADDRESSES IN TABLE REFLECT UNIT UNDER TEST
1091 ;*****
1092 001732 000000 VRCSR: .WORD 0
1093 001734 000000 VRBUF: .WORD 0 ;RECEIVE DATA BUFFER
1094 001736 000000 VXCSR: .WORD 0 ;XMITTER CSR
1095 001740 000000 VXBUF: .WORD 0 ;XMITTER DATA BUFFER
1096 001742 000000 VECT: .WORD 0 ;VECTOR FOR UNIT UNDER TEST
1097 001744 000000 CRCSR: .WORD 0 ;CONSOLE RECEIVE CSR
1098 001746 000000 CRBUF: .WORD 0 ;CONSOLE DATA BUFFER
1099
1100 ;*****
1101 ;TABLE OF VT61 COMMAND AND SEQUENCES
1102 ;*****
1103
1104
1105 .BEL =007
1106 001750 000007 BEL: .WORD 007 ;BELL
1107 000015 .CARRT =015
1108 001752 000015 CARRT: .WORD 015 ;CARRIAGE RETURN
1109 000012 .LNFED =012
1110 001754 000012 LNFED: .WORD 012 ;LINE FEED
1111 000011 .TAB =011
1112 001756 000011 TAB: .WORD 011 ;TAB
1113 ;*****
1114 001760 000001 .WORD 01 ;TABLE DELIMITER (ESC/N)
1115 ;*****
1116
1117 .CHOM =110
1118 001762 000110 CHOM: .WORD 110 ;HOME CURSOR H
1119
1120 .CRT =103
1121 001764 000103 CRT: .WORD 103 ;CURSOR RIGHT C
1122
1123 .CDWN =102
1124 001766 000102 CDWN: .WORD 102 ;CURSOR DOWN B
1125
1126 .CLFT =104
1127 001770 000104 CLFT: .WORD 104 ;CURSOR LEFT D
1128
1129 .CUP =101
1130 001772 000101 CUP: .WORD 101 ;CURSOR UP A

```

1131					
1132		000112	.EOS	=112	
1133	001774	000112	EOS:	.WORD 112	;ERASE TO END OF SCREEN J
1134					
1135					
1136			::*****		
1137	001776	000002		.WORD 2	;TABLE DELIMITER (ESCO)
1138			::*****		
1139					
1140					
1141		000101	.EMAIN	=101	
1142	002000	000101	EMAIN:	.WORD 101	;ENTER MAINTENANCE MODE A
1143		000141	.DMAIN	=141	
1144	002002	000141	DMAIN:	.WORD 141	;EXIT MAINTENANCE MODE SA
1145					
1146		000105	.LKKB	=105	
1147	002004	000105	LKKB:	.WORD 105	;LOCK KEYBOARD E
1148		000145	.UNLKKB	=145	
1149	002006	000145	UNLKKB:	.WORD 145	;UNLOCK KEYBOARD SE
1150					
1151		000103	.DIRECT	=103	
1152	002010	000103	DIRECT:	.WORD 103	;ENABLE LINEAR MODE C
1153					
1154		000133	.CLRCK	=133	
1155	002012	000133	CLRCK:	.WORD 133	;CLEAR RECEIVER CHECKSUM I
1156					
1157		000134	.CLTCK	=134	
1158	002014	000134	CLTCK:	.WORD 134	;CLEAR TRANSMITTER CHECKSUM
1159					
1160					
1161		000112	.EEMP	=112	
1162	002016	000112	EEMP:	.WORD 112	;ENABLE REVERSE VIDEO J
1163		000152	.DEMP	=152	
1164	002020	000152	DEMP:	.WORD 152	;DISABLE REVERSE VIDEO SJ
1165					
1166		000137	.IABT	=137	
1167	002022	000137	IABT:	.WORD 137	;INITIALIZE ABORT FLAG -
1168					
1169			::*****		
1170	002024	000003		.WORD 3	;TABLE DELIMITER (ESCAPE P)
1171			::*****		
1172					
1173		000131	.EAPNT	=131	
1174	002026	000131	EAPNT:	.WORD 131	;ENABLE AUTO PRINT MODE V
1175		000171	.DAPNT	=171	
1176	002030	000171	DAPNT:	.WORD 171	;DISABLE AUTO PRINT MODE SY
1177					
1178		000111	.EINST	=111	
1179	002032	000111	EINST:	.WORD 111	;ENABLE INSERT I
1180		000151	.ERPL	=151	
1181	002034	000151	ERPL:	.WORD 151	;ENABLE REPLACE SI
1182					
1183			::*****		
1184					
1185	002036	000004		.WORD 4	;TABLE DELIMITER (I/O)
1186			::*****		

1187					
1188		054433	.DCRAD	=054433	
1189	002040	054433	DCRAD:	.WORD 054433	;DIRECT CURSOR ADDRESSING
1190		067467	.R23C79	=067467	
1191	002042	067467	R23C79:	.WORD 067467	;CURSOR TO LOWER RIGHT
1192	002044	000000		.WORD 0	
1193					
1194	002046	047433	RCUR:	.WORD 047433	;DIRECT CURSOR ADDRESSING
1195		000131	.Y	=131	
1196		000131	.RDCUR	=00131	
1197	002050	000131	RDCUR:	.WORD 00131	;READ CURSOR POSITION Y
1198	002052	000000		.WORD 0	
1199					
1200		000117	.O	=117	
1201	002054	047433	ESCO:	.WORD 047433	;ESCAPE O
1202		000126	.XMTAL	=000126	
1203	002056	000126	XMTAL:	.WORD 000126	;TRANSMIT ALL V
1204	002060	000000		.WORD 0	
1205					
1206	002062	047433		.WORD 047433	;ESCAPE O
1207		000127	.TCUCH	=127	
1208	002064	000127	TCUCH:	.WORD 127	;XMIT CHARACTER AT CURSOR. W
1209	002066	000000		.WORD 0	
1210					
1211	002070	047433		.WORD 047433	;ESCAPE O
1212		000135	.TXRCK	=135	
1213	002072	000135	TXRCK:	.WORD 135	;XMIT RECIEVER CHECKSUM]
1214	002074	000000		.WORD 0	
1215					
1216	002076	047433		.WORD 047433	;ESCAPE O
1217		000136	.TXTCK	=136	
1218	002100	000136	TXTCK:	.WORD 136	;XMIT TRANSMITTER CHECKSUM
1219	002102	000000		.WORD 0	
1220					
1221	002104	147433		.WORD 147433	;ESCAPE O
1222		000140	.RABT	=140	
1223	002106	000140	RABT:	.WORD 140	;READ THE ABORT FLAG. \
1224	002110	000000		.WORD 0	
1225					
1226			;*****		
1227	002112	177777		.WORD -1	;END OF TABLE TERMINATOR
1228			;*****		
1229			;*****		
1230			;*****		
1231			;PERIPHERAL COMMANDS		
1232			;*****		
1233			;*****		
1234		000127	.EPNT	=127	
1235	002114	000127	EPNT:	.WORD 127	;ENABLE PRINT MODE. W
1236		000130	.DPNT	=130	
1237	002116	000130	DPNT:	.WORD 130	;DISABLE PRINT MODE X
1238					
1239		000135	.CPYSC	=135	;COPY SCREEN]
1240		000136	.ENAC	=136	;ENABLE AUTO COPY MODE ESC ^
1241		000137	.DISAC	=137	;DISABLE AUTO COPY MODE ESC -
1242		000150	.PSCRN	=000150	;PRINT THE SCREEN H/SH

```

1243
1244
1245      ;*****
1246      ;ESCAPE CODE EQUIVALENCES AND IDENTIFIERS
1247      ;*****
1248
1249      000033      .ESC      =033      ;PRIMARY ESCAPE CODE.
1250      000120      .P        =120
1251      002120      ESCP:     .WORD    050033      ;ESCAPE P
1252      000124      .TSTER    =124
1253      002122      TSTER:     .WORD    124      ;TEST TERMINAL(ESC O T)
1254      002040      ESCYI      =DCRAD      ;ESCYI EQUALS DCRAD/DCRADI
1255      000057      SLSH       =000057      ;SLASH CODE FOR TERMINAL IDENT ESC.
1256      000106      CKGP       =106      ;ENABLE REC.TO SUB 137 FROM ALL REC DATA
1257      000107      NCKGP      =107      ;ENABLE NORMAL RECEIVED DATA.
1258      000171      CPABRT     =171      ;COPIER ABORT
1259      000172      PRABRT     =172      ;PRINTER ABORT
1260      000170      NABRT      =170      ;NO ABORT SX
1261      002124      IDENT:     .WORD    0      ;VT61 IDENT CODE
1262      002054      ESCOI      =ESCO
1263      002120      ESCP:     =ESCP
1264      002126      ESCZI      =ESCZ
1265      055033      .ESCZ      =055033
1266      002126      ESCZ:     .WORD    055033      ;OCTAL EQUIV. OF ESZ SEQUENCE
1267      000122      .RESET     =122
1268      002130      RESET:     .WORD    122      ;VT61 INITIALIZE R
1269
1270      002132      ESCN:     .WORD    000033      ;ESCAPE N-FLAG
1271      002134      R01C00:     .WORD    020041      ;ROW1,COL. 0
1272      002136      R01C20:     .WORD    032041      ;ROW01,COLUMN 20
1273      002140      R22C00:     .WORD    020066      ;ROW22,COL.00
1274      002142      R12C00:     .WORD    020054      ;ROW 12,COLUMN 00
1275      020067      .R23C00    =020067
1276      002144      R23C00:     .WORD    020067      ;ROW23,COL.00
1277      025440      .R00C11    =025440
1278      002146      R00C11:     .WORD    025440      ;ROW,COL.11
1279      032040      .R00C20    =032040
1280      002150      R00C20:     .WORD    032040      ;ROW 0,COLUMN 20
1281      002152      R00C08:     .WORD    024040      ;ROW 00,COLUMN 8
1282      002154      CUHME:     .WORD    020040      ;OCTAL EQUIV. OF CURSOR HOME.
1283      002156      R00C80:     .WORD    067440      ;ROW 0,COLUMN 80.
1284      002160      R23C78:     .WORD    067067      ;ROW 23,COL. 78.
1285      000040      .R00       =40      ;ROW 0
1286      000041      .R01       =41      ;ROW 1
1287      000054      .R12       =54      ;ROW 12
1288      000066      .R22       =66      ;ROW 22
1289      000067      .R23       =67      ;ROW 23
1290      000040      .C00       =40      ;COLUMN 0
1291      000043      .C03       =43      ;COL. 3
1292      000050      .C08       =50      ;COL. 8
1293      000053      .C11       =53      ;COL. 11
1294      000064      .C20       =64      ;COL. 20
1295      000065      .C21       =65      ;COL. 21
1296      000110      .C40       =110     ;COL. 40
1297      000157      .C79       =157     ;COL. 79
1298

```

```

1299      ;*****
1300      ;TEMPORARY STORAGE LOCATIONS AND
1301      ;SPECIAL RECEIVE CODE EQUIVALENCES.
1302      ;*****
1303      SOM      =02      ;START OF MESSAGE
1304      EOM      =04      ;END OF MESSAGE
1305      XOFF     =23      ;TURN OFF TRANSMISSION
1306      XON      =21      ;TURN ON TRANSMISSION
1307      CHRD:    .WORD    0      ;STORAGE FOR SINGLE CH. READ
1308      SVER1:   .WORD
1309      SVER2:   .WORD
1310      ZERO:    .WORD    0      ;TEMP. STORAGE R1.
1311      TYP6:    .WORD    3000   ;TEMP. STORAGE R2.
1312      TSTPTR:  .WORD    0      ;MUST BE LEFT AS ZERO.
1313      MODE:    .WORD    0      ;TYPE 6 OCTAL CHAR-NO ZEROS
1314      FTLCNT:  .WORD    0      ;TEST POINTER IN MANUAL SELECT MODE
1315      ALWCNT:  .WORD    10.    ;BYTE0=TESTING MODE,BYTE1=INTERFACE TYPE
1316      ONE:     .WORD    1      ;COUNT OF INCOMPLETE XMITS.
1317      TOADD:   .WORD
1318      BUBCT:   .WORD
1319      TPREG:   0      ;# OF ALLOWABLE INCOMPLETE XMITS.
1320      PRESC:   .WORD
1321      ESSEQ:   .WORD      ;PRIMARY ESC COMMAND
1322      DLAY:    .WORD      ;SEQUENCE ASSEMBLY AREA
1323      ROSVE:   .WORD
1324      VSTAT:   .WORD    0      ;TEMP STORAGE FOR R0 ONLY.
1325      BLKM:    .WORD    0      ;FLAG LOCATION FOR BLOCK MODE XMITS.
1326      TSTNM:   .WORD    0      ;DISPLAY STORAGE FOR TEST NUMBER.
1327      PTYPE:   .WORD    0      ;PROCESSOR TYPE INDICATOR      ;:++C
1328      ;IF 0 THEN 11/45-11/70
1329      ;IF -1 THEN 11/35,40
1330      ;IF 1 THEN 11/04-11/34
1331      DLYCNT:  .WORD    0      ;USED IN WTBGND WAIT LOOP      ;:++C
1332
1333      ;*****
1334      ;AUTOMATIC SELECTION OF UNITS. TESTS 1 THROUGH 33 WILL BE
1335      ;REPITIVELY EXECUTED FOR ALL UNITS.
1336      ;*****
1337
1338      AUTO:    CLR      MODE      ;ZERO THE MODE SWITCH
1339      ;JMP      SETA      ;DO VECTOR SETUP
1340      AUTOA:   JSR      RO,TRPVEC ;GO FIND GOOD DL11S
1341      ;JSR      RO,CDEV    ;CHECK DL11S FOUND
1342      ;JSR      RO,INITA   ;INSURE VT61S ON DL11
1343      ;JMP      MODCK     ;VT61 PRESENT -BEGIN TESTING
1344      ;BR       AUTOA     ;NO VT61 FOUND LOOP IN CHECKING
1345
1346      ;*****
1347
1348      ;MANUAL UNIT AND TEST SELECTION. UNITS CAN BE
1349      ;SELECTED VIA CONSOLE OR AUTO SELECTION CAN
1350      ;BE UTILIZED. TESTS ENTERED VIA CONSOLE WILL
1351      ;BE EXECUTED IN THE ORDER ENTERED.
1352
1353      ;*****
1354

```

```

1355
1356 002270 012737 000001 002176 MANS: MOV #1,MODE ;SET MODE TO MANUAL SELECT.
1357 002276 000137 012012 JMP SETA ;GO SET UP CONSTANTS
1358 002302 104401 023512 MANSA: TYPE ,DMANA
1359 002306 004037 012470 JSR RO,TRPVEC ;FIND GOOD DL11'S
1360 002312 012703 001652 MOV #INTAB,R3
1361 002316 005002 BLDADD: CLR R2 /
1362
1363 002320 004037 020014 BLDADA: JSR RO,GTNUM ;GET A KEYBOARD INPUT
1364 002324 120127 000054 CMPB R1,#54 ;CHAR. = COMMA?
1365 002330 001004 BNE 1$ ;NO
1366 002332 004037 012424 JSR RO,TMNA ;YES, VERIFY ENTERED ADDRESS,
1367 002336 010223 MOV R2,(R3)+ ;STORE THIS ADDRESS,
1368 002340 000766 JSR BLDADD ;AND LOOK FOR ANOTHER ADDRESS.
1369 002342 120137 001754 1$: MPB R1,LFED ;CHAR. = LINE FEED?
1370 002346 001024 BNE 3$ ;NO
1371 002350 005702 TST R2 ;ANY ENTRIES CREATED?
1372 002352 001413 BEQ 2$ ;NO USE AUTO SELECTION OF UNITS
1373 002354 004037 012424 JSR RO,TMNA ;YES, VERIFY ENTERED ADDRESS,
1374 002360 010223 MOV R2,(R3)+ ;STORE LAST ADDRESS,
1375 002362 013723 002170 MOV ZERO,(R3)+ ;AND SET A TERMINATOR IN TABLE.
1376 002366 004037 012606 JSR RO,CDEV ;CHECK DL11 ON VT 61 SELECTED
1377 002372 005737 001612 TST DL1BL ;ANY DL11S GOOD?
1378 002376 001741 BEQ MANSA ;NO-BACK TO SQUARE ONE
1379 002400 000412 BR BLDTST ;YES- GO GET TESTS
1380 002402 004037 012606 2$: JSR RO,CDEV ;CHECK DL11'S
1381 002406 004037 013164 JSR RO,INITA ;VERIFY DL11 HAVE VT61 ATTACHED
1382 002412 000137 002426 JMP BLDTST ;BEGIN TEST SELECTION
1383 002416 000731 BR MANSA ;NO UNIT FOUND-LOOP
1384 002420 004037 017750 3$: JSR RO,OCTBIN ;KEEP BUILDING ADDRESS
1385 002424 000735 BR BLDADA
1386
1387 002426 104401 023612 BLDTST: TYPE ,DMANB ;TYPE 2ND PART OF MANUAL MESSAGE
1388 002432 012703 001652 MOV #INTAB,R3 ;USE INTAB AS TEST # STORAGE.
1389 002436 005004 CLR R4 ;CLEAR TEST COUNTER
1390 002440 005002 11$: CLR R2 ;CLEAR ASSEMBL WORD
1391 002442 004037 020014 10$: JSR RO,GTNUM ;GET A NUMERIC CHAR.
1392 002446 120127 000054 CMPB R1,#54 ;CHAR.=COMMA?
1393 002452 001006 BNE 1$ ;NO
1394 002454 110223 MOV R2,(R3)+ ;YES STORE A TEST #
1395 002456 005204 INC R4 ;AND INCREMENT TEST COUNT.
1396 002460 020437 000040 CMP R4,32. ;COUNT =32?
1397 002464 001415 BEQ MODCK ;YES ACCEPT NO MORE ENTRIES.
1398 002466 000764 BR 11$ ;NO KEEP LOOKING
1399 002470 120137 001754 1$: CMPB R1,LFED ;CHAR. = LINE FEED?
1400 002474 001006 BNE 2$ ;NO
1401 002476 110223 MOV R2,(R3)+ ;LOAD THE LAST TEST
1402 002500 105013 CLRB (R3) ;AND INSERT TEST TABLE TERMINATOR
1403 002502 112737 000001 002176 MOVB #1,MODE ;SET MODE SWITCH TO MANUAL
1404 002510 000403 BR MODCK ;AND BEGIN TESTING.
1405
1406 002512 004037 017750 2$: JSR RO,OCTBIN ;CONVERT CHAR.
1407 002516 000751 BR 10$
1408
1409 ;*****
1410 ;THIS ROUTINE LOOKS FOR THE OPERATIONAL MODE REQUESTED AND

```

```

1411                                     ;SELECTS THE NEXT UNIT TO BE TESTED.
1412
1413                                     ;MODE 0 = ACCEPTANCE TYPE TEST
1414                                     ;MODE 1 = OPERATOR SELECTION OF UNITS AND SEQUENCE OF TESTS.
1415                                     ;:*****
1416
1417 002521 012737 001612 001714 MODCK: MOV #DLTBL,DLTPT ;INITIAL SETUP OF ADDRESS
1418 002526 012737 001552 001712 MODCA: MOV #VVECT,VECT ;AND VECTOR POINTERS.
1419 002534 012701 001732 MODCA: MOV #VRCSR,R1 ;LOAD ADDRESS DESTINATION
1420 002540 013702 001714 MODCA: MOV DLTPT,R2 ;LOAD CURRENT ADDRESS POINTER
1421 002544 017703 177142 MODCA: MOV @VECT,R3 ;LOAD CURRENT VECTOR POINTER
1422 002550 005712 MODCA: TST (R2) ;ALL UNITS CHECKED?
1423 002552 001013 MODCA: BNE 1$ ;NO - CONTINUE
1424 002554 005737 002176 MODCA: TST MODE ;CHECK MODE
1425 002560 001002 MODCA: BNE 10$
1426 002562 000137 002246 MODCA: JMP AUTOA ;GO RESTART AUTO MODE
1427 002566 105777 177402 10$: TSTB @TSTPTR ;MANUAL LOOP REQUESTED?
1428 002572 100001 10$: BPL 2$ ;NO
1429 002574 000751 10$: BR MODCK ;YES-RESTART COMPLETE TEST.
1430 002576 000137 002302 2$: JMP MANSA ;GO RESTART MANUAL MODE
1431 002602 004037 013404 1$: JSR R0,LDADD ;NO-LOAD NEXT ADDRESSES
1432 002606 010337 001742 1$: MOV R3,VECT ;STORE VECT. OF UNIT UNDER TEST
1433 002612 012723 014312 1$: MOV #INTRC,(R3)+ ;YES - NOW SET UP RECEIVE VECTOR
1434 002616 012723 000340 1$: MOV #340,(R3)+ ;AND SET RECEIVER PSW TO 7
1435 002622 012723 015234 1$: MOV #INTXM,(R3)+ ;SET UP TRANSMIT VECTOR
1436 002626 012723 000340 1$: MOV #340,(R3)+ ;AND SET PSW TO 7.
1437 002632 005046 1437: CLR -(SP) ;CLEAR THE PSW,LSI11 STYLE.
1438 002634 012746 002642 1438: MOV #100$,-(SP)
1439 002640 000002 1439: RTI
1440 002642 010237 001714 100$: MOV R2,DLTPT ;SAVE ADDRESS POINTER.
1441 002646 012737 031101 015176 100$: MOV #RCRLB+477,REBUF ;SET UP END OF BUFFER
1442 002654 012737 031601 015504 100$: MOV #TCRLB+477,TEBUF
1443 002662 012737 030402 015174 100$: MOV #RCRLB,RBBUF ;INITIALIZE REC.BUFFER.
1444 002670 012737 031102 015502 100$: MOV #TCRLB,TBBUF ;INITIALIZE TRANSMIT BUFFER.
1445 002676 004037 016520 100$: JSR R0,RESPTR ;RESET INTERRUPT POINTERS.
1446 002702 005037 002226 100$: CLR BLKM ;CLEAR BLOCK MODE FLAG.
1447 002706 005037 021074 100$: CLR XMZER ;CLEAR ZERO TRANSMIT FLAG
1448 002712 005037 002224 100$: CLR VSTAT ;CLEAR ALL INTERRUPT FLAGS
1449 002716 004037 015674 100$: JSR R0,ZFLAG ;ISSUE ESC Z TO VT61
1450 002722 012637 002124 100$: MOV (SP)+,IDENT ;POP STACK INTO IDENT
1451 002726 100002 100$: BPL 11$ ;IF IDENT IS -1,CLEAR IT.
1452 002730 005037 002124 11$: CLR IDENT
1453 002734 11$:
1454 002734 012637 002162 11$: MOV (SP)+,CHRD ;POP STACK INTO CHRD
1455 002740 001375 11$: BNE 11$
1456 002742 104401 001171 11$: TYPE ,SCRLF
1457 002746 104401 025314 11$: TYPE ,DVUNIT ;ISSUE UNIT UNDER TEST MESSAGE
1458 002752 013746 001732 11$: MOV VRCSR,-(SP) ;SAVE VRCSR FOR TYPEOUT
1459 11$: ;TYPE THE ADDRESS
1460 002756 104403 11$: TYPOS ;GO TYPE--OCTAL ASCII
1461 002760 006 11$: .BYTE 6 ;TYPE 6 DIGIT(S)
1462 002761 001 11$: .BYTE 1 ;TYPE LEADING ZEROS
1463 002762 017746 176724 11$: MOV @VECT,-(SP) ;SAVE @VECT FOR TYPEOUT
1464 11$: ;TYPE THE VECTOR
1465 002766 104403 11$: TYPOS ;GO TYPE--OCTAL ASCII
1466 002770 006 11$: .BYTE 6 ;TYPE 6 DIGIT(S)

```

```

1467 002771 000 .BYTE 0 ;;SUPPRESS LEADING ZEROS
1468 002772 013746 002124 MOV IDENT,-(SP) ;;SAVE IDENT FOR TYPEOLT
1469 ;;TYPE THE IDENT
1470 002776 104403 TYPOS ;;GO TYPE--OCTAL ASCII
1471 003000 006 .BYTE 6 ;;TYPE 6 DIGITS
1472 003001 000 .BYTE 0 ;;SUPPRESS LEADING ZEROS
1473 003002 104401 001171 TYPE ,SCRLF ;;CARRIAGE RETURN AND LINE FEED
1474 003006 032737 000001 002124 BIT #BIT00,IDENT ;;UNIT HAVE A COPIER?
1475 003014 001402 BEQ 20$ ;;NO
1476 003016 104401 025433 TYPE ,DCOPYR ;;YES-ISSUE COPIER MESSAGE
1477 003022 032737 000002 002124 20$: BIT #BIT01,IDENT ;;UNIT HAVE A PRINTER?
1478 003030 001402 BEQ 21$ ;;NO
1479 003032 104401 025405 TYPE ,DPRTR ;;YES-ISSUE PRINTER MESSAGE.
1480 003036 062737 000002 001712 21$: ADD #2,VECPT ;;LEAVE WITH VECPOINT AT NEXT VECTOR.
1481 003044 005037 002200 CLR FTLCNT ;;CLEAR COUNT OF FATAL XMIT.
1482 003050 012737 031604 031602 MOV #ABBUF,ABUFP ;;RESET THE REC. DATA POINTER
1483 003056 052777 000100 176646 BIS #RDENA,@VRCSR ;;SET THE REC. INT. ENABLE FOR TESTS
1484 003064 105737 002176 TSTB MODE ;;CHECK TESTING MODE
1485 003070 001403 BEQ ASTRT ;;AUTO MODE
1486 003072 012737 001652 002174 MOV #INTAB,TSTPTR ;;LOAD THE INITIAL TEST NUMBER
1487
1488 ;;*****
1489 ;;*****
1490 ;THIS TEST ISSUES ALL ESCAPE SEQUENCES AND
1491 ;INSURES THE VT61 HAS NOT FAILED DURING AN
1492 ;ESC SEQUENCE BY ISSUING A ESC Z TO FORCE A
1493 ;VT61 RESPONSE. THE PURPOSE OF THE TEST IS TO ATTEMPT TO
1494 ;INSURE THAT SUBSEQUENT TESTS WILL NOT RESULT IN
1495 ;A "HUNG" UNIT. DATA IS NOT EVALUATED.
1496 ;;*****
1497
1498 003100 ASTRT:
1499 ;;*****
1500 003100 000004 TST1: SCOPE
1501 003102 012737 000001 001160 MOV #1,$TIMES ;;DO 1 ITERATION
1502 003110 012737 003116 001106 MOV #ESTST,$LPADR ;;SET SCOPE LOOP ADDRESS
1503
1504 003116 012701 001750 ESTST: MOV #BEL,R1 ;;POINT TO FIRST COMMAND
1505 003122 042777 000100 176602 BIC #RDENA,@VRCSR ;;CLEAR REC. INT. ENABLE
1506 003130 113737 001102 002230 MOVB $TSTNM,TSTNM ;;LOAD THE TEST NUMBER.
1507 003136 005037 002214 CLR PRESC
1508 003142 005004 CLR R4
1509 003144
1510 003144 013746 002170 ZERST: MOV ZERO,-(SP) ;;PUSH ZERO ON STACK
1511 003150 012702 002214 MOV #PRESC,R2 ;;SET UP SEQUENCE ADDRESS
1512 003154 012103 GCMD: MOV (R1)+,R3 ;;LOAD THE COMMAND
1513 003156 001405 BEQ 1$ ;;IF CHAR. ZERO,MUST BE XMIT TERMINATOR
1514 003160 100535 BMI ESTEX ;;TABLE EXPENDED - EXIT TEST.
1515 003162 120327 000004 CMPB R3,#4 ;;IS COMMAND ACTUALLY A DELIMITER?
1516 003166 103442 BLO DELIM ;;YES, GO UPDATE FUNCTIONS
1517 003170 001471 BEQ SPTN ;;NO, ITS A "10" - SPECIAL CASE.
1518 003172 005704 1$: TST R4 ;;SEE IF FLAG INDICATING SEQ.
1519 003174 100472 BMI SEQ4 ;;4 IS SET. - YES EXIT
1520 003176 010337 002216 2$: MOV R3,ESSEQ ;;PUSH THE SEQUENCE TO BE TESTED
1521 003202 INXMT:
1522 003202 013746 002216 MOV ESSEQ,-(SP) ;;PUSH ESSEQ ON STACK

```

1523	003206	005704			TST	R4	;DOES THIS SEQUENCE REQUIRE
1524	003210	001402			BEQ	3\$	;ADDITIONAL ESC?
1525	003212	013746	002214		MOV	PRESC,-(SP)	;PUSH PRESC ON STACK
1526							
1527	003216	004037	013666		3\$: JSR	R0,TESC	;GO TRANSMIT THIS SEQUENCE.
1528							
1529	003222	005704			4\$: TST	R4	;IN I/O SEQUENCES?
1530	003224	100007			BPL	40\$	;NO
1531	003226	012737	000054	017502	MOV	#44.,DCOUNT	;YES,SET UP TO DELAY 1+ SEC.
1532	003234	004037	017440		JSR	R0,DELAY	
1533	003240	005777	176470		TST	@VRBUF	;CLEAR ANY RECEIVE FLAGS
1534							
1535	003244	004037	015674		40\$: JSR	R0,ZFLAG	;ISSUE ESC Z SEQUENCE-GET IDENT
1536	003250				5\$:		
1537	003250	012637	002162		MOV	(SP)+,CHRD	;POP STACK INTO CHRD
1538	003254	123737	002162	002124	CMPB	CHRD,IDENT	;HAVE WE POPPED THE IDENT?
1539	003262	001045			BNE	T1ERR	;NO-ERROR CONDITION
1540	003264				POPIT:		
1541	003264	012637	002162		MOV	(SP)+,CHRD	;POP STACK INTO CHRD
1542	003270	001375			BNE	-4	
1543	003272	000724			BR	ZERST	;GET NEXT COMMAND
1544							
1545	003274	120327	000001		DELIM: CMPB	R3,#1	
1546	003300	001407			BEQ	1\$	;FIRST DELIMITER - SET ESCN
1547							
1548	003302	120327	000002		CMPB	R3,#BIT01	;SECOND DELIMITER - SET ESCO
1549	003306	001412			BEQ	2\$	
1550							
1551	003310	120327	000003		CMPB	R3,#3	;THIRD DELIMITER - SET ESCP
1552	003314	001413			BEQ	3\$	
1553	003316	000716			BR	GCMD	;INVALID CHARACTER - GET ANOTHER
1554							
1555	003320	012704	000001		1\$: MOV	#1,R4	;SET FIRST DELIMITER FLAG.
1556	003324	013737	002132	002214	MOV	ESCN,PRESC	;INSERT ESCN.
1557	003332	000710			BR	GCMD	
1558							
1559	003334	013737	002054	002214	2\$: MOV	ESCO,@#PRESC	;INSERT ESCO
1560	003342	000704			BR	GCMD	
1561							
1562	003344	013737	002120	002214	3\$: MOV	ESCP,@#PRESC	;INSERT ESCP
1563	003352	000700			BR	GCMD	
1564							
1565	003354	012704	177777		SPTN: MOV	#-1,R4	;SET FLAG INDICATING I/O
1566	003360	000675			BR	GCMD	;SEQUENCES.
1567							
1568	003362	005703			SEQ4: TST	R3	;CHECK IF COMMAND = 0
1569	003364	001706			BEQ	INXMT	;YES, COMPLETE SEQUENCE ASSEMBLED
1570	003366	110322			MOVB	R3,(R2)+	;NO - KEEP ASSEMBLING
1571	003370	000303			SWAB	R3	;POSITION HIGH ORDER BIT
1572	003372	110322			MCVB	R3,(R2)+	;AND ASSEMBLE IT
1573	003374	000667			BR	GCMD	;GET ANOTHER BYTE
1574							
1575	003376	004037	016076		T1ERR: JSR	R0,CLREG	
1576	003402	013737	002214	001124	MOV	PRESC,\$GDDAT	;AND INSERT IN ERROR
1577	003410	000337	001124		SWAB	\$GDDAT	;REASSEMBLE FAILING SEQUENCES
1578	003414	013737	002216	001126	MOV	ESSEQ,\$BDDAT	

```

1579 003422 105737 002217          TSTB  ESSEQ+1      ;IF UPPER BYTE IS CLEAR DO NOT SWAP
1580 003426 001402          BEQ    1$
1581 003430 000337 001126          SWAB  $BDDAT      ;MESSAGE 1
1582 003434 104001          1$:  ERROR  1          ;ISSUE ERROR MESSAGE
1583 003436 005237 002200          INC    FTLCNT      ;INCREMENT FATAL XMIT COUNT.
1584 003442 023737 002200 002202  CMP    FTLCNT,ALWCNT ;FATAL XMITS EXCEEDED ALLOWED?
1585 003450 103003          BHIS  FTXT1      ;YES-EXIT.
1586 003452 000704          BR     POPIT      ;CLEAR THE STACK AND TRY ANOTHER COMMAND
1587 003454          ESTEX:
1588 003454 012637 002162          MOV    (SP)+,CHRD    ;;POP STACK INTO CHRD
1589 003460 052777 000100 176244 FTXT1: BIS    #RDENA,@VRCR    ;SET THE REC. INT. ENABLE FOR TESTS
1590
1591          ;;*****
1592          ;ROUTINE TO INSURE ENTERING MAINTENANCE MODE CAUSES SOM AND
1593          ;EOM TO BE APPENDED TO ALL TRANSMITS FROM VT61 UNDER TEST.
1594          ;MAINTENANCE MODE IS ENTERED , THEN AN ESCAPE 2 SEQUENCE
1595          ;IS ISSUED TO THE UNIT AND THE RESULTING TRANSMISSION IS
1596          ;CHECKED OF SOM/EOM.
1597          ;;*****
1598
1599          ;;*****
1600 003466 000004          TST2:  SCOPE
1601 003470 012737 000005 001160          MOV    #5,$TIMES      ;;DO 5 ITERATIONS
1602 003476 012737 003504 001106          MOV    #CKMNT,$LPADR    ;;SET SCOPE LOOP ADDRESS
1603
1604 003504 004037 015512          CKMNT: JSR    R0,RESETV      ;RESET THE UNIT AND SETMAINT. MODE.
1605 003510 112777 000002 011770          MOVB   #SOM,@TBUF      ;ISSUE START OF MESSAGE.
1606 003516 004037 016374          JSR    R0,XMIT1
1607 003522 113777 002126 011756          MOVB   ESCZ,@TBUF
1608 003530 004037 016374          JSR    R0,XMIT1      ;SEND AN IDENT REQUEST.
1609 003534 113777 002127 011744          MOVB   ESCZ+1,@TBUF
1610 003542 004037 016374          JSR    R0,XMIT1
1611 003546 112777 000004 011732          MOVB   #EOM,@TBUF      ;ISSUE END OF MESSAGE.
1612 003554 004037 016374          JSR    R0,XMIT1
1613 003560 005037 002220          CLR    DLAY
1614 003564 032737 040000 002224 1$:  BIT    #RSOM,VSTAT    ;SET UP SOM DELAY OF 100M.S.
1615 003572 001003          BNE   CKEOM      ;RECEIVED THE START OF MESSAGE?
1616 003574 005337 002220          DEC    DLAY      ;YES-GO LOOK FOR EOM.
1617 003600 001371          BNE   1$      ;NO-RUN TIMEOUT DELAY
1618          ;AND KEEP LOOKING.
1619 003602 012701 000062          CKEOM: MOV    #50.,R1      ;SET MAX DELAY FOR 500 M.S.
1620 003606 032737 020000 002224 1$:  BIT    #REOM,VSTAT    ;RECEIVED END OF MESSAGE?
1621 003614 001007          BNE   10$      ;YES-CHECK FOR BOTH RECEIVED.
1622 003616 012737 000001 017502          MOV    #1,DCOUNT      ;DELAY FOR 10 M.S.
1623 003624 004037 017440          JSR    R0,DELAY
1624 003630 005301          DEC    R1
1625 003632 001365          BNE   1$      ;AND KEEP LOOKING.
1626 003634 032737 040000 002224 10$: BIT    #RSOM,VSTAT    ;RECEIVED SOM?
1627 003642 001404          BEQ    2$      ;NO ISSUE ERROR
1628 003644 032737 020000 002224          BIT    #REOM,VSTAT    ;RECEIVED EOM?
1629 003652 001007          BNE   EXMNT      ;YES, NO ERRORS-EXIT.
1630 003654 012737 006001 001124 2$:  MOV    #6001,$GDDAT    ;LOAD ERROR WITH EXPECTED
1631 003662 013737 002224 001126          MOV    VSTAT,$BDDAT    ;AND ACTUAL STATUS.
1632 003670 104022          ERROR  22
1633
1634 003672 000240          EXMNT:  NOP

```

```

1635      ;*****
1636      ;THIS TEST INSURES THAT THE CURSOR WILL RESPOND
1637      ;TO DIRECT CURSOR ADDRESSING. THE UNIT IS RESET AND THE CURSOR
1638      ;POSITION IS VERIFIED TO BE HOME. THE CURSOR IS THEN MOVED
1639      ;TO POSITION ROW 23 COLUMN 80 AND THE POSITION IS AGAIN
1640      ;VERIFIED. ERRORS ARE REPORTED IF THE POSITIONS ARE INCORRECT.
1641      ;*****
1642      ;*****
1643      ;*****
1644      ;*****
1645      ;*****
1646      003674 000004      TST3:  SCOPE
1647      003676 012737 000005 001160      MOV      #5,$TIMES      ;;DO 5 ITERATIONS
1648      003704 012737 003712 001106      MOV      #CURS1,$LPADR      ;;SET SCOPE LOOP ADDRESS
1649
1650      003712 013701 015502      CURS1:  MOV      TBBUF,R1      ;USE R1 AS XMIT BUFFER POINTER.
1651      003716 004037 015512      JSR      R0,RESETV      ;RESET THE UNIT AND WAIT FOR XON.
1652      003722 013721 002054      MOV      ESCOI,(R1)+      ;CLFT. RESET, READ CURSOR
1653      003726 113721 002050      MOV      RDCJR,(R1)+      ;POSITION, CURSOR LEFT.
1654      003732 012737 000003 015510      MOV      #3,XMCNT      ;XMIT 3 BYTES
1655
1656      003740 004037 016120      JSR      R0,XMREC      ;XMIT AND RECEIVE.
1657      003744 000402      BR      10$      ;NORMAL EXIT.
1658      003746 104011      ERROR  11      ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
1659      003750 000446      BR      2$      ;EXIT TEST.
1660      003752 013701 030402      10$:  MOV      RCRLB,R1      ;GET THE CURRENT CURSOR POSITION.
1661      003756 020137 002154      CMP      R1,CUHME      ;CURSOR REALLY HOME?
1662      003762 001405      BEQ      1$      ;YES EXIT
1663      003764 104012      ERROR  12      ;VT61 FAILURE MESSAGE
1664      003766 013746 002154      MOV      CUHME,-(SP)      ;PUSH CUHME ON STACK
1665      003772 004037 016600      JSR      R0,CURER      ;GO LOAD AND ISSUE CURSOR ERROR
1666
1667      003776 013701 015502      1$:  MOV      TBBUF,R1      ;LOAD XMIT BUFFER WITH
1668      004002 013721 002040      MOV      DCRAD,(R1)+      ;CURSOR TO ROW 23,COL.79
1669      004006 013721 002042      MOV      R23C79,(R1)+      ;READ CURSOR POSITION
1670      004012 013721 002054      MOV      ESCOI,(R1)+      ;IT AND CURSOR RIGHT
1671      004016 013721 002050      MOV      RDCUR,(R1)+      ;XMIT 7 BYTES.
1672      004022 012737 000007 015510      MOV      #7,XMCNT      ;XMIT AND RECEIVE
1673      004030 004037 016120      JSR      R0,XMREC      ;NORMAL EXIT.
1674      004034 000402      BR      20$      ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
1675      004036 104011      ERROR  11      ;EXIT TEST.
1676      004040 000412      BR      2$
1677      004042 012701 030402      20$:  MOV      #RCRLB,R1
1678
1679      004046 023711 002042      CMP      R23C79,(R1)      ;CHECK CURSOR POSITION TO LOWER RT.
1680      004052 001405      BEQ      2$      ;OK, EXIT
1681      004054 104012      ERROR  12      ;VT61 FAILURE MESSAGE
1682      004056 013746 002042      MOV      R23C79,-(SP)      ;PUSH R23C79 ON STACK
1683      004062 004037 016600      JSR      R0,CURER      ;LOAD AND ISSUE CURSOR ERROR.
1684      004066 000240      2$:  NOP
1685      ;*****
1686      ;ROUTINE TO INSURE THE UNIT CAN ENTER LINEAR ADDRESSING
1687      ;MODE. 81 CHARACTERS ARE ISSUED TO THE UNIT UNDER TEST
1688      ;THEN THE CURSOR POSITION IS READ AND MUST BE ROW1,COL.0.
1689      ;*****
1690

```

```

1691
1692 004070 000004
1693 004072 012737 000005 001160
1694 004100 012737 004106 001106
1695
1696 004106 004037 015512
1697 004112 013701 015502
1698 004116 012703 000120
1699 004122 004037 017504
1700 004126 013721 002046
1701 004132 013721 002050
1702 004136 012737 000123 015510
1703 004144 004037 016120
1704 004150 000402
1705 004152 104011
1706 004154 000421
1707 004156 023777 002134 011010
1708 004164 001415
1709 004166 013737 002054 001124
1710 004174 000337 001124
1711 004200 013737 002010 001126
1712 004206 104001
1713 004210 013746 002134
1714 004214 004037 016600
1715 004220 000240
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728 004222 000004
1729 004224 012737 000010 001160
1730 004232 012737 004240 001106
1731 004240 013701 015502
1732 004244 012737 001001 002226
1733 004252 005037 002224
1734 004256 004037 015512
1735 004262 013721 002040
1736 004266 013721 002144
1737 004272 013721 002054
1738 004276 013721 002056
1739 004302 012703 000050
1740 004306 004037 017504
1741 004312 012737 000057 015510
1742 004320 052777 000100 175410
1743 004326 012703 000050
1744 004332 012737 000001 017502
1745 004340 004037 017440
1746 004344 032737 100000 002224

;*****
1ST4: SCOPE
      MOV #5,$TIMES ;;DO 5 ITERATIONS
      MOV #CKLIN,$LPADR ;;SET SCOPE LOOP ADDRESS
CKLIN: JSR RO,RESETV ;RESET THE UNIT-SET MAINT AND LINEAR MODES
      MOV TBBUF,R1
      MOV #80.,R3
      JSR RO,BLDINC ;LOAD XMIT BUFFER WITH 80 CHAR AND
      MOV RCUR,(R1)+ ;READ CURSOR POSINION.
      MOV RDCUR,(R1)+
      MOV #83.,XMCNT
      JSR RO,XMREC ;XMIT THE BUFFER.
      BR 1$
      ERROR 11 ;LAST XMIT CAUSED UNIT TO HANG.
      BR LINXT ;EXIT TEST
1$: CMP RO1C00,@RBBUF ;CURSOR AT ROW1,COL. 0?
      BEQ LINXT ;YES-EXIT
      MOV ESCO,$GDDAT
      SWAB $GDDAT
      MOV DIRECT,$BDDAT ;ISSUE ESC SEQUENCE AND CURSOR
      ERROR 1
      MCV RO1C00,-(SP) ;;PUSH RO1C00 ON STACK
      JSR RO,CURER
LINXT: NOP

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

;*****
1ST5: SCOPE
      MOV #10,$TIMES ;;DO 10 ITERATIONS
      MOV #BASC3,$LPADR ;;SET SCOPE LOOP ADDRESS
BASC3: MOV TBBUF,R1 ;R1 = 1ST XMIT BUFFER ADDRESS.
      MOV #1001,BLKM ;SET UP TO XMIT A SOM -DATA- EOM.
      CLR VSTAT
      JSR RO,RESETV ;RESET THE UNIT AND WAIT FOR XON.
      MOV DCRAD,(R1)+
      MOV R23C00,(R1)+ ;CURSOR TO ROW 23, COL.0
      MOV ESCO,(R1)+
      MOV XMTAL,(R1)+ ;TRANSMIT THE LINE.
      MOV #40.,R3
      JSR RO,BLDINC ;40 CHAR. OF INCREMENTING CHAR.
      MOV #47.,XMCNT ;SET UP TO XMIT 47 BYTES
      BIS #TENA,@VXCSR ;TRANSMIT ENABLES
      MOV #40.,R3 ;MAXIMUM DELAY EQUAL 800 M.S. OR MORE DEPENDING ON PROCE
      MOV #1,DCOUNT
      JSR RO,DELAY ;DELAY FOR 10 MS DEPENDING ON PROCESSOR SPEED.
      BIT #RXOFF,VSTAT ;CHECK FOR XOFF
  
```

```

1747 004352 001007      BNE      3$      ;FOUND IT EXIT THIS SECTION.
1748 004354 005303      DEC      R3      ;DELAYED 800 M.S.?
1749 004356 001365      BNE      2$      ;NO-KEEP LOOKING FOR XOFF.
1750 004360 104012      ERROR    12      ;GENERAL VT61 FAILURE MESSAGE
1751 004362 012746 100000  MOV      #100000,-(SP) ;PUSH #100000 ON STACK
1752 004366 004037 015734  JSR      R0,CKSFT ;GO REPORT ERROR
1753 004372      3$:      MOV      #XMDNE,-(SP) ;PUSH #XMDNE ON STACK
1754 004372 012746 000001  TST      PTYPE ;PROCESSOR TYPE ;:++C
1755 004376 005737 002232  BEQ      8$      ;BR IF 11/45-70. ;:++C
1756 004402 001407      BMI      7$      ;BR IF 11/35,40. ;:++C
1757 004404 100403      MOV      #25,-(SP) ;PUSH #25. ON STACK
1758 004406 012746 000031  BR       10$     ;:++C
1759 004412 000405      7$:      MOV      #50,-(SP) ;PUSH #50. ON STACK
1760 004414      BR       10$     ;:++C
1761 004414 012746 000062  8$:      MOV      #100,-(SP) ;PUSH #100. ON STACK
1762 004420 000402      JSR      R0,WIBGND ;WAIT XMDNE ;:++C
1763 004422      BR       EXIT3    ;TIMEOUT-EXIT TEST.
1764 004422 012746 000144  10$:     JSR      R0,WIBGND ;RECEIVED A XON?
1765 004426 004037 021076  CMPB     @ABUFP,#XON ;YES-NO ERROR-EXIT
1766 004432 000411      BEQ      EXIT3
1767 004434 127727 025142 000021  ERROR    12      ;GENERAL VT61 FAILURE MESSAGE
1768 004442 001405      MOV      #000001,-(SP) ;PUSH #000001 ON STACK
1769      JSR      R0,CKSFT
1770 004444 104012      JSR      R0,RESPTR ;RESET INTERRUPT POINTERS.
1771 004446 012746 000001  EXIT3:   JSR      R0,CKSFT
1772 004452 004037 015734
1773 004456 004037 016520
1774
1775      ;*****
1776      ;ROUTINE TO VERIFY OPERATION OF XOFF AND XON TO THE VT61.
1777      ;A FULL SCREEN TRANSMIT IS INITIATED AND A SERIES OF XOFF AND
1778      ;XON ARE ISSUED TO THE TERMINAL SEQUENTIALLY.
1779      ;ERRORS ARE REPORTED IF XOFF DOES NOT STOP,OR XON RESTART
1780      ;THE TRANSMISSION. TEST IS ENDED WHEN EOM IS SENSED.
1781      ;*****
1782
1783      ;*****
1784      ;*****
1785      ;*****
1786      ;*****
1787      ;*****
1788      ;*****
1789      ;*****
1790      ;*****
1791      ;*****
1792      ;*****
1793      ;*****
1794      ;*****
1795      ;*****
1796      ;*****
1797      ;*****
1798      ;*****
1799      ;*****
1800      ;*****
1801      ;*****
1802      ;*****

```

```

1803 004604 001406          BEQ      ONOFA          ;XMIT STOPPED-GO RESTART IT.
1804 004606 005304          DEC      R4
1805 004610 001364          BNE      OFFLP          ;COUNTER NO EQUAL 300 MS-LOOP
1806 004612 013737 002224 001120 MOV      VSTAT,$GDADR ;UNIT DID NOT RESPOND TO XOFF
1807 004620 104015          ERROR    15          ;ISSUE ERROR
1808
1809 004622 112777 000021 010656 ONOFA: MOV      #XON,@IBUFP
1810 004630 004037 016374          JSR      R0,XMIT1 ;SEND A XON TO THE VT61.
1811 004634 012704 000036          MOV      #30.,R4 ;SET UP FOR 300MS DELAY.
1812 004640 032737 020000 002224 ONLP: BIT      #REOM,VSTAT ;EOM RECEIVED?
1813 004646 001020          BNE      ONOFT          ;YES-EXIT
1814 004650 013705 031602          MOV      ABUFP,R5
1815 004654 012737 000001 017502 MOV      #1,DCOUNT
1816 004662 004037 017440          JSR      R0,DELAY ;ALLOW 300 MS FOR XMIT TO RESTART
1817 004666 023705 031602          CMP      ABUFP,R5
1818 004672 001317          BNE      ONOFLP          ;IT RESTARTED-GO STOP IT.
1819 004674 005304          DEC      R4
1820 004676 001360          BNE      ONLP          ;NOT YET 300 MS LOOP.
1821 004700 013737 002224 001120 MOV      VSTAT,$GDADR ;XMIT DID NOT RESTART-ISSUE
1822 004706 104016          ERROR    16          ;ERROR AND EXIT
1823 004710 000240          ONOFT: NOP
1824
1825          ;*****
1826          ;ROUTINE TO TEST VT61 PAM AND THE COMMUNICATION PATHS.
1827          ;THIS ROUTINE ISSUES A SERIES OF PATTERNS(77/100,100/77,
1828          ;52/125,INCREMENTING,AND REV. VIDEO INCREMENTING) TO THE VT61.
1829          ;THE SCREEN IS THEN TRANSMITTED TO THE HOST AND AFTER EACH
1830          ;ITERATION RECEIVED DATA IS CHECKED AND ALL ERRORS(INCLUDING
1831          ;TRANSMISSION) ARE REPORTED.
1832          ;MITTED TO THE HOST COMPUTER AND THE RESULTS ARE CHECKED AND
1833          ;ALL ERRORS(INCLUDING TRANSMISSION) REPORTED.
1834          ;*****
1835
1836          ;*****
1837 004712 000004          TST7:  SCOPE
1838 004714 012737 000001 001160 MOV      #1,$TIMES ;DO 1 ITERATION
1839 004722 012737 004730 001106 MOV      #MEM1,$LPADR ;SET SCOPE LOOP ADDRESS
1840
1841 004730 004037 015512          MEM1: JSR      R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
1842 004734 005005          CLR      R5 ;CLEAR PATTERN OFFSET.
1843 004736 016504 005444          MEMA: MOV      MPATT(R5),R4 ;LOAD PATTERN TO BE TRANSMITTED
1844 004742 004037 016520          JSR      R0,RESPTR ;RESET POINTERS
1845 004746 042737 077577 002224 BIC      #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1846 004754 012702 003600          MOV      #TOTCH,R2 ;LOAD A COUNT OF SCREEN
1847 004760 112777 000002 010520 MOV      #SOM,@IBUFP ;ISSUE START OF MESSAGE.
1848 004766 004037 016374          JSR      R0,XMIT1
1849 004772 005302          MEMB: DEC      R2 ;DECREMENT XMIT COUNT
1850 004774 001414          BEQ      10$ ;COUNT ZERO?

```

1851								
1852	004776	004037	005412		12\$:	JSR	RO,PATGN	;NO-GENERATE NEXT BYTE TO XMIT.
1853	005002	110477	010500			MOVB	R4,@TBUFP	;LOAD THE CHARACTER.
1854	005006	004037	016374			JSR	RO,XMIT1	;NO-XMIT ANOTHER BYTE.
1855	005012	023737	002200	002202		CMP	FTLCNT,ALWCNT	;EXCEEDED FATAL ERROR COUNT?
1856	005020	103764				BLO	MEMB	;NO-CHECK IF ANOTHER TRANSMISSION REQUIRED.
1857	005022	000137	005464			JMP	MEMXT	;YES-GO ABORT TEST.
1858	005026	112777	000004	010452	10\$:	MOVB	#EOM,@TBUFP	;ISSUE END OF MESSAGE.
1859	005034	004037	016374			JSR	RO,XMIT1	
1860	005040	004037	016520			JSR	RO,RESPTR	;RESET INTERRUPT POINTERS.
1861								
1862	005044	013701	015502			MOV	TBBUF,R1	;LOAD XMIT BUFFER WITH
1863	005050	013721	002132			MOV	ESCN,(R1)+	
1864	005054	013721	001762			MOV	CHOM,(R1)+	;CURSOR HOME
1865	005060	013721	002126			MOV	ESCZ1,(R1)+	;ESCAPE Z
1866	005064	013721	002054			MOV	ESCO,(R1)+	
1867	005070	013721	002056			MOV	XMTAL,(R1)+	;TRANSMIT ALL
1868	005074	013711	001754			MOV	LNFEED,(R1)	;LINE FEED.
1869	005100	012737	000010	015510		MOV	#8,XMCNT	;SET UP TO XMIT 8 BYTES
1870	005106	004037	016120			JSR	RO,XMREC	;XMIT,WAIT FOR REC. EOM
1871	005112	000402				BR	1\$	;NORMAL EXIT
1872	005114	104011				ERROR	11	;LAST TRANSMIT CAUSED VT61 TO HANG
1873	005116	000562				BR	MEMXT	;EXIT TEST
1874	005120	042737	077577	002224	1\$:	BIC	#77577,VSTAT	;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1875	005126	005002				CLR	R2	;CLEAR RECEIVE COUNTER.
1876	005130	016504	005444			MOV	MPATT(R5),R4	;LOAD PATTERN
1877	005134	012703	031402			MOV	#TCRLB+300,R3	;SET UT ERROR STORAGE
1878	005140	013701	015174			MOV	RBBUF,R1	;SET UP RECEIVE POINTER
1879	005144	005037	002220		MEMC:	CLR	DLAY	;SET UP TIME OUT DELAY
1880	005150	013737	015174	015200		MOV	RBBUF,RBUFP	;RESET RECEIVE POINTER
1881	005156	023701	015200		1\$:	CMP	RBUFP,R1	;RECEIVED A CHAR?
1882	005162	001013				BNE	MEMD	;YES-GO CHECK IT.
1883	005164	032737	020000	002224		BIT	#REOM,VSTAT	;HAVE WE RECEIVED EOM?
1884	005172	001033				BNE	CKDAT	;YES, GO CHECK FOR DATA ERRORS
1885	005174	005337	002220			DEC	DLAY	;RUN TIME OUT DELAY.
1886	005200	001366				BNE	1\$	;NOT EXPIRED-KEEP LOOKING.
1887	005202	005237	002200			INC	FTLCNT	;TRANSMISSION FAILED-INCR. FATAL COUNT
1888	005206	104011				ERROR	11	
1889	005210	000525				BR	MEMXT	
1890	005212	005202			MEMD:	INC	R2	;DATA IN. INCREMENT COUNTER
1891	005214	004037	005412			JSR	RO,PATGN	;GET GOOD CHARACTER,PUT IN R4 AND
1892	005220	122705	000010			CMPB	#10,R5	;CHECKING REV. VIDEO DATA?
1893	005224	001002				BNE	1\$	;NO-DO NOT MODIFY
1894	005226	052704	000200			BIS	#BIT07,R4	;YES-FORCE BIT 7.
1895	005232	121104			1\$:	CMPB	(R1),R4	;COMPARE DATA
1896	005234	001743				BEQ	MEMC	
1897	005236	020227	003600			CMP	R2,#TOTCH	;COMPARING LAST CHAR?
1898	005242	001740				BEQ	MEMC	;YES-NEVER COUNT AS A ERROR.
1899								
1900	005244	020327	031452			CMP	R3,#TCRLB+350	;STORED 20 ERRORS?
1901	005250	103335				BHIS	MEMC	;YES-STORE NO MORE.
1902	005252	110423				MOVB	R4,(R3)+	;STORE THE GOOD DATA.
1903	005254	111123				MOVB	(R1),(R3)+	;STORE THE BAD DATA.
1904	005256	010223				MOV	R2,(R3)+	;STORE THE RECEIVE COUNT.
1905	005260	000731				BR	MEMC	
1906	005262	022703	031402		CKDAT:	CMP	#TCRLB+300,R3	

1907	005266	001415			BEQ	CKMEM		
1908	005270	012701	031402		MOV	#TCRLB+300,R1	:LOAD FIRST ERROR ADDRESS.	
1909	005274	004037	016076		JSR	RO,CLREG	:CLEAR ERROR REGISTERS	
1910	005300	112137	001124		MOVB	(R1)+,\$GDDAT	:LOAD THE GOOD DATA.	
1911	005304	112137	001126		MOVB	(R1)+,\$BDDAT	:LOAD THE ERROR BUFFER	
1912	005310	012137	001120		MOV	(R1)+,\$GDADR	:LOAD RECEIVE COUNT	
1913	005314	104004			ERROR	4	:ISSUE DATA ERROR MESSAGE.	
1914	005316	020103			CMP	R1,R3	:ISSUED ALL ERRORS?	
1915	005320	103765			BLO	1\$	:NO-CONTINUE	
1916								
1917	005322	020227	003600		CKMEM:	CMP R2,#TOTCH	:DID WE XFER 1920 TIMES?	
1918	005326	001406			BEQ	1\$	:YES - GO CHECK STATUS	
1919	005330	012737	003600	001124	MOV	#TOTCH,\$GDDAT	:NO, PUT GOOD COUNT IN GDDAT	
1920	005336	010237	001126		MOV	R2,\$BDDAT	:AND ACTUAL COUNT IN BDDAT.	
1921	005342	104005			ERROR	5	:ISSUE COUNT ERROR.	
1922								
1923	005344				1\$:			
1924	005344	012746	060000		MOV	#60000,-(SP,	:;PUSH #60000 ON STACK	
1925	005350	004037	015734		JSR	RO,CKSFT		
1926	005354	062705	000002		ADD	#2,R5	:INCREMENT PATTERN POINTER	
1927	005360	005765	005444		TST	MPATT(R5)	:TEST NEXT PATTERN	
1928	005364	001437			BEQ	MEMXT	:ZERO-END OF TEST EXIT.	
1929	005366	100007			BPL	2\$	:NOT INCRMENTING PATTERN.	
1930	005370	122705	000010		CMPB	#10,R5	:SET REVERSE VIDEO?	
1931	005374	001004			BNE	2\$	:NO.	
1932	005376	012703	005460		MOV	#SETREV,R3	:YES-ENTER REVERSE VIDEO	
1933	005402	004037	016460		JSR	RO,LDXMIT	:AND RE-ISSUE INCREMENTING PATTERN.	
1934	005406	000137	004736		2\$:	JMP MEMA	:NOT ZERO, GO EXERCISE IT.	
1935								
1936	005412	042704	000200		PATGN:	BIC #200,R4	:CLEAR REV. VIDEO BIT IF SET.	
1937	005416	005704			TST	R4	:CHECK R4 FOR PATTERN	
1938	005420	100402			BMI	1\$	:IF MINUS, DO INCREMENTING.	
1939	005422	000304			SWAB	R4	:OTHERWISE SWAP BYTES AND	
1940	005424	000200			RTS	RO	:EXIT.	
1941	005426	105204			1\$:	INCB R4	:ADD ONE TO INCREMENTING	
1942	005430	120427	000177		CMPB	R4,#177	:HAVE WE EXCEEDED LIMIT	
1943	005434	103402			BLO	2\$	:NO, EXIT	
1944	005436	016504	005444		MOV	MPATT(R5),R4	:YES, RESET PATTERN AND	
1945	005442	000200			2\$:	RTS RO	:EXIT.	
1946								
1947		005444			MPATT	.EVEN		
1948	005444	037500				.WORD	037500	:PATTERN 77,100
1949	005446	040077				.WORD	040077	:PATTERN 100,77
1950	005450	025125				.WORD	025125	:PATTERN 52,125
1951	005452	100040				.WORD	100040	:PATTERN INCREMENTING
1952	005454	100040				.WORD	100040	:PATTERN INCREMENTING-REV. VIDEO.
1953	005456	000000				.WORD	0	:PATTERN TABLE TERMINATOR
1954								:SEQUENCE TO ENTER REVERSE VIDEO.
1955	005460	033	112	112	SETREV:	.BYTE .ESC,.O,.EEMP,0		
1956	005463	000						
1957	005464	000240			MEMXT:	NOP		
1958								
1959								
1960								
1961								
1962								

;;*****
 :ROUTINE TO TEST THE ABILITY OF THE VT61 TO CALCULATE
 :AND TRANSMIT CHECKSUMS OF BOTH TRANSMITTED AND RECEIVED

```

1963 ;DATA. SUBTEST A TRANSMITS A FULL BUFFER UPDATING A CALCULATED
1964 ;CHECKSUM ON EACH CHARACTER TRANSMITTED. AN ESCAPE SEQUENCE
1965 ;REQUESTING THE RECEIVER CHECKSUM IS EMBEDDED AT THE END OF
1966 ;XMIT BUFFER AND THE RECEIVED CHECKSUM IS COMPARED TO THE
1967 ;CALCULATED. SUBTEST B PERFORMS THE SAME TYPE OF CHECK ON
1968 ;THE VT61 TRANSMIT CHECKSUM,UTILIZING THE DATA SENT TO THE VT61
1969 ;IN SUBTEST A,DURING A FULL SCREEN TRANSMIT.
1970
1971 ;:*****
1972 ;:*****
1973
1974 005466 000004 TST10: SCOPE
1975 005470 012737 000003 001160 MOV #3,$TIMES ;DO 3 ITERATIONS
1976 005476 012737 005504 001106 MOV #CKSUMA,$LPADR ;SET SCOPE LOOP ADDRESS
1977
1978 005504 004037 015512 CKSUMA: JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
1979 005510 012737 001001 002226 MOV #1001,BLKM ;SET UP TO XMIT A SOM -DATA- EOM.
1980 005516 004037 016520 JSR R0,RESPTR ;RESET INTERRUPT POINTERS
1981 005522 012703 006202 MOV #ITSUMA,R3 ;DIS. RECT. MODE AND CLEAR CHECKSUM
1982 005526 004037 016460 JSR R0,LDXMIT
1983 005532 042737 077577 002224 BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
1984 005540 013701 015502 MOV TBBUF,R1 ;LOAD XMIT BUFFER WITH
1985 005544 012703 000473 MOV #315.,R3
1986 005550 004037 017504 JSR R0,BLDINC ;314 INCREMENTING CHAR.
1987 005554 113721 002132 MOVB ESCN,(R1)+
1988 005560 113721 001762 MOVB CHOM,(R1)+ ;CURSOR HOME
1989 005564 113721 002054 MOVB ESCO,(R1)+
1990 005570 113721 002055 MOVB ESCO+1,(R1)+
1991 005574 113711 002072 MOVB TXRCK,(R1) ;TRANSMIT RECEIVER CHECKSUM.
1992 005600 005004 CLR R4 ;CLEAR CHECKSUM REGISTER
1993 005602 012705 000004 MOV #EOM,R5 ;PRELOAD CHECKSUM REG. WITH
1994 005606 004037 020124 JSR R0,CALCK ;EOM FROM PRIOR XMIT.
1995 005612 052737 002000 002224 BIS #CKSUM,VSTAT ;REQUEST CHECKSUM CALCULATIONS.
1996 005620 012737 000500 015510 MOV #320.,XMCNT ;SETUP TO XMIT 320 BYTES
1997 005626 052777 000100 174102 BIS #TENA,@VXCSR ;ENABLE XMIT INTERRUPTS
1998 005634 012746 020000 MOV #REOM,-(SP) ;PUSH #REOM ON STACK
1999 005640 005737 002232 TST PTYPE ;PROCESSOR TYPE
2000 005644 001407 BEQ 2$ ;IF 0 THEN 11/45-11/70
2001 005646 100403 BMI 1$ ;IF -1 THEN 11/35,40
2002 005650 012746 000005 MOV #5.,-(SP) ;PUSH #5. ON STACK
2003 005654 000405 BR 3$
2004 005656 1$: MOV #10.,-(SP) ;PUSH #10. ON STACK
2005 005656 012746 000012 BR 3$
2006 005662 000402 2$:
2007 005664 3$: MOV #20.,-(SP) ;PUSH #20. ON STACK
2008 005664 012746 000024 JSR R0,WIBGND ;LOOK FOR EOM.
2009 005670 004037 021076 BR CKEXT ;ERROR EXIT IF NOT FOUND
2010 005674 000546 CMPB @RBBUF,R4 ;COMPARE CHECKSUMS
2011 005676 127704 007272 BEQ CKSUMB ;GOOD GO TO SUBTEST B
2012 005702 001414 JSR R0,CLREG ;BAD COMPARE
2013 005704 004037 016076 MOVB R4,$GDDAT ;LOAD CALCULATED CHECKSUM
2014 005710 110437 001124 MOVB @RBBUF,$BDDAT ;AND VT61 RECEIVER CHECKSUM
2015 005714 117737 007254 001126 ERROR 13 ;ISSUE ERROR
2016 005722 104013 MOV #60001,-(SP) ;PUSH #60001 ON STACK
2017 005724 012746 060001 JSR R0,CKSFT ;ERROR.
2018 005730 004037 015734

```

```

2019
2020 005734 042737 077577 002224 CKSUMB: BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2021 005742 005004 CLR R4 ;CLEAR CHECKSUM REGISTER
2022 005744 012737 001001 002226 MOV #1001,BLKM ;SET UP TO XMIT A SOM -DATA- EOM.
2023 005752 052737 000100 002224 BIS #TXSUM,VSTAT ;SET UP FOR XMIT CHECKSUM GENERATION.
2024 005760 013701 015502 MOV TBBUF,R1 ;LOAD XMIT BUFFER WITH
2025 005764 004037 020172 JSR R0,LDBUF ;LOAD THE BUFFER WITH:
2026 005770 033 117 134 .BYTE .ESC,.O,.CLTCK,.ESC,.O,.XMTAL,.ESC,.O,.TXTCK,O
2027 005773 033 117 126
2028 005776 033 117 136
2029 006001 000
2030 006002 012737 000011 015510 MOV #9,XMCNT ;SET UP TO XMIT 9 BYTES
2031 006010 052777 000100 173720 BIS #TENA,@VXCSR ;ALLOW XMIT INTERRUPTS
2032 006016 012746 000001 MOV #XMDNE,-(SP) ;PUSH #XMDNE ON STACK
2033 006022 005737 002232 TST PTYPT ;PROCESSOR TYPE
2034 006026 001407 BEQ 2$ ;IF 0 THEN 11/45-11/70
2035 006030 100403 BMI 1$ ;IF -1 THEN 11/35,40
2036 006032 012746 000001 MOV #1,-(SP) ;PUSH #1 ON STACK
2037 006036 000405 BR 3$
2038 006040 1$:
2039 006040 012746 000002 MOV #2,-(SP) ;PUSH #2. ON STACK
2040 006044 000402 BR 3$
2041 006046 2$:
2042 006046 012746 000004 MOV #4,-(SP) ;PUSH #4. ON STACK
2043 006052 004037 021076 JSR R0,WTBGND ;LOOK FOR XMIT DONE.
2044 006056 000455 OR CKEXT ;TIME OUT - EXIT TEST.
2045 006060 005037 002220 CKSRC: CLR DLAY ;SET UP TIME OUT DELAY
2046 006064 013702 031602 MOV ABUF,R2 ;RESET THE RECEIVER FLAG
2047 006070 023702 031602 1$: CMP ABUF,R2 ;RECEIVED A CHAR?
2048 006074 001007 BNE 2$ ;YES-GO CHECK IT.
2049 006076 005337 002220 DEC DLAY ;RUN TIME OUT DELAY.
2050 006102 001372 BNE 1$
2051 006104 005237 002200 INC FILCNT ;TIMED OUT-INCREMENT FATAL XMIT COUNT
2052 006110 104011 ERROR 11 ;ISSUE HUNG MESSAGE AND EXIT.
2053 006112 000437 BR CKEXT
2054 006114 122777 000004 023460 2$: CMPB #EOM,@ABUF ;RECEIVED EOM CHAR?
2055 006122 001356 BNE CKSRC
2056 006124 042737 020000 002224 BIC #REOM,VSTAT ;CLEAR THE EOM FLAG
2057 006132 032737 020000 002224 BIT #REOM,VSTAT ;NOW WAIT FOR LAST EOM FLAG
2058 006140 001774 BEQ -6 ;FROM XMIT TRANSMITTER CHECKSUM.
2059 006142 120477 007026 CMPB R4,@RBBUF ;COMPARE 61 TO HOST CHECKSUM.
2060 006146 001421 BEQ CKEXT ;EQUAL - EXIT TEST
2061 006150 004037 016076 JSR R0,CLREG
2062 006154 110437 001124 MOVB R4,$GDDAT ;LOAD THE HOST CALCULATED CHECKSUM
2063 006160 117737 007010 001126 MOVB @RBBUF,$BDDAT ;LOAD THE VT61 TRANSMITTED CHECKSUM
2064 006166 104014 ERROR 14 ;ISSUE VT61 XMIT CHECKSUM ERROR
2065 006170 012746 060001 MOV #60001,-(SP) ;PUSH #60001 ON STACK
2066 006174 004037 015734 JSR R0,CKSFT ;CHECK FOR STATUS ERROR
2067 006200 000404 BR CKEXT
2068
2069 006202 033 117 103 ITSUMA: .BYTE .ESC,.O,.DRECT,.ESC,.O,.CLRCK,O,O
2070 006205 033 117 133
2071 006210 000 000
2072
2073 006212 004037 016520 CKEXT: JSR R0,RESPIR
2074

```

```

2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086 006216 000004
2087 006220 012737 000005 001160
2088 006226 012737 006234 001106
2089
2090 006234 013701 015502
2091 006240 004037 015512
2092 006244 004037 020172
2093 006250 033 103 033
2094 006253 117 131 033
2095 006256 102 033
2096 006260 117 131 033
2097 006263 104 033 117
2098 006266 131
2099 006267 033 101 033
2100 006272 117 131 007
2101 006275 000
2102 006276 012737 000024 015510
2103 006304 012737 000004 016366
2104 006312 012737 031202 016370
2105 006320 004037 016120
2106 006324 000402
2107 006326 104011
2108 006330 000436
2109 006332 012701 006416
2110 006336 012702 031202
2111 006342 012703 001764
2112
2113 006346 021112
2114 006350 001415
2115 006352 113737 002132 001125
2116 006360 111337 001124
2117 006364 005037 001126
2118 006370 104001
2119 006372 011237 030402
2120 006376 011146
2121 006400 004037 016600
2122 006404 022122
2123 006406 022337 001772
2124 006412 001355
2125 006414 000404
2126
2127
2128 006416 020440
2129 006420 020441
2130 006422 020041

;*****
;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
;CURSOR COMMAND AND CURRENT IS COMPARED TO GOOD
;AND ANY ERRORS REPORTED.
;*****

11$: SCOPE
MOV #5,$TIMES ;;DO 5 ITERATIONS
MOV #CURS1A,$LPADR ;;SET SCOPE LOOP ADDRESS

CURS1A: MOV TBBUF,R1 ;LOAD XMIT BUFFER ADDRESS
JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
JSR R0,LDBUF ;LOAD THE BUFFER WITH:
.BYTE .ESC,.CRT,.ESC,.O,.RDCUR,.ESC,.CDWN,.ESC

.BYTE .O,.RDCUR,.ESC,.CLFT,.ESC,.O,.RDCUR

.BYTE .ESC,.CUP,.ESC,.O,.RDCUR,.BEL,O

MOV #20,$XMCNT ;SET TO XMIT 20 CHARACTERS
MOV #4,$RECIIT ;SET RECEIVE ITERATION TO 4
MOV #TCRLB+100,$WDSTOR ;SET UP WORD STORAGE POINTER
JSR R0,$XMRCC ;XMIT ,AND WAIT FOR REC.DONE
BR 11$ ;NORMAL EXIT
ERROR 11 ;LAST XMIT CAUSED VT61 TO HANG.
BR CUR1XT ;EXIT TEST
11$: MOV #GDCURP,R1 ;R1=GOOD POSITION TABLE
MOV #TCRLB+100,R2 ;R2=ACTUAL CURSOR POSITION
MOV #CRT,R3 ;R3=CURSOR COMMAND TABLE

12$: CMP (R1),(R2) ;COMPARE GOOD TO ACTUAL
BEQ 2$ ;OK-GO UPDATE POINTERS.
MOVB ESCN,$GDDAT+1
MOVB (R3),$GDDAT ;LOAD COMMAND IN ESC ERROR
CLR $BDDAT
ERROR 1 ;AND ISSUE IT
MOV (R2),RCRLB ;LOAD BAD CURSOR POSITION
MOV (R1),-(SP) ;PUSH (R1) ON STACK
JSR R0,CURER ;LOAD AND ISSUE CURSOR ERROR MESSAGE
2$: CMP (R1)+,(R2)+ ;INCREMENT POSITION POINTERS.
CMP (R3)+,$CUP ;CHECK FOR COMMAND TERM.(CUP).
BNE 12$ ;NOT AT TERMINATOR-COMPARE AGAIN
BR CUR1XT ;EXIT TEST

GDCURP: .WORD 20440 ;ROW 0, COL. 1
.WORD 20441 ;ROW 1, COL. 1
.WORD 20041 ;ROW 1, COL. 0

```

```

2131 006424 020040          .WORD 20040          ;ROW 0, COL. 0
2132 006426 000240          CUR1XT: NOP
2133
2134          ;*****
2135          ;ROUTINE TO INSURE THAT READ CHARACTER AT CURSOR
2136          ;FUNCTIONS CORRECTLY. COMMAND SEQUENCE IS: RESET, A, CURSOR
2137          ;LEFT, READ CHARACTER AT CURSOR.
2138          ;AN ERROR IS REPORTED IF THE LAST READ IS NOT AN 'A'.
2139          ;*****
2140
2141          ;*****
2142 006430 000004          IST12: SCOPE
2143 006432 012737 000005 001160          MOV #5,$TIMES          ;;DO 5 ITERATIONS
2144 006440 012737 006446 001106          MOV #CURS1B,$LPADR          ;;SET SCOPE LOOP ADDRESS
2145
2146 006446 013701 015502          CURS1B: MOV TBBUF,R1
2147 006452 004037 015512          JSR R0,RESETV          ;RESET THE UNIT AND WAIT FOR XON.
2148 006456 012721 000101          MOV #101,(R1)+          ;A
2149 006462 113721 002132          MOVB ESCN,(R1)+
2150 006466 113721 001770          MOVB CLFT,(R1)+          ;CURSOR LEFT
2151 006472 013721 002054          MOV ESCOI,(R1)+
2152 006476 013711 002064          MOV TCUCH,(R1)          ;TRANSMIT CH. AT CURSOR
2153 006502 012737 000006 015510          MOV #6,XMCNT          ;SET UP TO XMIT 6 CHARACTERS
2154 006510 004037 016120          JSR R0,XMREC          ;XMIT STRING AND WAIT FOR EOM.
2155 006514 000402          BR 10$          ;NORML EXIT
2156 006516 104011          ERROR 11          ;LAST XMIT CAUSED VT61 TO HANG/FAIL
2157 006520 000430          BR 2$          ;EXIT TEST
2158 006522 127727 006446 000101 10$: CMPB @RBBUF,#101          ;CHARACTER READ=A
2159 006530 001424          BEQ 2$          ;YES-NEXT SUBTEST
2160 006532 013737 002054 001124          MOV ESCOI,$GDDAT
2161 006540 000337 001124          SWAB $GDDAT          ;REASSEMBLE ESC DATA
2162 006544 005037 001126          CLR $BDDAT
2163 006550 113737 002064 001127          MOVB TCUCH,$BDDAT+1          ;LOAD FAILING ESC SEQUENCE
2164 006556 104001          ERROR 1          ;AND ISSUE IT
2165 006560 004037 016076          JSR R0,CLREG
2166 006564 112737 000101 001124          MOVB #101,$GDDAT          ;LOAD GOOD EH. AND CH.
2167 006572 117737 006376 001126          MOVB @RBBUF,$BDDAT
2168 006600 104004          ERROR 4          ;READ AND ISSUE THEM.
2169
2170 006602 000240          2$: NOP          ;END OF TEST
2171          ;*****
2172          ;ROUTINE TO VERIFY OPERATION OF REPLACE AND INSERT MODE.
2173          ;INITIALLY ROW 0 IS WRITTEN TO 80 INCREMENTING CHAR.
2174          ;ON THE FIRST PASS(REPLACE MODE) A CHARACTER IS REPLACED
2175          ;AT HOME AND THE CHAR. AT ROW0,COL.0(172) AND ROW1,COL0(NULL)
2176          ;ARE VERIFIED. ON THE SECOND PASS, INSERT MODE IS ENTERED
2177          ;AND THE RESULTING INSERTION(AT HOME) IS VERIFIED.ROW0,COL0
2178          ;SHOULD BE 172 AND ROW1,COL0 SHOULD BE 161.
2179          ;*****
2180
2181          ;*****
2182 006604 000004          IST13: SCOPE
2183 006606 012737 000005 001160          MOV #5,$TIMES          ;;DO 5 ITERATIONS
2184 006614 012737 006622 001106          MOV #INRPL,$LPADR          ;;SET SCOPE LOOP ADDRESS
2185
2186 006622 004037 015512          INRPL: JSR R0,RESETV          ;RESET THE UNIT

```

2187	006626	013701	015502			MOV	TBBUF,R1	
2188	006632	005201				INC	R1	;LEAVE ROOM IN BUFFER FOR SOM.
2189	006634	012703	000120			MOV	#80.,R3	;CREATE A LINE OF 80 INCREMENTING
2190	006640	004037	017504			JSR	RO,BLDINC	;CHAR. ON THE SCREEN.
2191	006644	105011				CLRB	(R1)	
2192	006646	013703	015502			MOV	TBBUF,R3	
2193	006652	004037	016460			JSR	RO,LDXMIT	
2194	006656	005005				CLR	R5	;USE R5 AS TEST INDEXER.
2195	006660	012737	000002	016366	INAG:	MOV	#2,RECITT	;SET UP TO RECEIVE 2 CHAR.
2196	006666	012737	031302	016372		MOV	#TCRLB+200,BYSTOR	;SET UP STORAGE AREA.
2197	006674	013701	015502			MOV	TBBUF,R1	
2198	006700	004037	020172			JSR	RO,LDBUF	;LOAD THE BUFFER WITH:
2199	006704	033	110	172		.BYTE	.ESC,.CHOM,172,.ESC,.CHOM,.ESC,.O,.TCUCH	
2200	006707	033	110	033				
2201	006712	117	127					
2202	006714	033	102	033		.BYTE	.ESC,.CDWN,.ESC,.O,.TCUCH,0	
2203	006717	117	127	000				
2204	006722	012737	000015	015510		MOV	#13.,XMCNT	;SET UP TO XMIT 13 CAHR.
2205	006730	004037	016120			JSR	RO,XMREC	
2206	006734	000402				BR	1\$	;NORMAL EXIT
2207	006736	104011				ERROR	11	;LAST XMIT CAUSED UNIT TO HANG.
2208	006740	000433				BR	INRXT	;EXIT TEST.
2209	006742	026537	007020	031302	1\$:	CMP	TDATA(R5),TCRLB+200	;COMPARE GOOD TO REC.DATA.
2210	006750	001407				BEQ	2\$	;GOOD-LOOP OR EXIT.
2211	006752	016537	007012	001126		MOV	TFUNCT(R5),\$BDDAT	
2212	006760	013737	002120	001124		MOV	ESCP,\$GDDAT	;LOAD ESCAPE SEQ. ERROR.
2213	006766	104001				ERROR	1	
2214	006770	005725			2\$:	TST	(R5)+	;INCREMENT INDEXER.
2215	006772	020527	000004			CMP	R5,#4	;THRU WITH TEST?
2216	006776	001414				BEQ	INRXT	;YES-EXIT.
2217	007000	012703	007024			MOV	#ENSRT,R3	;NO-SECOND PASS- ENTER
2218	007004	004037	016460			JSR	RO,LDXMIT	;INSERT MODE AND DO AGAIN.
2219	007010	000723				BR	INAG	
2220								
2221	007012	000151	000111	177777	TFUNCT:	.WORD	.ERPL,.EINST,-1	
2222	007020	172	000	172	TDATA:	.BYTE	172,0,172,160	
2223	007023	160						
2224	007024	033	120	111	ENSRT:	.BYTE	.ESC,.P,.EINST,0	
2225	007027	000						
2226	007030	000240			INRXT:	NOP		
2227								
2228					;;*****			
2229								
2230								
2231								
2232								
2233								
2234								
2235								
2236								
2237								
2238					;;*****			
2239								
2240					;;*****			
2241	007032	000004			TST14:	SCOPE		
2242	007034	012737	000005	001160		MOV	#5,\$TIMES	;;DO 5 ITERATIONS

```

2243 007042 012737 007050 001106      MOV      #CKSCRA,$LPADR  ;;SET SCOPE LOOP ADDRESS
2244
2245 007050 004037 015512      CKSCRA: JSR      R0,RESETV      ;RESET THE UNIT.
2246 007054 013701 015502      MOV      TBBUF,R1
2247 007060 004037 020172      JSR      R0,LDBUF      ;LOAD THE XMIT BUFFER WITH:
2248 007064      060      033      102      .BYTE      60,,ESC,,CDWN,,ESC,,CLFT,61,,ESC,,Y,,R23,,C00
2249 007067      033      104      061
2250 007072      033      131      067
2251 007075      040
2252 007076      012      033      110      .BYTE      .LNFED,,ESC,,CHOM,,ESC,,O,,TCURH,,RFL,0
2253 007101      033      117      127
2254 007104      007      000
2255 007106 012737 000020 015510      MOV      #16,,XMCNT      ;SET UP TO XMIT 16 BYTES.
2256 007114 004037 016120      JSR      R0,XMREC
2257 007120 000402      BR      1$      ;NORMAL EXIT
2258 007122 104011      ERROR      11      ;LAST XMIT CAUSED UNIT TO HANG.
2259 007124 000452      BR      GDSCRL      ;EXIT TEST.
2260 007126 127727 006042 000061 1$: CMPB      @RBBUF,#61      ;CHARACTER AT HOME A 1?
2261 007134 001401      BEQ      CKSCRB      ;YES-NEXT TEST
2262 007136 104023      ERROR      23      ;NO-ISSUE NO SCROLL ERROR.
2263 007140 012737 000002 016366 CKSCRB: MOV      #2,RECITT      ;SET UP FOR TWO REC. LOOPS.
2264 007146 012737 031302 016370      MOV      #TCRLB+200,WDSTOR      ;SET UP CURSOR POSITION STROAGE.
2265 007154 013701 015502      MOV      TBBUF,R1
2266 007160 004037 020172      JSR      R0,LDBUF      ;LOAD XMIT BUFFER WITH:
2267 007164      033      131      067      .BYTE      .ESC,,Y,,R23,,C79,101,,ESC,,O,,RDCUR
2268 007167      157      101      033
2269 007172      117      131
2270 007174      033      110      033      .BYTE      .ESC,,CHOM,,ESC,,O,,TCUCH,0
2271 007177      117      127      000
2272 007202 012737 000015 015510      MOV      #13,,XMCNT      ;SET UP TO XMIT 13 BYTES.
2273 007210 004037 016120      JSR      R0,XMREC      ;XMIT AND WAIT FOR RECEIVED DONE.
2274 007214 000402      BR      1$
2275 007216 104011      ERROR      11      ;LAST XMIT CAUSED VT61 TO HANG.
2276 007220 000414      BR      GDSCRL      ;ERROR EXIT
2277 007222 127737 005746 002170 1$: CMPB      @RBBUF,ZERO      ;NULL RECEIVED?
2278 007230 001410      BEQ      GDSCRL      ;YES-EXIT TEST
2279 007232 104023      ERROR      23      ;NO-ISSUE NO SCROLL ERROR.
2280 007234 013777 031302 005732      MOV      TCRLB+200,@RBBUF      ;LOAD RECEIVED CURSOR POSITION.
2281 007242 013746 002144      MOV      R23C00,-(SP)      ;;PUSH R23C00 ON STACK
2282 007246 004037 016600      JSR      R0,CURER      ;GO ISSUE CURSOR ERROR.
2283 007252 000240      GDSCRL: NOP
2284
2285
2286      ;;*****
2287      ;THIS TEST INSURES THAT THE VT61 CURSOR CAN BE
2288      ;POSITIONED TO VERY POSSIBLE ROW/COLUMN POSITON
2289      ;ON THE SCREEN. THIS IS TESTED BY FILLING THE
2290      ;COMPLETE SCREEN WITH A CHARACTER(A) AND THEN
2291      ;POSITONING THE CURSOR (VIA DCA) TO EVERY POSITION
2292      ;AND THE 'A' AT THAT POSITION IS REPLACED WITH A SPACE.
2293      ;THE SCREEN IS THEN READ TO VERIFY THAT ONLY SPACES
2294      ;EXIST ON THE SCREEN. ALL POSITIONS CONTAINING
2295      ;NON-SPACES ARE REPORTED.
2296      ;;*****
2297
2298

```

```

2299 *****
2300 007254 000004 1ST15: SCOPE
2301 007256 012737 000001 001160 MOV #1,$TIMES ;;DO 1 ITERATION
2302 007264 012737 007272 001106 MOV #CURS2,$LPADR ;;SET SCOPE LOOP ADDRESS
2303
2304 007272 042737 077577 002224 CURS2: BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2305 007300 004037 015512 JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
2306 007304 012702 003600 MOV #TOTCH,R2 ;LOAD A COUNT OF SCREEN(1920).
2307 007310 112777 000002 006170 MOV #SOM,@TBUFP ;ISSUE START OF MESSAGE.
2308 007316 004037 016374 JSR R0,XMIT1
2309 007322 005302 1$: DEC R2 ;DECREMENT XMIT COUNT
2310 007324 001413 BEQ 10$ ;COUNT = ZERO?
2311
2312 007326 112777 000101 006152 MOV #101,@TBUFP ;LOAD THE CHARACTER(A).
2313 007334 004037 016374 JSR R0,XMIT1 ;NO-XMIT ANOTHER BYTE.
2314 007340 023737 002200 002202 CMP FTLCNT,ALWCNT ;EXCEEDED FATAL ERROR COUNT?
2315 007346 103765 BLO 1$ ;NO-CHECK IF DONE NOW
2316 007350 000137 010022 JMP C2XT ;YES-ABORT TESTING THIS UNIT.
2317 007354 112777 000004 006124 10$: MOV #EOM,@TBUFP ;ISSUE END OF MESSAGE.
2318 007362 004037 016374 JSR R0,XMIT1
2319 007366 004037 016520 JSR R0,RESPTR ;RESET INTERRUPT POINTERS.
2320 007372 013737 002160 017002 MOV R23C78,LNRW ;SET UP 1ST ADDRESS
2321 007400 013701 015502 MOV TBUF,R1 ;LOAD XMIT BUFFER WITH
2322 007404 013721 002040 MOV DCRAD,(R1)+
2323 007410 010102 MOV R1,R2 ;R2 POINTS TO CURSOR ADD. IN BUFFER
2324 007412 013721 002160 MOV R23C78,(R1)+ ;CURSOR TO LOWER RIGHT -1.
2325 007416 112721 000040 MOV #40,(R1)+ ;SPACE
2326 007422 012737 000005 015510 2$: MOV #5,XMCNT ;SET UP TO XMIT 5 CHARACTERS
2327 007430 042737 077577 002224 BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2328 007436 052777 000100 172272 BIS #TENA,@VXCSR ;XMIT INTERRUPTS.
2329 007444 012746 000001 MOV #XMDNE,-(SP) ;PUSH #XMDNE ON STACK
2330 007450 005737 002232 TST PTYPE ;PROCESSOR TYPE
2331 007454 001407 BEQ 51$ ;IF 0 THEN 11/45-11/70
2332 007456 100403 BMI 50$ ;IF -1 THEN 11/35,40
2333 007460 012746 000001 MOV #1,-(SP) ;PUSH #1. ON STACK
2334 007464 000405 BR 53$
2335 007466 50$: MOV #2,-(SP) ;PUSH #2. ON STACK
2336 007466 012746 000002 BR 53$
2337 007472 000402 51$: MOV #4,-(SP) ;PUSH #4. ON STACK
2338 007474 012746 000004 53$: JSR R0,WIBGND ;LOOK FOR XMIT DONE
2339 007474 012746 000004 021076 BR C2XT ;NOT FOUND-ERROR EXIT
2340 007500 004037 021076 CMP (R2),CUHME ;DELETED TO HOME?
2341 007504 000546 BEQ 3$ ;YES
2342 007506 021237 002154 JSR R0,CMPOS ;NO-GET NEXT POSITION TO BE DELETED
2343 007512 001405 MOV LNRW,(R2) ;LOAD IT IN XMIT BUFFER
2344 007514 004037 016676 BR 2$ ;AND DELETE IT.
2345 007520 013712 017002 JSR R0,RESPTR ;RESET INTERRUPT POINTERS
2346 007524 000736 3$: MOV CUHME,LNRW ;LOAD INITIAL CHECK POSITION(HOME)
2347 007526 004037 016520 MOV #1001,BLKM ;SET UP TO XMIT A SOM -DATA- EOM.
2348 007532 013737 002154 017002 MOV TBUF,R1 ;LOAD XMIT BUFFER WITH
2349 007540 012737 001001 002226 MOV R1,R2 ;STORE ERRORS IN XMIT BUFFER
2350 007546 013701 015502 BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL
2351 007552 010102 MOV ESCN,(R1)+
2352 007554 042737 077577 002224 MOV CHOM,(R1)+ ;CURSOR HOME
2353 007562 013721 002132
2354 007566 013721 001762

```

2355	007572	013721	002054		MOV	ESCO,(R1)+		
2356	007576	013721	002056		MOV	XMTAL,(R1)+	;TRANSMIT ALL	
2357	007602	012737	000005	015510	MOV	#5,XMCNT		
2358	007610	052777	000100	172120	BIS	#TENA,@VXCSR	;SET XMIT ENABLE	
2359	007616	012746	000001		MOV	#XMDNE,-(SP)	;PUSH #XMDNE ON STACK	
2360	007622	005737	002232		TST	P1YPE	;PROCESSOR TYPE	::**C
2361	007626	001407			BEQ	61\$	;IF 0 THEN 11/45-11/70	::**C
2362	007630	100403			BMI	60\$	;IF -1 THEN 11/35,40	::**C
2363	007632	012746	000001		MOV	#1,-(SP)	;PUSH #1. ON STACK	
2364	007636	000405			BR	63\$		::**C
2365	007640			60\$:				
2366	007640	012746	000003		MOV	#3,-(SP)	;PUSH #3. ON STACK	
2367	007644	000402			BR	63\$		::**C
2368	007646			61\$:				
2369	007646	012746	000006		MOV	#6,-(SP)	;PUSH #6. ON STACK	
2370	007652	004037	021076		JSR	RO,WTBGND	;LOOK FOR SOM OR XMIT DONE	::**C
2371	007656	000461			BR	C2XT	;NOT FOUND-ERROR EXIT	
2372	007660	013701	015200		MOV	RBUF,R1	;SET UP RECEIVE FLAG	
2373	007664	005037	002220		CLR	DLAY	;SET UP TIME OUT DELAY	
2374	007670	020137	015200		40\$:	CMP	R1,RBUF	;CHARACTER RECEIVED?
2375	007674	103411			BLO	41\$	;YES-GO CHECK IT.	
2376	007676	032737	020000	002224	BIT	#REOM,VSTAT	;LOOK FOR END OF MESSAGE	
2377	007704	001025			BNE	C2CK	;FOUND IT, EXIT TEST	
2378	007706	005337	002220		DEC	DLAY	;RUN TIME OUT DELAY.	
2379	007712	001366			BNE	40\$	;AND LOOK FOR RECEIVED CH.	
2380	007714	104011			ERROR	11	;LAST XMIT CAUSED VT61 TO HANG.	
2381	007716	000420			BR	C2CK	;GO SEE IF ANY ERRORS STORED.	
2382	007720	013737	015174	015200	41\$:	MOV	RBBUF,RBUF	;RESET RECEIVE POINTER
2383	007726	127727	005242	000040	CMPB	@RBBUF,#40	;CHAR EQUAL A SPACE?	
2384	007734	001003			BNE	6\$	;NOT A SPACE-MUST BE ERROR-STORE 11	
2385	007736	004037	016740		5\$:	JSR	RO,CPP0S	;UPDATE CURSOR POSITION
2386	007742	000746			BR	4\$		
2387	007744	022702	031126		6\$:	CMP	#TCRLB+20.,R2	;STORED 10 ERRORS?
2388	007750	101772			BLOS	5\$	;YES-IGNORE ANY FURTHER ERRORS.	
2389	007752	013722	017002		MOV	LNRW,(R2)+	;STORE FAILING CURSOR POSITION	
2390	007756	000767			BR	5\$		
2391								
2392	007760	020237	015502		2CK:	CMP	R2,TBBUF	;ANY ERRORS STORED?
2393	007764	001416			BEQ	C2XT	;NO EXIT TEST	
2394	007766	013701	015502		MOV	TBBUF,R1	;USE R1 AS ERROR POINTER	
2395	007772	021137	002042		1\$:	CMP	(R1),R23C79	;CURSOR TO LOWER RIGHT?
2396	007776	001411			BEQ	C2XT	;YES-NOT AN ERROR.	
2397	010000	104012			ERROR	12	;NO-ISSUE ERROR MESSAGES	
2398	010002	012746	020040		MOV	#20040,-(SP)	;PUSH #20040 ON STACK	
2399	010006	012177	005162		MOV	(R1)+,@RBBUF	;LOAD FAILING POS.	
2400	010012	004037	016600		JSR	RO,CURER	;ISSUE CURSOR ERROR	
2401	010016	020102			CMP	R1,R2	;DONE WITH ERRORS?	
2402	010020	103764			BLO	1\$	;NO, DUMP ANOTHER.	
2403	010022	000240			2XT:	NOP	;EXIT TEST	
2404								
2405								
2406								
2407								
2408								
2409								
2410								

 ;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 FEEL IS ISSUED
 ;THE CURSOR POSITION IS THEN READ AND MUST BE ROW1,COL20.
 ;A CARRIAGE RETURN IS THEN ISSUED AND CURSOR POSITION VERIFIED

```

2411          ;TO BE ROW1,COLO.
2412
2413          ;:*****
2414
2415          ;:*****
2416 010024 000004          1ST16: SCOPE
2417 010026 012737 000005 001160      MOV      #5,$TIMES      ;;DO 5 ITERATIONS
2418 010034 012737 010042 001106      MOV      #NWLN,$LPADR    ;;SET SCOPE LOOP ADDRESS
2419
2420 010042 004037 015512          NWLN: JSR      R0,RESETV      ;RESET THE UNIT AND ENTER MAINT.MODE
2421 010046 013701 015502          MOV      TBBUF,R1
2422 010052 004037 020172          JSR      R0,LDBUF      ;LOAD XMIT BUFFER WITH-
2423 010056      033      131      040      .BYTE      .ESC,.Y,.R00,.C20
2424 010061      064
2425 010062      012      033      117      .BYTE      .LNFED,.ESC,.O,.RDCUR,.BEL,O
2426 010065      131      007      000
2427 010070 012737 000011 015510      MOV      #9.,XMCNT      ;SETUP TO XMIT 9 CHARACTERS
2428 010076 004037 016120          JSR      R0,XMREC      ;GO DO IT
2429 010102 000402          BR      30$      ;NORMAL EXIT.
2430 010104 104011          ERROR      11      ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
2431 010106 000454          BR      4$      ;EXIT TEST
2432 010110 023777 002136 005056 30$: CMP      R01C20,@RBBUF    ;CHECK CURSOR POS. S/B ROW 1, COL 20.
2433 010116 001412          BEQ      3$
2434 010120 005037 001124          CLR      $GDDAT
2435 010124 013737 001754 001126      MOV      LNFED,$BDDAT
2436 010132 104001          ERROR      1      ;ISSUE IT
2437 010134 013746 002136          MOV      R01C20,-(SP)      ;;PUSH R01C20 ON STACK
2438 010140 004037 016600          JSR      R0,CURER      ;SETUP AND ISSUE CURSOR ERROR
2439 010144 013701 015502          3$: MOV      TBBUF,R1
2440 010150 013721 001752          MOV      CARRT,(R1)+      ;LOAD XMIT BUFFER WITH
2441 010154 013721 002054          MOV      ESCOI,(R1)+      ;CARRIAGE RETURN, READ CURSOR
2442 010160 013721 002050          MOV      RDCUR,(R1)+      ;POSITION
2443 010164 012737 000004 015510      MOV      #4,XMCNT      ;SET UP TO TRANSMIT 4 CHARACTERS
2444 010172 004037 016120          JSR      R0,XMREC      ;GO DO IT
2445 010176 000402          BR      40$      ;NORMAL EXIT.
2446 010200 104011          ERROR      11      ;TRANSMISSION CAUSED VT61 TO FAIL/HANG
2447 010202 000416          BR      4$      ;EXIT TEST
2448 010204 023777 002134 004762 40$: CMP      R01C00,@RBBUF    ;CHECK CURSOR POS. S/B ROW1, COL 0.
2449 010212 001412          BEQ      4$      ;EXIT TEST IF GOOD.
2450 010214 005037 001124          CLR      $GDDAT
2451 010220 013737 001752 001126      MOV      CARRT,$BDDAT
2452 010226 104001          ERROR      1      ;ISSUE IT
2453 010230 013746 002134          MOV      R01C00,-(SP)      ;;PUSH R01C00 ON STACK
2454 010234 004037 016600          JSR      R0,CURER      ;SET UP AND ISSUE CURSOR ERROR
2455 010240 C00240          4$: NOP
2456
2457          ;:*****
2458
2459          ;ROUTINE TO VERIFY PROPER OPERATION OF ERASE TO END-OF-
2460          ;SCREEN. SCREEN IS WRITTEN TO 1920 INCREMENTING CHAR.
2461          ;ERASE TO END OF SCREEN IS THEN ISSUED AND THE
2462          ;ENTIRE SCREEN IS READ VERIFYING THAT IT IS ALL NULLS.
2463          ;:*****
2464
2465          ;:*****
2466

```

```

2467 010242 000004 TST17: SCOPE
2468 010244 012737 000003 001160 MOV #3,$TIMES ;;DO 3 ITERATIONS
2469 010252 012737 010260 001106 MOV #ERSE,$LPADR ;;SET SCOPE LOOP ADDRESS
2470
2471
2472 010260 004037 015512 ERSE: JSR R0,RESETV ;RESET THE UNIT -SET MAINT. MODE.
2473 010264 005077 004704 CLR @RBBUF ;CLEAR THE CHECK LOCATION.
2474 010270 004037 017532 JSR R0,DATSC ;FILL THE SCREEN.
2475 010274 013701 015502 MOV TBBUF,R1
2476 010300 004037 020172 JSR R0,LDBUF ;LOAD XMIT BUFFER WITH:
2477 010304 033 110 033 .BYTE .ESC,.CHOM,.ESC,.EOS,.ESC,.O,.XMTAL,O
2478 010307 112 033 117
2479 010312 126 000
2480 010314 113737 002132 001125 MOVB ESCN,$GDDAT+1
2481 010322 113737 001774 001124 MOVB EOS,$GDDAT ;LOAD ERROR WITH ERASE TO EOS
2482 010330 005037 001126 CLR $BDDAT
2483 010334 005077 004634 CLR @RBBUF
2484 010340 012737 000007 015510 MOV #7,XMCNT ;SET UP TO XMIT 7 BYTES
2485 010346 004037 016120 JSR R0,XMREC ;XMIT AND WAIT FOR REC. DONE
2486 010352 000402 BR 5$
2487 010354 104011 ERROR 11 ;ESC ERROR
2488 010356 000413 BR ERSXT ;EXIT TEST
2489 010360 127737 004610 002170 5$: CMPB @RBBUF,ZERO ;VT61 XMITTED SOM/EOM ONLY?
2490 010366 001407 BEQ ERSXT ;YES-EXIT TEST.
2491 010370 104001 ERROR 1 ;NO-ERASE TO END OF SCREEN
2492 010372 004037 016076 JSR R0,CLREG ;GO CLEAR ERROR STORAGE
2493 010376 117737 004572 001126 MOVB @RBBUF,$BDDAT
2494 010404 104004 ERROR 4 ;ISSUE DATA ERROR
2495 010406 000240 ERSXT: NOP
2496
2497
2498
2499
2500 ;*****
2501 ;ROUTINE TO SET UP END OF PASS INDICATION.
2502 ;SELF TEST(ESC P T) IS ISSUED TO THE UNIT UNDER TEST
2503 ;AND AN ERROR IS ISSUED IF THE UNIT CANNOT RESPOND AFTER
2504 ;SELF TEST IS COMPLETE. IF SELF TEST IS SUCCESSFUL THE
2505 ;SCREEN IS WRITTEN TO 23 LINES OF INCREMENTING CHARACTERS
2506 ;AND 23 LINES OF INCREMENTING CHAR. IN REVERSE VIDEO.
2507 ;THE IDENT IS THEN CHECKED AND IF A COPIER IS PRESENT A
2508 ;COPY SCREEN COMMAND IS ISSUED(NOTE: THIS COMMAND WILL CAUSE
2509 ;THE UNIT TO BE 'BUSY' AND NOT RESPOND TO ANY FURTHER COMMANDS
2510 ;UNTIL THE SCREEN HAS BEEN COMPLETELY COPIED.)
2511 ;*****
2512 ;*****
2513 010410 000004 TST20: SCOPE
2514 010412 012737 000001 001160 MOV #1,$TIMES ;;DO 1 ITERATION
2515 010420 012737 010426 001106 MOV #LSTST,$LPADR ;;SET SCOPE LOOP ADDRESS
2516
2517 LSTST:
2518 010426 013746 002170 MOV ZERO,-(SP) ;;PUSH ZERO ON STACK
2519 010432 013746 002122 MOV TSTER,-(SP) ;;PUSH TSTER ON STACK
2520 010436 013746 002054 MOV ESCO,-(SP) ;;PUSH ESCO ON STACK
2521 010442 004037 013666 JSR R0,TESC ;TRANSMIT IT.
2522 010446 004037 015630 JSR R0,GETON ;GO LOOK FOR A XON.

```

2523	010452	000407			BR	18		:VT61RESPONDED-NOT HUNG	
2524	010454	013737	001732	001124	MOV	VRCR,\$GDDAT		:LOAD THE ADDRESS	
2525	010462	013737	001742	001126	MOV	VECT,\$BDDAT		:LOAD THE VECTOR	
2526	010470	104010			ERROR	10		:REPORT SELF TEST FAILURE	
2527	010472	004037	015512		JSR	RO,RESETV	18:	:RESET AND SET MAINT. MODE.	
2528	010476	005037	002210		CLR	BUBCT		:SET UP HALF-SCREEN FLAG.	
2529	010502	042737	077577	002224	BIC	#77577,VSTAT	28:	:CLEAR ALL FLAGS BUT XOFF AND XMKIL.	
2530	010510	012737	001001	002226	MOV	#1001,BLKM		:SET UP TO XMIT A SOM -DATA- EOM.	
2531	010516	013701	015502		MOV	*3BUF,R1		:SET UP BEG. OF XMIT BUFFER	
2532	010522	012703	000500		MOV	#320.,R3		:FILL BUFFER WITH INCREMENTING CHAR.	
2533	010526	004037	017504		JSR	RO,BLDINC			
2534	010532	012737	001700	015510	MOV	#960.,XMCNT		:SEND 12 LINE TO VT61	
2535	010540	052777	000100	171170	BIS	#TENA,@VXCSR		:ENABLE XMIT INTERRUPTS	
2536	010546	012746	000001		MOV	#XMDNE,-(SP)		:PUSH #XMDNE ON STACK	
2537	010552	005737	002232		TST	P1YPE		:PROCESSOR TYPE	::**C
2538	010556	001407			BEQ	58		:IF 0 THEN 11/45-11/70	::**C
2539	010560	100403			BMI	48		:IF -1 THEN 11/34,40	::**C
2540	010562	012746	000031		MOV	#25.,-(SP)		:PUSH #25. ON STACK	
2541	010566	000405			BR	68			::**C
2542	010570						48:		
2543	010570	012746	000062		MOV	#50.,-(SP)		:PUSH #50. ON STACK	
2544	010574	000402			BR	68			::**C
2545	010576						58:		
2546	010576	012746	000144		MOV	#100.,-(SP)		:PUSH #100. ON STACK	
2547	010602	004037	021076		JSR	RO,WIBGND	68:	:LOOK FOR XMDNE.	::**C
2548	010606	000430			BR	ENDSEL		:NOT FOUND-EXIT.	
2549	010610	005737	002210		TST	BUBCT		:DONE WITH SCREEN?	
2550	010614	001007			BNE	38		:YES-EXIT	
2551	010616	012703	005460		MOV	#SETREV,R3		:NO-ISSUE ENTER REVERSE VIDEO	
2552	010622	004037	016460		JSR	RO,LDXMIT		:ESCAPE SEQUENCE.	
2553	010626	005237	002210		INC	BUBCT		:INCREMENT SCREEN HALF FLAG.	
2554	010632	000723			BR	28		:AND ISSUE SECOND HALF IN REV. VIDEO.	
2555	010634	032737	000001	002124	BIT	#BIT00,IDENT	38:	:IDENT = COPIER?	
2556	010642	001412			BEQ	ENDSEL		:NO	
2557	010644	013746	002170		MOV	ZERO,-(SP)		:PUSH ZERO ON STACK	
2558	010650	012746	000135		MOV	#.CPYSC,-(SP)		:PUSH #.CPYSC ON STACK	
2559	010654	013746	002132		MOV	ESCN,-(SP)		:PUSH ESCN ON STACK	
2560	010660	004037	013666		JSR	RO,TEEC			
2561	010664	004037	020200		JSR	RO,CKABRT		:CHECK FOR A PERIPHERAL ABORT.	
2562	010670	105737	002176		ENDSEL: TSTB	MODE		:IF IN MAN MODE DO NOT ENTER EOP.	
2563	010674	001402			BEQ	ENDPS			
2564	010676	000137	003100		JMP	ASTRT			
2565	010702	042777	000100	171022	ENDPS: BIC	#RDENA,@VRCR		:CLEAR REC.INT. BEFORE NEXT UNIT SELECT.	
2566					.SBTTL	END OF PASS ROUTINE			
2567									
2568									
2569									
2570									
2571									
2572									
2573									
2574									
2575	010710				\$EOP:				
2576	010710	000004			SCOPE				
2577	010712	005037	001102		CLR	\$TSTNM		:ZERO THE TEST NUMBER	
2578	010716	005037	001160		CLR	\$TIMES		:ZERO THE NUMBER OF ITERATIONS	

```

2579 010722 005237 001100      INC    $PASS      ;;INCREMENT THE PASS NUMBER
2580 010726 042737 100000 001100 BIC    #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
2581 010734 005327          DEC    (PC)+      ;;LOOP?
2582 010736 000001          SEOPCT: .WORD    1
2583 010740 003022          BGT    $DOAGN      ;;YES
2584 010742 012737          MOV    (PC)+,@(PC)+ ;;RESTORE COUNTER
2585 010744 000001          SENDCT: .WORD    1
2586 010746 010736          SEOPCT
2587 010750 104401 011015      TYPE    $ENDMG      ;;TYPE 'END PASS #'
2588 010754 013746 001100      MOV    $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
2589 010760 104405          TYPDS      ;;GO TYPE--DECIMAL ASCII WITH SIGN
2590 010762 104401 011012      TYPE    ,SENULL      ;;TYPE A NULL CHARACTER
2591 010766 013700 000042      $GET42: MOV    @#42,R0      ;;GET MONITOR ADDRESS
2592 010772 001405          BEQ    $DOAGN      ;;BRANCH IF NO MONITOR
2593 010774 000005          RESET      ;;CLEAR THE WORLD
2594 010776 004710          SENDAD: JSR    PC,(R0)      ;;GO TO MONITOR
2595 011000 000240          NOP      ;;SAVE ROOM
2596 011002 000240          NOP      ;;FOR
2597 011004 000240          NOP      ;;ACT11
2598 011006          $DOAGN:
2599 011006 000137          JMP    @(PC)+      ;;RETURN
2600 011010 002534          $RTNAD: .WORD    MODCA
2601 011012 377 377 000      $ENULL: .BYTE   -1,-1,0      ;;NULL CHARACTER STRING
2602 011015 015 042412 042116 $ENDMG: .ASCII  <15><12>/END PASS #/
2603 011022 050040 051501 020123
2604 011030 000043
2605
2606      ;;*****
2607      ;ROUTINE TO ECHO THE KEYBOARD. KEYS FOR TAB,BELL,CARRIAGE
2608      ;AND LINE FEED ECHO A MNEMONIC, NON-DISPLAY CHAR. ECHO OCTAL
2609      ;EQUIVALENTS AND DISPLAY CHAR. ECHO THEMSELVES.
2610      ; (EXAMPLES-CHAR.,SPACE,ESC,SPACE OR 037,SPACE.) A
2611      ;CONTROL C (003) WILL CAUSE A TEST EXIT.
2612
2613      ;;*****
2614
2615      ;;*****
2616      TST21: SCOPE
2617      MOV    #1,$TIMES      ;;DO 1 ITERATION
2618      MOV    #KYBD,$LPADR    ;;SET SCOPE LOOP ADDRESS
2619
2620      KYBD: JSR    R0,RESPTR
2621      MOV    #DKYBD,R2      ;;LOAD MESSAGE ADDRESS INR2
2622      JSR    R0,DSMES      ;;DISPLAY KEYBOARD MESSAGE
2623      MOV    #DCNTZ,R3      ;;ISSUE CONTROL C EXIT MESSAGE
2624      JSR    R0,LDXM!T
2625      MOV    #EXMAIN,R3
2626      JSR    R0,LDXMIT      ;;ISSUE EXIT MAINTENANCE MODE.
2627      KYSTRT: BIC    #77577,VSTAT      ;;CLEAR ALL FLAGS BUT XOFF AND XMR11.
2628      TSTB    @ABUF      ;;SEE IF A CHAR. RECEIVED
2629      BNE     11$          ;;YES-GO PROCESS IT
2630      WAIT    11$          ;;WAIT FOR A CH.
2631      MOVB    @ABUF,R1      ;;GET RAW RECEIVED DATA
2632      JSR     R0,EXTST      ;;CHECK FOR EXIT CONDITIONS
2633      BR      10$          ;;NO EXIT -CONTINUE.
2634      JMP     ASTR1      ;;EXIT TEST 4
2635      CLRB    @ABUF      ;;CLEAR CHAR FROM BUFFER

```

2635	011144	032737	000400	002224	BIT	#ESC,VSTAT	;CHAR.=ESC(033)?
2636	011152	001405			BEQ	12\$	;NO
2637	011154	005037	015202		CLR	ESAMB	;YES - RESET ESC ASSEMBLY FLAG
2638	011160	012703	026271		MOV	#DESC,R3	;LOAD ESC MESSAGE ADDRESS
2639	011164	000454			BR	KYBXMT	
2640	011166	120127	000041		12\$: CMPB	R1,#41	;CHAR. LESS THAN 41 OR
2641	011172	103415			BLO	2\$	;HIGHER THAN 176, GO ECHO
2642	011174	120127	000176		CMPB	R1,#176	;OCTAL EQUIVALENT
2643	011200	101012			BHI	2\$	
2644	011202	110177	004300		MOVB	R1,@TBUF	;LOAD CHAR. IN XMIT BUFF.
2645	011206	004037	016374		JSR	RO,XMIT1	;GO XMIT IT
2646	011212	112777	000040	004266	MOVB	#40,@TBUF	;LOAD A SPACE
2647	011220	004037	016374		JSR	RO,XMIT1	;AND XMIT IT.
2648	011224	000727			BR	KYSTRT	
2649	011226	120137	001750		2\$: CMPB	R1,BEL	;CHAR.=BELL?
2650	011232	001003			BNE	3\$	
2651	011234	012703	026545		MOV	#DBELL,R3	;LOAD BELL MESSAGE ADDRESS
2652	011240	000426			BR	KYBXMT	
2653	011242	120137	001756		3\$: CMPB	R1,TAB	;CHAR. =TAB?
2654	011246	001003			BNE	4\$	
2655	011250	012703	026526		MOV	#DTAB,R3	;YES-ECHO 'TAB'
2656	011254	000420			BR	KYBXMT	
2657	011256	123701	001752		4\$: CMPB	CARRT,R1	;CHAR.=CARRIAGE RETURN?
2658	011262	001003			BNE	5\$	
2659	011264	012703	026533		MOV	#DCR,R3	;YES - ECHO 'C/R'.
2660	011270	000412			BR	KYBXMT	
2661	011272	120137	001754		5\$: CMPB	R1,LNFED	;CHAR.=LINE FEED?
2662	011276	001003			BNE	6\$	;NO CHECK FOR CONTROL Z
2663	011300	012703	026540		MOV	#DLF,R3	;YES - ECHO 'L/F'.
2664	011304	000404			BR	KYBXMT	
2665	011306	004037	017674		6\$: JSR	RO,BINOC	;CONVERT BINARY TO OCTAL
2666	011312	012703	002164		MOV	#SVER1,R3	
2667	011316	042737	077577	002224	KYBXMT: BIC	#77577,VSTAT	;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2668	011324	004037	016460		JSR	RO,LDMIT	;GO XMIT BUFFER
2669	011330	000665			BR	KYSTRT	;WAIT FOR NEXT CHAR.
2670							
2671							;SEQUENCE TO EXIT MAINTENANCE MODE.
2672	011332	033	117	141	EXMAIN: .BYTE	.ESC,.O,.DMAIN,0	
2673	011335	000					
2674							
2675							;*****
2676							;ROUTINE TO UTILIZE THE VT61 AS A PRINTER CONTROLLER.
2677							;AFTER TEST MESSAGE IS DISPLAYED, THE TEST WAITS
2678							;FOR A C/R BEFORE ACTUALLY ENTERING TEST. A PATTERN
2679							;OF INCREMENTING, ROLLING CHAR. WILL BE OUTPUTTED UNTIL A
2680							;CONTROL C(003) IS RECEIVED.
2681							;*****
2682							;*****
2683							;*****
2684							;*****
2685	011336	000004			IST22: SCOPE		
2686	011340	012737	000001	001160	MOV	#1,\$TIMES	;DO 1 ITERATION
2687	011346	012737	011354	001106	MOV	#TPRNT,\$LPADR	;SET SCOPE LOOP ADDRESS
2688							
2689	011354	012702	026730		TPRNT: MOV	#DPRNT,R2	;LOAD PRINTER MESSAGE ADDRESS
2690	011360	004037	C17600		JSR	RO,DSMES	;AND ISSUE IT

```

2691 011364 012703 011332      MOV      #EXMAIN,R3
2692 011370 004037 016460      JSR      R0,LDXMIT      ;ISSUE EXIT MAINTENANCE MODE.
2693 011374 004037 017772      JSR      R0,GTCR        ;GO SET CARRIAGE RETURN
2694 011400
2695 011400 013746 002170      3$:      MOV      ZERO,-(SP)      ;;PUSH ZERO ON STACK
2696 011404 013746 002114      MOV      EPNT,-(SP)      ;;PUSH EPNT ON STACK
2697 011410 013746 002132      MOV      ESCN,-(SP)      ;;PUSH ESCN ON STACK
2698 011414 004037 013666      JSR      R0,TEEC
2699 011420 013701 015502      MOV      TBBUF,R1      ;LOAD R1 WITH XMIT BUFFER
2700 011424 012705 000041      4$:      MOV      #41,R5      ;R5=1ST CHAR
2701 011430 042737 077577      5$:      BIC      #77577,VSTAT      ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
2702 011436 013701 015502      MOV      TBBUF,R1
2703 011442 012703 000204      MOV      #132,R3      ;R3= LINE WIDTH
2704 011446 004037 017510      JSR      R0,BLDINA      ;GO BUILD A SLIDING PATTERN.
2705 011452 013721 001752      MOV      CARRT,(R1)+    ;LOAD A C/R AND L/F
2706 011456 013721 001754      MOV      LNFED,(R1)+
2707 011462 012737 000206      15510  MOV      #134,XMCNT      ;SET UP TO XMIT BY BYTES.
2708 011470 052777 000100      BIS      #TENA,@VXCSR
2709 011476 032737 000091      002224  BIT      #XMDNE,VSTAT      ;WAIT FOR XMIT DONE
2710 011504 001774
2711 011506 004037 021022      BEQ      #-6
2712 011512 000402      JSR      R0,EXTST      ;CHECK FOR EXIT REQUEST.
2713 011514 000137 003107      BR      6$            ;NO-CONTINUE
2714 011520 004037 020206      JMP      ASTRT        ;YES-EXIT TEST!
2715 011524 122705 000177      6$:      JSR      R0,CKABRT      ;CHECK FOR A PERIPHERAL ABORT.
2716 011530 001337      CMPB     #177,R5      ;EXCEEDED PATT. LIMIT?
2717 011532 000734      BNE      5$          ;NO
2718      BR      4$        ;YES RESET IT

```

;;*****

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

;;*****

;;*****

```

2729 011534 000004      15510  001160  001106  TST23:  SCOPE
2730 011536 012737 000001      MOV      #1,$TIMES      ;;DO 1 ITERATION
2731 011544 012737 011552      MOV      #LPTST,$LPADR    ;;SET SCOPE LOOP ADDRESS
2732
2733 011552 004037 016520      LPTST:  JSR      R0,RESPTR      ;RESET POINTERS
2734 011556 012702 026553      MOV      #DLOOP,R2      ;LOAD LOOP MESSAGE ADDRESS
2735 011562 004037 017600      JSR      R0,DSMES        ;DISPLAY IT
2736 011566 012703 011332      MOV      #EXMAIN,R3
2737 011572 004037 016460      JSR      R0,LDXMIT      ;ISSUE EXIT MAINTENANCE MODE.
2738 011576 004037 020572      JSR      R0,LOOP        ;GO LOOP VT61
2739 011602 000137 003100      JMP      ASTRT        ;ENTER MAN MODE VIA SCOPE ROUTINE.

```

;;*****

;;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.

2746

```

2747      ;THIS TEST IS RUN IN MAINTENANCE MODE, THEREFORE THE KEYS
2748      ;WILL ECHO THEIR POSITION, NOT THEIR INDICATED MNEMONIC. 10
2749      ;ERRORS WILL CAUSE AN AUTOMATIC EXIT FROM TEST.
2750      ;;*****
2751      ;;*****
2752      TST24: SCOPE
2753      MOV      #1,$TIMES      ;;DO 1 ITERATION
2754      MOV      #PDKBD,$LPADR  ;;SET SCOPE LOOP ADDRESS
2755      PDKBD: MOV      #DKBD ,R2
2756      JSR      R0 ,DSMES      ;DISPLAY KEYBOARD TEST MESSAGE.
2757      CLR      BUBCT          ;CLEAR ERROR COUNT LOCATION.
2758      CLR      R5
2759      DOAROW: MOV      DTTBL(R5),R4      ;SET UP 'GOOD' CHAR. POINTER
2760      MOV      MSTBL(R5),R3
2761      BEQ      FEXIT          ;MESSAGE WAS ZERO-EXIT.
2762      BMI      CLMAIN          ;IF MESSAGE IS -1,CLEAR MAINT. MODE.
2763      JSR      R0,LDXMIT        ;ISSUE 'ROW OR FUNCTION' MESSAGE.
2764      JSR      R0,CKKBD        ;GO CHECK IT.
2765      CMPB     BUBCT,#10.      ;TEN ERROR EXIT?
2766      BLO      1$             ;NO-CONTINUE.
2767      BR       FEXIT          ;YES-EXIT TEST.
2768      1$:      TST      (R5)+      ;INCREMENT OFFSET.
2769      BR       DOAROW          ;NO-DO NEXT ROW/FUNCTION.
2770      FEXIT:  MOV      #DEXT,R2
2771      JSR      R0,DSMES        ;ISSUE EXIT MESSAGE
2772      JMP      ASTR
2773      CLMAIN: MOV      #RSMAN,R3      ;SET UP TO EXIT MAINT. MODE.
2774      JSR      R0,LDXMIT
2775      TST      (R5)+          ;INCREMENT OFFSET.
2776      BR       DOAROW          ;NOW TEST CONTROL AND SHIFT FUNCTIONS.
2777      RSMAN:  .BYTE     .ESC,.O,.DMAIN,0
2778      117      141
2779      033
2780      000
2781      ;TABLE OF MESSAGE ADDRESSES.
2782      MSTBL:  .WORD     DTOP,DSEC,DTHRD,DBOT
2783      .WORD     DSPCE,DKPD,-1,DCONT,DLSHFT,DRSHFT,0
2784      DTTBL:  .WORD     ROW1,ROW2,ROW3,ROW4,SPCB
2785      .WORD     KYPD,0,CNTRA,SHFTA,SHFTA
2786      ;;*****
2787      ;SUBROUTINE TO ALLOW SETUP FROM MULTIPLE ENTRIES
2788      ;;*****
2789      SETA:
2790      012012
  
```

```

2803      .SBTTL INITIALIZE THE COMMON TAGS
2804      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2805      MOV     #SCMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
2806      CLR     (R6)+          ;;CLEAR MEMORY LOCATION
2807      CMP     #SWR,R6 ;;DONE?
2808      BNE     -6              ;;LOOP BACK IF NO
2809      MOV     #STACK,SP      ;;SETUP THE STACK POINTER
2810      ;;INITIALIZE A FEW VECTORS
2811      MOV     #SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2812      MOV     #340,@#IOTVEC+2 ;;LEVEL 7
2813      MOV     #ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2814      MOV     #340,@#EMTVEC+2 ;;LEVEL 7
2815      MOV     #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2816      MOV     #340,@#TRAPVEC+2 ;;LEVEL 7
2817      MOV     #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
2818      MOV     #340,@#PWRVEC+2 ;;LEVEL 7
2819      MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
2820      CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
2821      CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2822      MOVB    #1,$ERMAX      ;;ALLOW ONE ERROR PER TEST
2823      MOV     #.,$LPADR      ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2824      MOV     #.,$LPERR      ;;SETUP THE ERROR LOOP ADDRESS
2825      ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2826      ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2827      MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
2828      MOV     #64,$@#ERRVEC  ;;SET UP ERROR VECTOR
2829      MOV     #DSWR,SWR      ;;SETUP FOR A HARDWARE SWICH REGISTER
2830      MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
2831      CMP     #-1,@SWR       ;;TRY TO REFERENCE HARDWARE SWR
2832      BNE     66$           ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2833      ;;AND THE HARDWARE SWR IS NOT = -1
2834      BR      65$           ;;BRANCH IF NO TIMEOUT
2835      64$: MOV     #65$,(SP)  ;;SET UP FOR TRAP RETURN
2836      RTI
2837      65$: MOV     #SWREG,SWR  ;;POINT TO SOFTWARE SWR
2838      MOV     #DISPREG,DISPLAY
2839      66$: MOV     (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
2840
2841      .SBTTL TYPE PROGRAM NAME
2842      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2843      INC     #-1            ;;FIRST TIME?
2844      BNE     67$           ;;BRANCH IF NO
2845      CMP     #SENDAD,@#42   ;;ACT-11?
2846      BEQ     67$           ;;BRANCH IF YES
2847      TYPE    ,68$          ;;TYPE ASCIZ STRING
2848      BR      67$           ;;GET OVER THE ASCIZ
2849      ;;68$: .ASCIZ <CRLF>*CZVTHCO*<CRLF>
2850      67$:
2851      TYPE    ,STUPM        ;;ISSUE SET-UP MESSAGE.
2852      MOV     #TRPA,@#10     ;;AND VECTOR
2853      SPL     0              ;;PROCESSOR IS 11/45-11/70?
2854      MOV     #8.,PMULT      ;;YES-DELAY MULTIPLIER 8
2855      MOV     #0,PTYPE       ;;11/45-11/70.
2856      BR      RTRP
2857
2858      TRPA: POP2SP          ;NO
  
```

```

2859 012332 012737 012362 000010      MOV      #TRPB,@#10      ;RELOAD TRAP ADDRESS
2860 012340 006737 002162              SXT      CHRDR          ;PROCESSOR IS 11/40 OR 35?
2861 012344 012737 000003 017500      MOV      #3,PMULT        ;YES-DELAY MULTIPLIER=3      ;:++C
2862 012352 012737 177777 002232      MOV      #-1,PTYPE       ;11/35 OR 11/40.      ;:++C
2863 012360 000407              BR        RTRP
2864
2865 012362 022626              TRPB:    POP2SP
2866 012364 012737 000002 017500      MOV      #2,PMULT        ;PROCESSOR MUST BE 11/05-11/54      ;:++C
2867 012372 012737 000001 002232      MOV      #1,PTYPE       ;SET PROCESSOR TYPE      ;:++C
2868 012400 012737 000012 000010      RTRP:    MOV      #12,@#10 ;RESTORE TRAP CATCHER
2869 012406 105737 002176              TSTB     MODE           ;CHECK MODE FOR CORRECT EXIT.
2870 012412 001402              BEQ      70$
2871 012414 000137 002302              JMP      MANSA          ;EXIT TO MANUAL SELECT
2872 012420 000137 002246              70$:    JMP      AUTOA          ;EXIT TO AUTO MODE.
2873
2874      ;:*****
2875      ;SUBROUTINE TO VERIFY ADDRESS AS THEY ARE ENTERED IN
2876      ;MANUAL SELECT MODE.
2877      ;:*****
2878
2879 012424 020227 160000      TMNAD:   CMP      R2,#160000 ;INSURE ADDRESS IS IN RANGE.
2880 012430 103407              BLO      BDEXT          ;ITS NOT, TYPE A ? AND EXIT.
2881 012432 012737 012446 000004      MOV      #BDEXTA,@#4    ;SET UP TIME OUT TRAP.
2882 012440 005712              TST      @R2           ;CHECK THE ADDRESS.
2883 012442 000240              NOP
2884 012444 000405              BR        ALEXT        ;ITS OK, USE A NORMAL EXIT.
2885 012446 022626              BDEXTA: POP2SP          ;ADDRESS TRAPPED- PURGE IT, TYPE A
2886 012450 104401 027252      BDEXT:   TYPE      ,QMRK    ;QUESTION MARK, RESTORE THE TRAP
2887 012454 012700 002316      MOV      #BLDADD,R0     ;LOCATION, AND EXIT TO GET NEXT ENTRY.
2888 012460 012737 000006 000004      ALEXT:   MOV      #6,@#4
2889 012466 000200              RTS      R0
2890
2891      ;:*****
2892      ;THIS ROUTINE MAPS ALL POSSIBLE DL11 ADDRESSES AND STORES
2893      ;THEM IN A TABLE (INTAB). ALL ADDRESSES WHICH DO NOT
2894      ;RESULT IN TIMEOUTS ARE STORED.
2895      ;:*****
2896
2897 012470 012701 000300      TRPVEC:  MOV      #300,R1    ;START AT BEG. OF FLOATING VECTORS
2898 012474 012702 000302              MOV      #302,R2
2899 012500 012703 000004              MOV      #4,R3          ;R3 CONTAINS IOT TRAP INST.
2900 012504 010221              1$:    MOV      R2,(R1)+    ;START LOADING ADDRESSES
2901 012506 010321              MOV      R3,(R1)+    ;LOAD THE TRAP
2902 012510 062702 000004              ADD      #4,R2        ;ASSUME 4 REGISTERS PER INTERFACE
2903 012514 020127 001000              CMP      R1,#1000    ;DONE?
2904 012520 002771              BLT      1$           ;NO CONTINUE LOADING TRAPS
2905 012522 005037 000006              CLR      @#6
2906 012526 012737 012566 000004              MOV      #TPENT,@#4    ;SET UP TIME-OUT TRAP ADDRESS
2907 012534 005001              CLR      R1          ;CLEAR THE TABLE POINTER
2908 012536 012705 001652              MOV      #INTAB,R5    ;R5=DESTINATION TABLE
2909 012542 016102 001716      FADD:    MOV      STRTAB(R1),R2 ;PUT THE ADDRESS TO BE TESTED IN R2
2910 012546 026102 001724      TRPE:   CMP      ENDTAB(R1),R2 ;HAVE WE EXCEEDED END OF TABLE ADDRESS?
2911 012552 103407              BLO      TBLCK         ;YES GET NEXT BASE ADDRESS.
2912 012554 005712              TST      @R2          ;ADDRESS THE DEVICE IF POSSIBLE
2913 012556 010225              MOV      R2,(R5)+    ;IF WE GOT THIS FAR THERE IS A DEVICE THERE-SAVE IT
2914 012560 062702 000010      FADD1:  ADD      #10,R2  ;INCREMENT TO THE NEXT POSSIBLE ADDRESS

```

```

2915 012564 000770
2916 012566 022626
2917 012570 000773
2918 012572 005721
2919 012574 032701 000004
2920 012600 001760
2921 012602 005015
2922 012604 000200
2923
2924
2925
2926
2927
2928 012606 005046
2929 012610 012746 012616
2930 012614 000002
2931 012616 012737 000004 000004
2932 012624 012737 012722 000020
2933 012632 012737 000340 000022
2934 012640 000005
2935 012642 012703 001552
2936 012646 012702 001652
2937 012652 012705 001612
2938 012656 012701 001732
2939 012662 005712
2940 012664 001506
2941 012666 100403
2942
2943 012670 062702 000002
2944 012674 000770
2945 012676 004037 013404
2946 012702 012701 001200
2947 012706 052777 000100 167022
2948 012714 005301
2949 012716 001376
2950 012720 000756
2951 012722 042777 000100 167006
2952 012730 162716 000010
2953 012734 012613
2954 012736 005726
2955 012740 022626
2956
2957
2958
2959
2960 012742 005046
2961 012744 012746 012752
2962 012750 000002
2963 012752 012301
2964 012754 012721 014214
2965 012760 012721 000340
2966 012764 012721 014276
2967 012770 012711 000340
2968 012774 012704 000001
2969 013000 005001
2970 013002 052777 000100 166722

TPENT: BR TRPE ;GO TEST THE NEXT ADDRESS
        POP2SP ;RESTORE THE STACK AND TEST
        BR FADD1 ;NEXT ADDRESS
TBLCK: TST (R1)+ ;BUMP AREA COUNTER BY 2.
        BIT #BIT02,R1 ;SEE IF BOTH DL11 AREAS CHECKED.
        BEQ FADD ;NO-GO CHECK THE OTHER AREA
        CLR (R5) ;SET UP TABLE TERMINATOR OF ZEROS.
        RTS R0

;*****
;THIS ROUTINE WILL INSURE THAT THE DEVICE(DL11)
;WILL INTERRUPT WHEN XMIT INT. ENABLE BIT IS SET.
;*****

CDEV: CLR -(SP) ;CLEAR THE PSW,LSI11 STYLE.
        MOV #100$,-(SP)
        RTI
100$: MOV #4,@#4 ;INSTALL IOT TRAP INST. AT LOCATION 4.
        MOV #TDEV,@#IOTVEC ;SET UP IOT TRAP EXIT ADDRESS
        MOV #340,@#IOTVEC+2 ;SET PSW TO 7-ALLOW NO OTHER INTERRUPTS
        RESET ;INSURE ALL XMIT FLAGS HIGH.
        MOV #VVECT,R3 ;VECTOR STORAGE ADDRESS SET
        MOV #INTAB,R2 ;PRIMARY DEVICE TABLE ADDRESS SET
        MOV #DLTBL,R5 ;FIN DEVICE TABLE ADDRESS SET.
CDEVA: MOV #VRCSR,R1 ;VT61 DEVICE ADDRESS SET.
        TST (R2) ;CHECKED ALL DEVICES?
        BEQ AOUT ;YES-EXIT
        BMI 1$ ;INSURE ADDRESS IS IN PROPER RANGE(17xxxx)

        ADD #2,R2 ;ADDRESS IS DEFINITELY NOT GOOD -PURGE
        BR CDEVA ;AND LOOK FOR ANOTHER.
1$: JSR R0,LDADD ;LOAD NEXT ADDRESSES TO BE CHECKED
        MOV #1200,R1 ;NOW USE R1 AS FAILSAFE COUNTER
        BIS #TENA,@VXCSR ;SET XMIT ENABLE
        DEC R1 ;IF DEVICE DOES NOT INTERRUPT WITHIN
        BNE -2 ;APPROX. 200US IT IS NOT A DL11.
        BR CDEVA ;THEREFORE, GO TRY ANOTHER DEVICE.
TDEV: BIC #TENA,@VXCSR ;CLEAR XMIT ENABLE.
        SUB #10,(R6) ;RESET TO RECEIVER VECTOR ADDRESS
        MOV (R6)+,(R3) ;STORE IT IN VECTOR TABLE(VVECT).
        TST (R6)+ ;POP THE OLD PSW AND DISCARD
        POP2SP ;POP THE ADD. AND PSW PRIOR TO INTERRUPT.

;*****
;THIS ROUTINE IS A QUICK TEST OF ANY DL11 ENCOUNTERED
;A DATA PATTERN WILL BE RUN ON ALL ENTRIES IN INTAB
;*****

CLR -(SP) ;CLEAR THE PSW,LSI11 STYLE.
MOV #100$,-(SP)
RTI
100$: MOV (R3)+,R1 ;GET THE RECEIVE VECTOR ADDRESS
        MOV #RECAD,(R1)+ ;AND STORE SAME.
        MOV #340,(R1)+ ;SET RECEIVE PSW TO 7.
        MOV #TSMAD,(R1)+ ;STORE THE XMIT VECTOR ADDRESS
        MOV #340,(R1) ;SET XMIT PSW TO 7.
        MOV #BIT00,R4 ;R4 IS NOW DATA PATTERN OF 1.
        CLR R1 ;SET UP FAILSAFE DELAY.
        BIS #RDENA,@VRCSR ;SET RECEIVE ENABLE.
  
```

```

2971 013010 052777 000104 166720 1$: BIS #TCOMB,@VXCSR ;ENABLE XMIT INT. AND MAINTENACE .
2972 013016 105704 1$: TSTB R4 ;XMIT PATTERN COMPLETE?
2973 013020 001423 BEQ GDAD ;YES GO STORE THIS ADDRESS
2974 013022 005301 DEC R1 ;CYCLE TIMEOUT DELAY
2975 013024 001374 BNE 1$ ;NOT YET 'TIMEOUT',KEEP CYCLING.
2976 013026 162703 000002 SUB #2,R3 ;RESET VECTOR POINTER
2977 013032 042777 000104 166676 BIC #TCOMB,@VXCSR ;CLEAR XMIT AND RECEIVE INT. ENABLES.
2978 013040 042777 000100 166664 BIC #RDENA,@VRCSR
2979 013046 104401 023461 TYPE ,DLERR ;ISSUE DL11 FAILURE MESSAGE.
2980 013052 013746 001732 MOV VRCSR,-(SP) ;SAVE VRCSR FOR TYPEOUT
2981 ;TYPE BD. ADDRESS
2982 013056 104403 TYPOS ;GO TYPE--OCTAL ASCII
2983 013060 006 .BYTE 6 ;TYPE 6 DIGIT(S)
2984 013061 001 .BYTE 1 ;TYPE LEADING ZEROS
2985 013062 104401 001171 TYPE ,SCRLF
2986 013066 000673 BR CDEVA ;GO TRY ANOTHER SET OF ADDRESSES.
2987 013070 013725 001732 GDAD: MOV VRCSR,(R5)+ ;SAVE GOOD ADDRESS IN DL TABLE
2988 013074 005077 166634 CLR @VRBUF ;CLEAR ANY RECEIVE FLAG STILL SET.
2989 013100 000666 BR CDEVA ;CHECK ANOTHER DL11
2990 013102 005015 AOUT: CLR (R5) ;SET A ZERO TABLE TERMINATOR.
2991 013104 012737 000006 000004 MOV #6,@#4 ;RESTORE LOCATION 4 TO HALT CONDITION
2992 013112 005037 000006 CLR @#6 ;TO CATCH ERRORS AND ILLEGAL INTERRUPTS.
2993 013116 012737 021226 000020 MOV #SCOPE,@#IOTVEC ;RELOAD IOT VECTOR FOR SCOPE
2994 013124 012737 000340 000022 MOV #340,@#IOTVEC+2 ;LOOP.
2995 013132 012701 000300 MOV #300,R1
2996 013136 012702 000302 MOV #302,R2
2997 013142 010221 1$: MOV R2,(R1)+
2998 013144 005021 CLR (R1)+ ;RESTORE HALTS TO ALL LOCATIONS CONTAINING IOTS
2999 013146 062702 000004 ADD #4,R2 ;TO LOCATION 1000
3000 013152 020127 001000 CMP R1,#'000
3001 013156 103771 BLO 1$
3002 013160 000005 RESET ;CLEAR ALL FLAGS
3003 013162 000200 RTS R0
3004
3005 ;*****
3006 ;INITIALIZATION ROUTINE FOR AUTO SELECTION. THIS ROUTINE
3007 ;WILL INSURE THAT ALL DL11S IN DL1BL HAVE A VT61 CONNECTED
3008 ;ALL UNITS WHICH CANNOT CORRECTLY RESPOND WILL BE PURGED.
3009 ;*****
3010
3011 013164 012702 001612 INITA: MOV #DL1BL,R2 ;R2 POINTS TO DL11 ADDRESS TABLE
3012 013170 012703 001552 MOV #VVECT,R3 ;R3 POINTS TO DL11 VECTOR
3013 013174 012701 001732 11$: MOV #VRCSR,P1 ;POINTER TO VT61 DL11
3014 013200 005712 TST (R2) ;SEE IF ALL CHECKED
3015 013202 001447 BEQ INTXT ;YES-EXIT
3016 013204 004037 013404 JSR R0,LDADD ;NO-GO LOAD THE ADDRESSES
3017 013210 062703 000002 ADD #2,R3 ;UPDATE VECTOR COUNT
3018 013214 004037 015674 JSR R0,ZFLAG ;ISSUE ESCZ AND LOOK FOR RESPONSE.
3019 013220 2$:
3020 013220 012637 002162 MOV (SP)+,CHRD ;POP STACK INTO CHRD
3021 013224 100414 BMI 5$ ;TIMEOUT OCCURRED NO CHARACTER

```

```

3022 013226 123727 002162 000140      CMPB   CHRDL,#140      ;CHECK IDENT FOR VT61 IDENTIFIERS
3023 013234 103410      BLO     5$              ;NOT A VT61-SET UP TO PURGE ADDRESS
3024 013236 123727 002162 000172      CMPB   CHRDL,#172      ;IDENTS ARE SMALL A THRU Z
3025 013244 101004      BHI     5$              ;NOT A VT61-PURGE
3026 013246
3027 013246 012637 002162      4$:    MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
3028 013252 001375      BNE     4$              ;
3029 013254 000747      BR      11$             ;TEST ANOTHER ADDRESS
3030 013256
3031 013256 012637 002162      5$:    MOV     (SP)+,CHRD      ;;POP STACK INTO CHRD
3032 013262 001375      BNE     5$              ;
3033 013264 162702 000002      SUB     #2,R2          ;RESET ADDRESS AND VECTOR POINTERS
3034 013270 162703 000002      SUB     #2,R3
3035 013274 010246      MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3036 013276 012746 000001      MOV     #1,-(SP)        ;;PUSH #1 ON STACK
3037 013302 004037 013614      JSR     R0,BBLUP
3038 013306 010346      MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3039 013310 012746 000001      MOV     #1,-(SP)        ;;PUSH #1 ON STACK
3040 013314 004037 013614      JSR     R0,BBLUP
3041 013320 000725      BR      11$             ;TRY ANOTHER DL11 ADDRESS.
3042
3043 013322 005737 001612      INTXT:  TST     DLTBL          ;CHECK TO INSURE GOOD ADDRESSES
3044 013326 001012      BNE     EXINT          ;YES-GO TO NEXT TEST
3045 013330 104401 023370      TYPE    ,NOVT          ;NO-ISSUE NO VT61 MESSAGE.
3046 013334 012737 005670 017502      MOV     #3000.,DCOUNT    ;SET DELAY TO 30 SEC.
3047 013342 004037 017440      JSR     R0,DELAY          ;AND DO IT.
3048 013346 062700 000004      ADD     #4,R0          ;SET UP 'NO VT61 FOUND' EX11
3049 013352 000200      RTS     R0
3050 013354 012702 001612      EXINT:  MOV     #DLTBL,R2      ;LOAD AND ISSUE GOOD ADDRESSES
3051 013360 104401 023317      TYPE    ,DUNTST        ;OF RESPONSIVE VT61S.
3052 013364
3053 013364 012246      1$:    MOV     (R2)+,-(SP)      ;;SAVE (R2)+ FOR TYPEOUT
3054                                ;;TYPE AN ADDRESS
3055                                ;;GO TYPE--OCTAL ASCII
3056 013366 104403      TYPOS    .BYTE     6          ;;TYPE 6 DIGIT(S)
3057 013370      .BYTE     1          ;;TYPE LEADING ZEROS
3058 013371 001      .BYTE     1
3059 013372 104401 001171      TYPE    ,$CRLF
3060 013376 005712      TST     (R2)          ;AT END OF GOOD UNITS?
3061 013400 001371      BNE     1$              ;NO PRINT ANOTHER ADDRESS.
3062 013402 000200      RTS     R0
3063
3064      ;;*****
3065      ;SUBROUTINE TO LOAD 4 ADRESSES FROM THE LOCATION AT (R2).
3066      ;TO 4 LOCATION POINTED TO BY R1(TO VXBUFF+2).ROUTINE USES R4 AS
3067      ;WORK REG AND EXITSD WITH R2 INCREMENTED BY 2.
3068      ;;*****
3069 013404 012204      LDADD:  MOV     (R2)+,R4          ;LOAD THE ADDRESS
3070 013406 010421 000002      1$:    MOV     R4,(R1)+        ;STORE AN ADDRESS
3071 013410 062704 001742      ADD     #2,R4          ;INCREMENT ADDRESS
3072 013414 020127      CMP     R1,#VXBUF+2      ;LOADED 4?
3073 013420 002772      BLT     1$              ;NO LOAD ANOTHER
3074 013422 000200      RTS     R0          ;YES-EXIT
3075
3076      ;;*****
3077      ;ROUTINE TO RECEIVE CHARACTER(S). ENTERED WITH
      ;NUMBER OF CHARACTERS TO RECEIVE ON THE STACK.

```

```

3078 ;ROUTINE EXITS WITH CHARACTER(S) ON STACK. IF A
3079 ;PROGRAM TIME-OUT (100 M.S.) OCCURS BEFORE A CHARACTER
3080 ;IS RECEIVED ROUTINE EXITS WITH -1 ON STACK. FORMAT
3081 ;FOR DATA IS (BYTE2, BYTE1) ETC. A WORD OF ZEROS TERMINATES
3082 ;DATA STRING ON THE STACK. SOM/EOM, IF SENT, ARE RECEIVED
3083 ;BUT NOT STORED.
3084
3085 ;*****
3086
3087 RECTM:
3088 013424 012637 002222 MOV (SP)+,ROSVE ;;POP STACK INTO ROSVE
3089 013430 012637 002210 MOV (SP)+,@#BUBCT ;;POP STACK INTO @#BUBCT
3090 013434 013746 002170 MOV ZERO,-(SP) ;;PUSH ZERO ON STACK
3091 013440 005037 002162 1$: CLR CHRDR ;;CLEAR CHARACTER STORAGE LOCATION.
3092 013444 005037 002220 CLR DLAY ;;SET UP FAILSAFE DELAY
3093 013450 032777 000200 166254 3$: BIT #RECDN,@VRCR ;;SEE IF DONE FLAG SET
3094 013456 001007 BNE 4$
3095 013460 005337 002220 DEC DLAY ;;DECREMENT FAILSAFE CNTR.
3096 013464 001371 BNE 3$ ;;NOT AT ZERO-CONTINUE WAITING.
3097 013466 012737 177777 002162 31$: MOV #-1,CHRDR ;;SET UP FOR FAILSAFE EXIT.
3098 013474 000442 BR RECEX ;;EXIT ROUTINE.
3099 013476 117737 166232 002162 4$: MOV @VRBUF,CHRDR ;;STORE THIS CHARACTER.
3100 013504 042737 000200 002162 BIC #200,CHRDR ;;STRIP PARITY BIT.
3101 013512 122737 000057 002162 CMPB #SLSH,CHRDR ;;RECEIVED A IDENT SLASH(57)?
3102 013520 001007 BNE 41$ ;;NO-STORE A CHARACTER.
3103 013522 105337 002211 DECB BUBCT+1 ;;DECREMENT ALLOWABLE SLASH COUNT.
3104 013526 001757 BEQ 31$ ;;COUNT EQUAL ZERO-SET UP ERROR EXIT.
3105 013530 123727 002211 000213 CMPB BUBCT+1,#139. ;;RECEIVED FIRST SLASH?
3106 013536 103740 BLO 1$ ;;YES-IGNORE THIS ONE.
3107 013540 122737 000002 002162 41$: CMPB #SOM,CHRDR ;;IS CHAR. ACTUALLY SOM?
3108 013546 001003 BNE 5$ ;;NO
3109 013550 105237 002210 INCB BUBCT ;;YES -SET UP TO RECEIVE EOM ALSO
3110 013554 000731 BR 1$ ;;AND RECEIVE NEXT CHAR.
3111 013556 122737 000004 002162 5$: CMPB #EOM,CHRDR ;;CHAR. = EOM?
3112 013564 001410 BEQ RECEXA ;;YES- DO NOT PUSH IT ON STACK
3113 013566 105337 002210 DECB BUBCT ;;DECREMENT CHARACTER COUNT.
3114 013572 001403 BEQ RECEX ;;COUNT=0. EXIT WERE DONE.
3115 013574 013746 002162 MOV CHRDR,-(SP) ;;PUSH CHRDR ON STACK
3116 013600 000717 BR 1$ ;;GO READ AGAIN.
3117
3118 RECEX:
3119 013602 013746 002162 MOV CHRDR,-(SP) ;;PUSH CHRDR ON STACK
3120 013606 RECEXA:
3121 013606 013746 002222 MOV ROSVE,-(SP) ;;PUSH ROSVE ON STACK
3122 013612 000200 RTS PO
3123
3124 ;*****
3125 ;THIS ROUTINE WILL 'BUBBLE UP' XX WORDS TO
3126 ;ELIMINATE NON-RESPONSIVE ADDRESSES. ENTERED
3127 ;WITH ADDRESS TO BE 'BUBBLED' TO ON THE STACK. LOCATIONS
3128 ;ELIMINATED WILL BE FILLED WITH ZEROS. THE STACK MUST ALSO
3129 ;BE LOADED WITH THE NUMBER OF POSITIONS TO BUBBLE.
3130 ;*****
3131
3132 BBLUP:
3133 013614

```

```

3134 013614 012637 002222      MOV      (SP)+,ROSVE      ;;POP STACK INTO ROSVE
3135 013620 012637 002210      MOV      (SP)+,BUBCT      ;;POP STACK INTO BUBCT
3136 013624 012637 002206      MOV      (SP)+,TOADD      ;;POP STACK INTO TOADD
3137 013630 010446              MOV      R4,-(SP)      ;;PUSH R4 ON STACK
3138 013632 013704 002206      2$:      MOV      @#TOADD,R4      ;;PUT LAST GOOD DL11 ADDRESS IN R4
3139 013636 012437 002162      MOV      (R4)+,CHRD      ;;MOVE NEXT WORD TO CHRD FOR STORAGE
3140 013642 012464 177774      1$:      MOV      (R4)+,-4(R4)      ;;BUBBLE UP DATA.
3141 013646 001375              BNE      1$      ;;BUBBLE UNTIL ZERO BYTE MOVED.
3142 013650 005337 002210      DEC      BUBCT      ;;SUBTRACT ONE FROM BUBBLE COUNT.
3143 013654 001366              BNE      2$      ;;IF BUBBLE COUNT NOT ZERO - DO AGAIN.
3144 013656              3$:
3145 013656 012604              MCV      (SP)+,R4      ;;POP STACK INTO R4
3146 013660 013746 002222      MOV      ROSVE,-(SP)      ;;PUSH ROSVE ON STACK
3147 013664 000200              RTS      R0      ;;YES-EXIT

```

```

;;*****
;;THIS ROUTINE OUTPUTS THE ESC SEQUENCE FOUND ON
;;THE STACK. A WORD OF ZEROS MUST TERMINATE THE SEQUENCE.
;;FORMAT FOR STACK WORD IS SEQ-ESC, IE-XXX033.
;;*****

```

```

3155 013666              TESC:
3156 013666 012637 002222      MOV      (SP)+,ROSVE      ;;POP STACK INTO ROSVE
3157 013672 010437 002210      MOV      R4,BUBCT      ;;SAVE R4.
3158 013676 112777 000002 166034      MOV      #SOM,@VXBUF      ;;SEND A START OF MESSAGE.
3159 013704 012705 177777      1$:      MOV      #-1,R5      ;;ALL ONES THE CHECK LOCATION.
3160 013710 012604              MOV      (R6)+,R4      ;;GET COMMAND FROM STACK.
3161 013712 001415              BEQ      3$      ;;IF ZERO TERMINATOR FOUND-EXIT.
3162 013714 110405              MOV      R4,R5      ;;LOAD CHECK BYTE.
3163 013716 105704              2$:      TSTB      R4      ;;CHECK BYTE FOR A ZERO.
3164 013720 001406              BEQ      20$      ;;IF ZERO_DO NOT XMIT IT.
3165 013722 032777 000200 166006      BIT      #TRDY,@VXCSR
3166 013730 001774              BEQ      ,-6      ;;WAIT FOR XMIT READY BIT
3167 013732 110477 166002      MOV      R4,@VXBIF      ;;XMIT A BYTE.
3168 013736 000304              20$:      SWAB      R4      ;;GET THE OTHER BYTE.
3169 013740 120405              CMPB      R4,R5      ;;IF GOOD COMPARE WE HAVE CHECKED BOTH
3170 013742 001760              BEQ      1$      ;;BYTES SO POP ANOTHER WORD.
3171 013744 000764              BR      2$      ;;GO XMIT ANOTHER BYTE
3172 013746 032777 000200 165762      3$:      BIT      #TRDY,@VXCSR      ;;SEE IF READY SET
3173 013754 001774              BEQ      ,-6
3174 013756 012777 000004 165754      MOV      #EOM,@VXBUF      ;;SEND A EOM.
3175 013764 032777 000200 165744      BIT      #TRDY,@VXCSR      ;;SEE IF READY SET
3176 013772 001774              BEQ      ,-6
3177 013774 013704 002210      MOV      BUBCT,R4      ;;RESTORE R4.
3178 014000 013746 002222      MOV      ROSVE,-(SP)      ;;PUSH ROSVE ON STACK
3179 014004 000200              RTS      R0

```

```

;;*****
;;ROUTINE TO READ A CHARACTER FROM THE CONSOLE.
;;EXITS WITH CHARACTER ON THE STACK.
;;*****

```

```

3186 014006              (CONRD:
3187 014006 012637 002222      MOV      (SP)+,ROSVE      ;;POP STACK INTO ROSVE
3188 014012 032777 000200 165124      BIT      #RECDN,@$TKS      ;;LOOK FOR DONE BIT
3189 014020 001774              BEQ      ,-6      ;;WAIT FOR IT

```

```

3190 014022 117746 165120      MOVB  @STKB,-(R6)      ;PUSH CHARACTER TO STACK
3191 014026 042716 000200      BIC   #200,(R6)      ;STRIP ANY PARITY BIT.
3192 014032 013746 002222      MOV   ROSVE,-(SP)    ;:PUSH ROSVE ON STACK
3193 014036 000200              RTS   R0
3194
3195      ;:*****
3196      ;MANUAL TEST SELECT MONITOR
3197      ;SELECTS TESTS TO BE EXECUTED FROM THOSE ENTERED IN
3198      ;INITIAL DIALOGUE. IF TEST 377 WAS REQUESTED THE TESTS WILL
3199      ;REPEAT INFINITELY.
3200      ;:*****
3201
3202 014040 105737 002176      MONIT: TSTB  MODE      ;TEST MODE SWITCH
3203 014044 001012              BNE   1$      ;MANUAL MODE
3204 014046 023737 002200 002202      CMP   FTLCNT,ALWCNT  ;COMPARE FATAL XMTS WITH ALLOWED.
3205 014054 103405              BLO   100$     ;FATALS LESS THAN ALLOWED-CONTINUE.
3206 014056 104401 025123      TYPE  ,DABRT   ;ISSUE ABORT MESSAGE.
3207 014062 000005      200$:  RESET      ;CLEAR ALL INTERFACE FLAGS.
3208 014064 000137 012012      JMP   SETA     ;SET UP TO RESTART TEST.
3209 014070 000200      100$:  RTS   R0      ;AUTO MODE
3210 014072 005726      1$:   TST   (R6)+    ;POP THE STACK
3211 014074 022626              POP2SP      ;POP SCOPE RETURN AND VECTOR
3212 014076 005037 002200      CLR   FTLCNT  ;DO NOT INC. FATAL COUNT IN MANUAL MODE.
3213 014102 032777 000200 165034 10$:  BIT   #RECDN,@STKS ;CONSOLE ACTIVE?
3214 014110 001407              BEQ   11$
3215 014112 117701 165030      MOVB  @STKB,R1 ;STORE INPUT BUFFER
3216 014116 042701 000200      BIC   #200,R1 ;CLEAR THE PARITY BIT
3217 014122 122701 000003      CMPB  #3,R1  ;CHAR. EQUAL ESC. C?
3218 014126 001755              BEQ   200$    ;YES-EXIT
3219 014130 117701 166040      11$:  MOVB  @TSTPTR,R1 ;GET THE NEXT TEST #
3220 014134 003005              BGT   2$      ;NOT AT END OF LIST
3221 014136 042777 000100 165566      BIC   #RDENA,@VRCR  ;CLEAR REC. INTERRUPTS BEFORE NEXT UNIT SELECT.
3222 014144 000137 002534              JMP   MODCA   ;END OF LIST-GO SET UP NEXT 61
3223 014150 005301      2$:   DEC   R1      ;ADJUST OFFSET
3224 014152 006301              ASL   R1      ;USE TEST # TO FORM ADDRESS OFFSET
3225 014154 016137 023144 014212      MOV   TSTADD(R1),JMPADD+2 ;LOAD NEW ADDRESS
3226 014162 062737 000002 014212      ADD   #2,JMPADD+2 ;BYPASS INITIAL SCOPE LOOP
3227 014170 005237 002174              INC   TSTPTR ;INCREMENT TEST OPINTER
3228 014174 005037 177776              CLR   PSW   ;SET NON-INT. PRIORITY TO ZERO
3229 014200 005046              CLR   -(SP)  ;CLEAR THE PSW,LSI 11 STYLE.
3230 014202 012746 014210      MOV   #JMPADD,-(SP)
3231 014206 000002              RTI
3232 014210 000137 014210      JMPADD: JMP   JMPADD ;EXIT TO NEXT SELECTED TEST
3233      ;:*****
3234      ;:*****
3235      ;FOLLOWING ROUTINES ARE INTERRUPT HANDLERS FOR THE
3236      ;DL11 QUICK-TEST.
3237      ;:*****
3238      ;:*****
3239 014214 117737 165514 002162      RECAD: MOVB  @VRBUF,CHRD ;GET THE RECEIVED CHAR.
3240 014222 042737 000200 002162      BIC   #200,CHRD ;CLEAR ANY PARITY.
3241 014230 120437 002162      CMPB  R4,CHRD ;COMPARE RECEIVED TO XMITTED
3242 014234 001407              BEQ   UPD4    ;AND UPDATE PATTERN IF OK.
3243 014236 042777 000104 165472      TOFF: BIC   #TCOMB,@VXCSR ;DATA ERROR OCCURED OR WE ARE DONE
3244 014244 042777 000100 165460      BIC   #RDENA,@VRCR ;EITHER WAY-EXIT.
3245 014252 000002              REEX:  RTI

```

```

3246 014254 052777 000100 165454 UPD4: BIS #TENA,@VXCSR ;ENABLE XMIT INT.
3247 014262 106304 ASLB R4 ;UPDATE DATA PATTERN.
3248 014264 032704 000200 BIT #BIT07,R4 ;ROTATED TO PARITY BIT?
3249 014270 001770 BEQ REEX ;NO-CONTINUE TESTING
3250 014272 005004 CLR R4 ;YES-SET UP COMPLETE FLAG
3251 014274 000760 BR TOFF ;AND EXIT.
3252 014276 110477 165436 TSMAD: MOVB R4,@VXBUF ;XMIT DATA
3253 014302 042777 000100 165426 BIC #TENA,@VXCSR ;CLEAR XMIT INT. UNTIL LAST BIT REC.
3254 014310 000002 RTI
3255
3256 ;*****
3257 ;RECEIVE INTERRUPT ROUTINE.AFTER EACH RECEIVE
3258 ;CYCLE BUFFER POINTER (RBUF) WILL BE SET TO (RBBUF).
3259 ;MAX. EXECUTION TIME IS APPROX 200US, AVERAGE =100US.
3260 ;UPON RECEIPT OF XON, XMTKIL BIT IS CHECKED IN VSTAT
3261 ;AND IF SET, WILL BE CLEARED AND XMIT INT. ENABLE SET.
3262 ;LOCATION ESAMB IS USED FOR ESC ASSEMBLY FLAGS. IE. BIT
3263 ;00 SET MEANS A033 WAS RECEIVED, BIT 01 SET MEANS AN ESCP
3264 ;SEQUENCE IS BEING ASSEMBLED. BIT 03
3265 ;SET INDICATES AND ESCAPE 0 SEQUENCE IS BEING ASSEMBLED.
3266 ;LOCATIONS STRP AND STRP ARE USED TO STORE ESCAPE
3267 ;0 AND ESCAPE P SEQUENCES DETECTED,BUT NOT UTILIZED IN TEST.
3268 ;*****
3269
3270 INTRC:
3271 014312 010146 MOV R1,-(SP) ;:PUSH R1 ON STACK
3272 014314 017701 165414 MOV @VRBUF,R1 ;USE R1 FOR STORAGE OF STATUS AND CH.
3273 014320 042701 000200 BIC #200,R1 ;STRIP PARITY BIT.
3274 014324 032737 000100 002224 BIT #TXSUM,VSTAT ;CHECKSUM CALCULATION REQUESTED?
3275 014332 001403 BEQ 11$ ;NO
3276 014334 010105 MOV R1,R5 ;YES-STORE CHAR. AND
3277 014336 004037 020124 JSR R0,CALCK ;CALCULATE THE CHECKSUM.
3278 014342 005237 031602 11$: INC ABUFF ;INCREMENT THE RAW DATA POINTER
3279 014346 023727 031602 031666 CMP ABUFF,#ABBIIF+50. ;AT THE END OF BUFFER?
3280 014354 001003 BNE 12$ ;NO
3281 014356 012737 031604 031602 MOV #ABBUF,ABUFF ;YES-RESET IT
3282 014364 110177 015212 12$: MOVB R1,@ABUFF ;STORE THE RAW DATA
3283 014370 001505 BEQ 6$ ;IF CHAR. IS NULL-GO STORE IT
3284 014372 032737 000013 015202 BIT #BIT00+BIT01+BIT03,ESAMB ;ESC OR ESC 0?
3285 014400 001150 BNE ADESC ;YES-KEEP ASSEMBLING
3286 014402 120137 002132 CMPB R1,ESCN ? ;BYTE = ESCN?
3287 014406 101076 BHI 6$ ;NO-PROBABLY A DISPLAY CH.-STORE IT.
3288 014410 001007 BNE 1$ ;NO-DECODE FOR XON,XOFF,SOM,EOM
3289 014412 012737 000001 015202 MOV #1,ESAMB ;YES SET ESC ASSEMBLY FLAG.
3290 014420 052737 000400 002224 BIS #ESC,VSTAT ;SET ESC RECEIVED FLAG
3291 014426 000515 BR RSTER ;AND EXIT
3292 014430 120127 000023 1$: CMPB R1,#XOFF ;SEE IF RECEIVED BYTE WAS XOFF
3293 014434 001004 BNE 2$ ;NO
3294 014436 052737 100000 002224 BIS #RXOFF,VSTAT ;YES, SET XOFF IN STATUS REG.
3295 014444 000506 BR RSTER ;EXIT
3296 014446 120127 000021 2$: CMPB R1,#XON ;SEE IF BYTE WAS XON
3297 014452 001016 BNE 3$ ;NO
3298 014454 042737 100000 002224 BIC #RXOFF,VSTAT ;YES, CLEAR XOFF IN VSTAT.
3299 014462 032737 000200 002224 BIT #XMTKIL,VSTAT ;CHECK XMIT KILL BIT.
3300 014470 001474 BEQ RSTER ;NOT SET, EXIT
3301 014472 052777 000100 165236 BIS #TENA,@VXCSR ;SET XMIT INT. ENABLE.

```

```

3302 014500 042737 000200 002224      BIC      #XMKIL,VSTAT      ;CLEAR THE XMIT KILLED FLAG
3303 014506 000465                BR      RSTER                ;EXIT
3304 014510 120127 000002          3$:  CMPB      R1,#SOM          ;SEE IF BYTE WAS SOM
3305 014514 001004                BNE      4$                    ;NO
3306 014516 052737 040000 002224 31$:  BIS      #RSOM,VSTAT      ;YES, SET SOM IN VSTAT.
3307 014524 000456                BR      RSTER                ;EXIT
3308
3309 014526 120127 000004          4$:  CMPB      R1,#EOM          ;WAS BYTE EOM?
3310 014532 001012                BNE      5$                    ;NO
3311 014534 052737 020000 002224      BIS      #REOM,VSTAT      ;NOW SET EOM IN VSTAT.
3312 014542 013737 015174 015200      MOV      RBBUF,RBUF      ;RESET THE BUFFER POINTER.
3313 014550 042737 000100 002224      BIC      #TXSUM,VSTAT      ;CLEAR CHECKSUM REQUEST BIT.
3314 014556 000441                BR      RSTER                ;AND EXIT
3315 014560 123701 001752          5$:  CMPB      CARRT,R1        ;CHAR. =CARRIAGE RETURN?
3316 014564 001403                BEQ      51$                   ;YES-GO SET END OF LINE FLAG
3317 014566 123701 001754          CMPB      LNFED,R1        ;CHAR.= LINEFEED?
3318 014572 001004                BNE      6$                    ;NO- GO STORE IT
3319 014574 052737 001000 002224 51$:  BIS      #EPL,VSTAT      ;SET END OF LINE INDICATOR
3320 014602 000427                BR      RSTER
3321
3322 014604 023737 015200 015176 6$:  CMP      RBUF,RBUF      ;IS CIRCULAR BUFFER FILLED?
3323 014612 001003                BNE      61$                   ;NO
3324 014614 013737 015174 015200      MOV      RBBUF,RBUF      ;YES, RESET POINTER TO BEGINNING
3325 014622 032737 000020 002224 61$:  BIT      #COMGP,VSTAT      ;RECEIVING GRAPHICS CHAR.?
3326 014630 001402                BEQ      7$                    ;NO
3327 014632 162701 000137          SUB      #137,R1            ;YES-SUBTRACT 137 FROM RECEIVED CHAR.
3328
3329 014636 032737 000040 002224 7$:  BIT      #REVID,VSTAT      ;REVERSE VIDEO MODE?
3330 014644 001402                BEQ      70$                   ;NO STORE RECEIVED BYTE.
3331 014646 052701 000200          BIS      #200,R1            ;YES-FORCE BIT7 AS REV. VIDEO IND.
3332 014652 110177 000322          70$:  MOV      R1,RBUF      ;STORE BYTE AND
3333 014656 005237 015200          INC      RBUF      ;INCREMENT POINTER.
3334
3335 014662 005701          RSTER:  TST      R1                    ;CHECK FOR STATUS ERROR
3336 014664 100014          BPL      RECXT                ;NO, EXIT ROUTINE
3337
3338 014666 052737 004000 002224      BIS      #RSTT,VSTAT      ;SET STATUS ERROR FLAG IN VSTAT
3339 014674 027727 000326 177777      CMP      @STTEP,#-1      ;IS ERROR TABLE FULL?
3340 014702 001405                BEQ      RECXT                ;YES, EXIT ROUTINE
3341 014704 010177 000316          MOV      R1,@STTEP          ;NO, STORE STATUS ERR. AND CHECK
3342 014710 062737 000002 015226      ADD      #2,STTEP      ;INCREMENT STATUS ERR. PCINTER
3343
3344 014716          RECXT:  MOV      (SP)+,R1                ;POP STACK INTO R1
3345 014716 012601          RTI                        ;EXIT
3346 014720 000002          ADESC:  BIT      #2,ESAMB          ;ASSEMBLING ESC P?
3347 014722 032737 000002 015202      BNE      AESCP            ;YES-GO GET LAST CH.
3348 014730 001063          BIT      #BIT03,ESAMB          ;ASSEMBLING ESC O?
3349 014732 032737 000010 015202      BNE      AESCO            ;YES
3350 014740 001062          CMPB      #120,R1            ;CH.- A P?
3351 014742 122701 000120          BNE      10$                   ;NO KEEP CHECKING
3352 014746 001004          BIS      #BIT01,ESAMB          ;YES-SET ESCP ASSEMBLY FLAG
3353 014750 052737 000002 015202      BR      RSTER                ;AND EXIT
3354 014756 000741          10$:  CMPB      #77,R1            ;CHAR.IS AN ESC ? ?
3355 014760 122701 000077          BEQ      110$                   ;YES-FAKE AN ESC O.
3356 014764 001403          CMPB      #117,R1           ;CHAR = O?
3357 014766 122701 000117

```

3358	014772	001004			BNE	11\$		;NO
3359	014774	052737	000010	015202	BIS	#BIT03,ESAMB		;YES SET ESC 0 ASSEMBLY FLAG
3360	015002	000727			BR	RSTER		;AND EXIT
3361	015004	123701	002050		11\$:	CMPB	RDCUR,R1	;BYTE= CURSOR POSITION?
3362	015010	001004			BNE	1\$		;NO-
3363	015012	052737	000004	002224	BIS	#CURPOS,VSTAT		;YES-SET RECEIVED CURSOR POSITION.
3364	015020	000424			BR	CESAM		
3365	015022	122701	000057		1\$:	CMPB	#SLSH,R1	;BYTE=TERMINAL ID ESC?
3366	015026	001004			BNE	2\$		;NO-CHECK FOR GRAPHICS SEQUENCE.
3367	015030	052737	000002	002224	BIS	#TRMID,VSTAT		;YES-SET TERM. IDENT FLAG IN VSTAT
3368	015036	000415			BR	CESAM		
3369	015040	122701	000106		2\$:	CMPB	#CKGP,R1	;RECEIVED GRAPHICS CHAR. SEQUENCE?
3370	015044	001004			BNE	3\$		;NO
3371	015046	052737	000020	002224	BIS	#COMGP,VSTAT		;YES-SET GRAPHICS DATA FLAG.
3372	015054	000406			BR	CESAM		
3373	015056	122701	000107		3\$:	CMPB	#NCKGP,R1	;RECEIVED RESET GRAPHICS SEQ.?
3374	015062	001003			BNE	CESAM		;NO
3375	015064	042737	000020	002224	BIC	#COMGP,VSTAT		;YES-SET NORMAL CHAR. RECEIVE.
3376	015072	005037	015202		CESAM:	CLR	ESAMB	;CLEAR ASSEMBLY FLAG.
3377	015076	000671			BR	RSTER		;AND EXIT.
3378								
3379	015100	110137	015232		AESCP:	MOVB	R1,STRP	;STORE ANY UNCHECKED FOR ESC. P
3380	015104	000772			BR	CESAM		
3381								
3382	015106	123701	002016		AESCO:	CMPB	EEMP,R1	;BYTE=ESC 0 -REV. VIDEO- ?
3383	015112	001004			BNE	1\$		;NO
3384	015114	052737	000040	002224	BIS	#REVID,VSTAT		;YES-SET REVERSE VIDEO MODE IN VSTAT.
3385	015122	000763			BR	CESAM		
3386								
3387	015124	123701	002020		1\$:	CMPB	DEMP,R1	;BYTE=ESC 0 DISABLE REV. VIDEO MODE?
3388	015130	001004			BNE	2\$		;NO
3389	015132	042737	000040	002224	BIC	#REVID,VSTAT		;YES-CLEAR REVERSE VIDEO MODE IN VSTAT.
3390	015140	000754			BR	CESAM		
3391	015142	122701	000171		2\$:	CMPB	#CPABRT,R1	;COPIER ABORT?
3392	015146	001403			BEQ	3\$		;YES-SET ABORT FLAG IN VSTAT
3393	015150	122701	000172		CMPB	#PRABRT,R1		;PRINTER ABORT?
3394	015154	001004			BNE	4\$		;NO
3395	015156	052737	010000	002224	3\$:	BIS	#PABRT,VSTAT	;YES-SET THE ABORT FLAG.
3396	015164	000742			BR	CESAM		;AND EXIT.
3397	015166	110137	015230		4\$:	MOVB	R1,STRO	;STORE ESCAPE 0 COMMAND
3398	015172	000737			BR	CESAM		
3399								
3400	015174	000000			RBBUF:	.WORD		;ADDRESS OF STAT OF BUFFER
3401	015176	000000			REBUF:	.WORD		;ADDRESS OF END OF BUFFER.
3402	015200	000000			RBUFP:	.WORD		;READ BUFFER POINTER.
3403	015202	000000			ESAMB:	.WORD	0	;ESCAPE SEQ.ASSEMBLY AREA
3404								
3405	015204	000010			STTER:			
3406	015204	000000				0		
3407	015206	000000				0		
3408	015210	000000				0		
3409	015212	000000				0		
3410	015214	000000				0		
3411	015216	000000				0		
3412	015220	000000				0		
3413	015222	000000				0		

```

3414 015224 177777
3415 015226 000000
3416 015230 000000
3417 015232 000000
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
3430
3431
3432
3433 015234 005737 002224
3434 015240 100004
3435 015242 052737 000200 002224
3436 015250 000510
3437
3438 015252 105737 002227
3439 015256 001406
3440 015260 112777 000002 164452
3441 015266 105037 002227
3442 015272 000002
3443 015274 005737 015510
3444 015300 001006
3445 015302 112777 000004 164430
3446 015310 105037 002226
3447 015314 000452
3448 015316 105777 000164
3449 015322 001016
3450 015324 005737 021074
3451 015330 001023
3452 015332 023737 015506 015504
3453 015340 001004
3454 015342 013737 015502 015506
3455 015350 000740
3456 015352 005237 015506
3457 015356 000735
3458
3459 015360 032737 002000 002224 1$:
3460 015366 001404
3461 015370 117705 000112
3462 015374 004037 020124
3463 015400 117777 000102 164332 22$:
3464 015406 023737 015506 015504
3465 015414 001004
3466 015416 013737 015502 015506
3467 015424 000402
3468 015426 005237 015506
3469

;*****
;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.
;*****
INTXM: TST VSTAT ;HAS 61 TRANSMITTED XOFF?
      BPL NOKIL ;NO XMIT ANOTHER
      BIS #XMKIL,VSTAT ;SET XMIT KILLED BIT IN VSTAT
      BR KIENA ;GO KILL XMIT ENABLE

NOKIL: TSTB BLKM+1 ;SOM/EOM TRANSMIT?
      BEQ NOSOM ;NO
      MOVB #SOM,@VXBUF ;YES-ISSUE START OF MESSAGE.
      CLRB BLKM+1 ;AND CLEAR SOM FLAG.
      R-1

NOSOM: TST XMCNT ;XMITTED THE BUFFER?
      BNE 100$ ;NO-XMIT A NORMAL CHAR.
      MOVB #EOM,@VXBUF ;YES SEND EOM AND EXIT
      CLRB BLKM
      BR 2$

100$: TSTB @TBUF ;CHECK FOR CH.= ZERO. IF SO DO NOT XMIT
      BNE 1$ ;OR COUNT BYTE. OR ARE WE
      TST XMZER ;XMITTING ZEROS?
      BNE 22$ ;YES-XMIT NEXT BYTE
      CMP TBUF,TBUF ;AT END OF BUFFER?
      BNE 10$ ;NO
      MOV TBUF,TBUF ;YES-RESET BUFFER POINTER
      BR NOKIL

10$: INC TBUF
      BR NOKIL ;LOOK FOR NON-ZERO BYTE TO TRANSMIT.

1$: BIT #CKSUM,VSTAT ;CHECKSUM REQUESTED?
      BEQ 22$
      MOVB @TBUF,R5 ;YES,LOAD THE BYTE
      JSR RO,CALCK ;AND CALCULATE THE NEW CHECKSUM.
      MOVB @TBUF,@VXBUF ;TRANSMIT A CHARACTER
      CMP TBUF,TBUF ;AT END OF CIRCULAR BUFFER?
      BNE 11$ ;NO
      MOV TBUF,TBUF ;YES, RESET IT TO START.
      BR 12$ ;BY-PASS INCREMENT BUFF. POINTER
      INC TBUF ;INCREMENT BUFFER POINTER.
  
```

```

3470 015432 005337 015510      12$: DEC      XMCNT      ;DECREMENT THE TRANSMIT COUNT
3471 015436 001401              BEQ      2$          ;YES,CLEANUP,REQUEST ERRORS AND EXIT.
3472 015440 000002              RTI              ;NO, CONTINUE
3473 015442 105737 002226      2$: TSTB      BLKM      ;JOM/EOM XMIT?
3474 015446 001014              BNE      TXEX      ;YES-DO NOT SET XMDNE UNTIL EOM SENT.
3475 015450 052737 000001 002224 BIS      #XMDNE,VSTAT ;SET THE DONE BIT IN VSTAT.
3476 015456 042737 002000 002224 BIC      #CKSUM,VSTAT ;CLEAR THE CHECKSUM FLAG WHEN DONE.
3477 015464 013737 015502 015506 MOV      TBBUF,TBUPF ;RESET BUFFER POINTER.
3478 015472 042777 000100 164236 KIENA: BIC      #TENA,@VXCSR ;CLEAR XMIT. INT. ENABLE
3479 015500 000002              TXEX: RTI
3480
3481
3482 015502 000000              TBBUF: .WORD      ;CONTAINS INITIAL ADDRESS
3483 015504 000000              TEBUF: .WORD      ;CONTAIN LAST ADDRESS
3484 015506 000000              TBUPF: .WORD      ;CONTAINS CURRENT LOCATION
3485
3486 015510 000000              XMCNT: .WORD      0      ;LOADED WITH NUMBER OF XMITS.
3487 ;:*****
3488
3489
3490 ;SUBROUTINE TO ISSUE RESET TO THE VT6', ENTERS MAINTENANCE MODE
3491 ;AND FORCES LINEAR ADDRESSING.
3492 ;:*****
3493
3494 015512 113737 001102 002230 RESETV: MOVB      $TSTNM,TSTNM ;LOAD THE TEST NUMBER IN ERROR PRINT AREA.
3495 015520 013746 002170              MOV      ZERO,-(SP) ;:PUSH ZERO ON STACK
3496 015524 013746 002130              MOV      RESET,-(SP) ;:PUSH RESET ON STACK
3497 015530 013746 002054              MOV      ESCO,-(SP) ;:PUSH ESCO ON STACK
3498 015534 004037 013666              JSR      RO,TESC ;GO XMIT IT
3499 015540 052737 100000 002224 BIS      #RXOFF,VSTAT ;TURN RECEIVER OFF (REV. C0)
3500 015546 004037 015630              JSR      RO,GETON ;GO LOOK FOR XON.
3501 015552 000405              BR      1$          ;FOUND IT.
3502 015554 005237 002200              INC      FTLCNT ;ADD 1 TO FATAL XMIT COUNT.
3503 015560 010037 001120              MOV      RO,$GDADR ;NO XON ISSUE XON ERROR
3504 015564 104017              ERROR 17
3505 015566
3506 015566 013746 002170      1$: MOV      ZERO,-(SP) ;:PUSH ZERO ON STACK
3507 015572 013746 002000              MOV      EMAN,-(SP) ;:PUSH EMAN ON STACK
3508 015576 013746 002054              MOV      ESCO,-(SP) ;:PUSH ESCO ON STACK
3509 015602 013746 002010              MOV      DRECT,-(SP) ;:PUSH DRECT ON STACK
3510 015606 013746 002054              MOV      ESCO,-(SP) ;:PUSH ESCO ON STACK
3511 015612 004037 013666      2$: JSR      RO,TESC
3512 015616 005037 002224              CLR      VSTAT ;CLEAR INT. FLAGS AFTER TERMINAL RESET
3513 015622 005037 017354              CLR      HDPLG ;CLEAR PRINT HEADER FLAG.
3514 015626 000200              RTS      RO
3515
3516 ;:*****
3517 ;SUBROUTINE TO WAIT FOR AN XON. NO XON EXIT IS PC +2.
3518 ;:*****
3519
3520 015630 012737 000454 002210 GETON: MOV      #300.,BUBCT ;SET UP TO LOOK FOR 3 SEC.
3521 015636 032737 100000 002224 1$: BIT      #RXOFF,VSTAT ;RECEIVED XON? (REV. C0)
3522 015644 001412              BEQ      GOTON ;YES-EXIT.
3523 015646 012737 000001 017502 MOV      #1,DOUNT ;NO-DELAY 10 M.S.
3524 015654 004037 017440              JSR      RO,DELAY
3525 015660 005337 002210              DEC      BUBCT ;AT END OF DELAY?

```

3526 015664 001364
 3527 015666 062700 000002
 3528 015672 000200
 3529
 3530
 3531
 3532
 3533
 3534
 3535
 3536
 3537 015674
 3538 015674 012637 015732
 3539 015700 013746 002170
 3540 015704 013746 002126
 3541 015710 004037 013666
 3542 015714 012746 106003
 3543 015720 004037 013424
 3544 015724 013746 015732
 3545 015730 000200
 3546 015732 000000
 3547
 3548
 3549
 3550
 3551
 3552
 3553
 3554 015734
 3555 015734 012637 002222
 3556 015740 010137 002164
 3557 015744 010237 002166
 3558
 3559 015750 012601
 3560 015752 013702 002224
 3561
 3562 015756 042702 003576
 3563 015762 020102
 3564 015764 001432
 3565
 3566 015766 010137 001124
 3567 015772 013737 002224 001126
 3568 016000 104003
 3569
 3570
 3571
 3572
 3573
 3574
 3575
 3576 016002 012701 015204
 3577 016006 013702 015226
 3578 016012 020102
 3579 016014 001416
 3580 016016 004037 016076
 3581 016022 013737 001732 001120

```

      BNE      1$          ;NO
      ADD      #2,R0       ;YES-SET UP ERROR EXIT.
GOTON: RTS      R0

;*****
      ;SUBROUTINE TO ISSUE ESCZ AND LOOK FOR A RESPONSE-EITHER
      ;A -1 OR THE RETURNED IDENT. THE -1 INDICATES NO
      ;RESPONSE FROM THE UNIT UNDER TEST.
;*****

ZFLAG:
      MOV      (SP)+,ROSV1  ;;POP STACK INTO ROSV1
      MOV      @#ZERO,-(SP) ;;PUSH @#ZERO ON STACK
      MOV      @#ESCZ,-(SP) ;;PUSH @#ESCZ ON STACK
      JSR      R0,TESE      ;GO ISSUE ESZ SEQUENCE
      MOV      #106003,-(SP) ;;PUSH #106003 ON STACK
      JSR      R0,RECTM     ;GO READ THE CHARACTER
      MOV      ROSV1,-(SP)  ;;PUSH ROSV1 ON STACK
      RTS      R0
ROSV1: .WORD    0

;*****
      ;ROUTINE TO CHECK SOFTWARE STATUS REGISTER (VSTAT)
      ;RECEIVE FLAGS ONLY. ENTERED WITH ANTICIPATED
      ;TATUS WORD ON THE STACK.
;*****

CKSFT:
      MOV      (SP)+,ROSV1  ;;POP STACK INTO ROSV1
      MOV      R1,SVER1     ;SAVE R1
      MOV      R2,SVER2     ;SAVE R2

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

      BIC      #003576,R2   ;CLEAR NON-ERROR BITS
      CMP      R1,R2        ;COMPARE ANTICIPATED TO ACTUAL.
      BEQ      NOER         ;NO UNUSAL BITS EXIT

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

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

      MOV      #STTER,R1    ;SET R1 EQUAL TO FIRST ENTRY
      MOV      STTER,R2     ;SET R2 EQUAL LAST ENTRY
1$:    CMP      R1,R2        ;ARE THEY EQUAL
      BEQ      NOER         ;YES-RESET POINTERS AND EXIT.
      JSR      R0,CLREG     ;CLEAR ERROR PRINT LOC.
      MOV      VRCSR,$GDADR ;LOAD ADDRESS
  
```

```

3582 016030 017737 163676 001124      MOV    @VRCSR,$GDDAT    ;LOAD CSR
3583 016036 112137 001126      2$:    MOV    (R1)+,$BDDAT    ;MOVE CHARACTER AND
3584 016042 112137 001123      MOV    (R1)+,$BDADR+1  ;STATUS BITS TO ERROR REGISTERS.
3585 016046 104002              ERROR    2          ;ISSUE ERROR MESSAGE
3586 016050 000760              BR        1$          ;DO AGAIN
3587 016052 013701 002164      NOER:   MOV    SVER1,R1      ;RESTORE R1 AND
3588 016056 013702 002166      MOV    SVER2,R2      ;R2.
3589 016062 012737 015204 015226      MOV    #STTER,STTEP    ;RESET STATUS ERROR POINTER.
3590 016070 013746 002222      MOV    ROSVE,-(SP)    ;PUSH ROSVE ON STACK
3591 016074 000200              RTS     R0          ;EXIT
3592
3593      ;*****
3594      ;SUBROUTINE TO CLEAR ERROR/DATA OUTPUT LOCATIONS. NEEDED
3595      ;ONLY WHEN DISPLAYING BYTES IN WORD LOCATIONS.
3596      ;*****
3597
3598 016076 005037 001120      CLREG:  CLR    $GDADR
3599 016102 005037 001122      CLR    $BDADR
3600 016106 005037 001124      CLR    $GDDAT
3601 016112 005037 001126      CLR    $BDDAT
3602 016116 000200              RTS     R0
3603      ;*****
3604      ;SUBROUTINE TO TRANSMIT THE BUFFER AND WAIT FOR XMIT DONE
3605      ;AND END OF RECEIVE MESSAGE. SUBROUTINE WILL LOOP IF LOCATION
3606      ;RECITT IS PRE-LOADED WITH A NUMBER HIGHER THAN(IE. MULTIPLE
3607      ;RECEIVES CAN BE ACCOMPLISHED WITH ONLY ONE ENTRY TO SUB-
3608      ;ROUTINE).WDSTOR AND BYSTOR ARE THE WORD(CURSOR POS.) AND BYTE
3609      ;STORAGE LOCATIONS,RESPECTIVELY.DEFAULT STORAGE IS THE REC. BUFFER.
3610
3611      ;*****
3612
3613 016120      XMREC:
3614 016120 010546              MOV    R5,-(SP)      ;:PUSH R5 ON STACK
3615 016122 012737 001001 002226      MOV    #1001,BLKM      ;:SET UP FOR A SOM/EOM TRANSMIT.
3616 016130 042737 077577 002224      BIC    #77577,VSTAT    ;:CLEAR ALL FLAGS BUT XOFF AND XMKIL.
3617 016136 013701 016372              MOV    BYSTOR,R1    ;:LOAD THE STORAGE POINTERS
3618 016142 013702 016370              MOV    WDSTOR,R2
3619 016146 052777 000100 163562      BIS    #TENA,@VXCSR    ;:SET INTERRUPT ENABLES
3620 016154 042737 061466 002224      XMITT: BIC    #61466,VSTAT    ;:CLEAR SOM,EOM,EPL,ESC,REV.VID., PARA. DELIM.,IDENT,CUR.
3621 016162 005037 002220      1$:    CLR    DLAY      ;:SET UP TIME OUT DELAY.
3622 016166 032737 000001 002224      BIT    #XMDNE,VSTAT    ;:IS XMIT DONE?
3623 016174 001015              BNE     3$          ;:YES-LOOK FOR RECEIVE DONE.
3624 016176 032737 020000 002224      2$:    BIT    #REOM,VSTAT    ;:RECEIVED AN EOM?
3625 016204 001401              BEQ     20$         ;:NO
3626 016206 000435              BR      CKSTR      ;:YES-GO HANDLE DATA
3627 016210 032737 100000 002224      20$:   BIT    #RXOFF,VSTAT    ;:NO- IS XOFF SET?
3628 016216 001761              BEQ     1$          ;:NO-STILL TRANSMITONG.
3629 016220 005337 002220              DEC    DLAY      ;:YES- RUN DELAY
3630 016224 001364              BNE     2$          ;:WAITING FOR XON
3631 016226 000416              BR      XMAD2      ;:NO XON-REPORT VT61 FAILURE.
3632
3633 016230 013705 031602      3$:    MOV    ABUFP,R5      ;:LOAD CH. RECEIVED FLAG.
3634 016234 005037 002220              CLR    DLAY      ;:SET UP RECEIVE DELAY.
3635 016240 032737 020000 002224      4$:    BIT    #REOM,VSTAT    ;:RECEIVE END OF MESSAGE?
3636 016246 001015              BNE     CKSTR      ;:YES-CHECK DATA STORAGE POINTERS
3637 016250 020537 031602      CMP     R5,ABUFP    ;:RECEIVED ANOTHER CHARACTER?
  
```

```

3638 016254 001365          BNE      3$          ;YES-RESET CH. FLAG AND DELAY
3639 016256 005337 002220  5$:      DEC      DLAY      ;RUN DELAY
3640 016262 001366          BNE      4$          ;AND KEEP LOOKING FOR EOM.
3641 016264 062700 000002  XMD2:    ADD      #2,R0      ;TIME OUT OCCURRED-SET UP ERROR EXIT.
3642 016270 005237 002200          INC      FTLCNT     ;INCREMENT FATAL XMIT COUNT.
3643 016274 004037 016520          JSR      R0,RESPTR   ;AND REST ALL INTERRUPT POINTERS.
3644 016300 000422          BR       CKVST
3645 016302 020102          CKSTR:  CMP      R1,R2      ;STORAGE POINTERS CLEARED?
3646 016304 001413          BEQ      CHKITT     ;YES--LEAVE DATA IN REC. BUFFER.
3647 016306 032737 000004 002224  BIT      #CURPOS,VSTAT ;RECEIVED A CURSOR POSITION?
3648 016314 001403          BEQ      STRBYT     ;NO-GO STORE A BYTE.
3649 016316 017722 176652          MOV      @RBBUF,(R2)+ ;YES,STORE IT.
3650 016322 000404          BR       CHKITT     ;AND CHECK ITERATION COUNT.
3651 016324 005701          STRBYT: TST      R1        ;STORING A CHAR?
3652 016326 001402          BEQ      CHKITT     ;NO
3653 016330 117721 176640          MOVB     @RBBUF,(R1)+ ;STORE A RECEIVED BYTE
3654 016334 005337 016366          CHKITT: DEC      RECITT   ;DONE RECEIVING?
3655 016340 001305          BNE      XMITT     ;NO-LOOP SUBROUTINE
3656 016342 004037 021202          JSR      R0,CKOFF    ;SEE IS XOFF IS UP.
3657 016346          CKVST:
3658 016346 012746 060001          MOV      #60001,-(SP)  ;;PUSH #60001 ON STACK
3659 016352 004037 015734          JSR      R0,CKSFT
3660 016356 004037 016520          JSR      R0,RESPTR   ;RESET INTERRUPT POINTERS.
3661 016362 012605          MOV      (SP)+,R5        ;;POP STACK INTO R5
3662 016364 000200          XMIT:   RTS      R0        ;EXIT SUBROUTINE.
3663 016366 000000          RECITT: .WORD 0          ;RECEIVE ITERATION COUNT.
3664 016370 000000          WDSTOR: .WORD 0          ;WORD STORAGE POINTER
3665 016372 000000          BYSTOR: .WORD 0          ;BYTE STORAGE POINTER
3666
3667          ;;*****
3668          ;SUBROUTINE TO XMIT THE BYTE AT TBUFP.
3669          ;;*****
3670
3671 016374 042737 000001 002224  XMIT1: BIC      #1,VSTAT      ;CLEAR XMIT DONE FLAG
3672 016402 012737 000001 015510  MOV      #1,XMCNT     ;SET UP TO XMIT 1 BYTE
3673 016410 052777 000100 163320  BIS      #TENA,@VXCSR
3674 016416          1$:
3675 016416 012746 000001          MOV      #XMDNE,-(SP)    ;;PUSH #XMDNE ON STACK
3676 016422 005737 002232          TST      PTYP      ;PROCESSOR TYPE
3677 016426 001403          BEQ      2$          ;IF 0 THEN 11/45-11/70
3678 016430 012746 000001          MOV      #1,-(SP)      ;;PUSH #1. ON STACK
3679 016434 000402          BR       3$          ;
3680 016436          2$:
3681 016436 012746 000002          MOV      #2,-(SP)      ;;PUSH #2. ON STACK
3682 016442 004037 021076          3$:  JSR      R0,WTBGND    ;LOOK FOR XMIT DONE
3683 016446 000401          BR       FTLEXT     ;HUNG TRANSMIT-CLEAR FLAGS AND EXIT
3684 016450 000402          BR       NORXT     ;NORMAL EXIT.
3685 016452 005037 002224          FTLEXT: CLR      VSTAT     ;CLEAR ANY FLAGS
3686 016456 000200          NORXT:  RTS      R0        ;AND EXIT
3687
3688          ;;*****
3689          ;SUBROUTINE TO ISSUE A BYTE AT A TIME UNTIL A ZERO
3690          ;BYTE IS ENCOUNTERED.
3691          ;;*****
3692
3693 016460 112777 000002 177020  LDxMIT: MOVB     #SOM,@TBUFP    ;SEND THE START OF MESSAGE.

```

```

3694 016466 000403          BR      2$
3695 016470 112377 177012 1$:  MOVB  (R3)+,@TBUFP ;MOVE A BYTE TO XMIT BUFFER
3696 016474 001403          BEQ      LDOUT ;IF A ZERO BYTE-EXIT
3697 016476 004037 016374 2$:  JSR      RO,XMIT1 ;GO XMIT A BYTE
3698 016502 000772          BR      1$ ;XMIT AGAIN.
3699 016504 112777 000004 176774 LDOUT: MOVB  #EOM,@TBUFP ;SEND THE END OF MESSAGE.
3700 016512 004037 016374      JSR      RO,XMIT1
3701 016516 000200          RTS      RO
3702
3703 ;*****
3704 ;ROUTINE TO RESET ALL INTERRUPT POINTERS.
3705 ;*****
3706
3707 016520 042777 000100 163210 RESPTR: BIC      #TENA,@VXCSR ;CLEAR INTERRUPT ENABLES
3708 016526 013737 015174 015200      MOV      RBBUF,RBUFP ;RESET RECEIVE BUF POINTER
3709 016534 013737 015502 015506      MOV      TBBUF,TBUFP ;RESET XMIT BUF POINTER
3710 016542 012737 015204 015226      MOV      #STTER,STTEP ;RESET RECEIVE STATUS ERR POINTER
3711 016550 005037 015510          CLR      XMCNT ;CLEAR TRANSMIT COUNT
3712 016554 005037 015202          CLR      ESAMB ;CLEAR ESC ASSEMBLY FLAGS
3713 016560 012737 000001 016366      MOV      #1,RECITT ;RESET REC. ITERATION COUNT
3714 016566 005037 016370          CLR      WDSTOR ;CLEAR STORAGE POINTERS
3715 016572 005037 016372          CLR      BYSTOR
3716 016576 000200          RTS      RO
3717
3718 ;*****
3719 ;SUBROUTINE TO ISSUE CURSOR POSITION ERROR. GOOD
3720 ;LINE/COLUMN MUST BE A WORD ON STACK. ERROR
3721 ;POSITION IS EXPECTED TO BE @RBBUF.
3722 ;*****
3723
3724 CURER:
3725 016600          MOV      (SP)+,ROSVE ;POP STACK INTO ROSVE
3726 016600 012637 002222          MOV      (SP)+,CHRD ;POP STACK INTO CHRD
3727 016604 012637 002162          SUB      #20040,CHRD ;EXTRACT MOD 40 FROM GOOD POSITION
3728 016610 162737 020040 002162      JSR      RO,CLREG
3729 016616 004037 016076          MOVB  CHRD+1,$GDDAT ;LOAD MESSAGE WITH GOOD
3730 016622 113737 002163 001124      MOVB  CHRD,$GDADR ;LINE AND COLUMN
3731 016630 113737 002162 001120      MOV      @RBBUF,CHRD ;LINE AND COLUMN.
3732 016636 017737 176332 002162      SUB      #20040,CHRD ;EXTRACT MOD 40 FROM BAD POSITION.
3733 016644 162737 020040 002162      MOVB  CHRD+1,$BDDAT ;LOAD MESSAGE WITH BAD
3734 016652 113737 002163 001126      MOVB  CHRD,$BDADR ;LINE AND COLUMN.
3735 016660 113737 002162 001122      MOV      ERROR 6 ;ISSUE ERROR
3736 016666 104006          MOV      ROSVE,-(SP) ;PUSH ROSVE ON STACK
3737 016670 013746 002222          RTS      RO
3738 016674 000200
3739
3740 ;*****
3741 ;*****
3742 ;SUBROUTINE TO DECREMENT CURSOR POSITION IN A
3743 ;LINEAR SEQUENCE. (IE. ROW 20, COL 1 ;ROW 20 COL0 ;ROW 17, COL 157).
3744 ;*****
3745
3746 CMPOS: CMPB  LNRW+1,#40 ;AT LEFT EDGE OF ROW?
3747 016676 123727 017003 000040      BEQ      1$ ;YES, GO ADJUST COL., ROW.
3748 016704 001403          DECB  LNRW+1 ;NO, DECREMENT COL. AND EXIT
3749 016706 105337 017003

```

```

3750 016712 000200      RTS      R0
3751 016714 123727 017002 000040 1$:  CMPB     LNRW,#40      ;AT ROW 0?
3752 016722 001405      BEQ      2$      ;YES, NO DECREMENT POSSIBLE-EXIT.
3753 016724 105337 017002      DECB     LNRW      ;NO, DECREMENT ROW AND
3754 016730 112737 000157 017003      MOVB    #157,LNRW+1 ;SET COL. TO RIGHT EDGE.
3755 016736 000200      2$:      RTS      R0
3756
3757      ;*****
3758      ;SUBROUTINE TO INCREMENT CURSOR POSITION IN A LINEAR
3759      ;SEQUENCE (IE. ROW 10, COL 78, ROW 10, COL 79, ROW 11 , COL 0).
3760      ;*****
3761
3762 016740 123727 017003 000157 CPPOS:  CMPB     LNRW+1,#157    ;AT RIGHT EDGE OF ROW
3763 016746 001403      BEQ      1$      ;YES, ADJUST ROW AND COLUMN.
3764 016750 105237 017003      INCB     LNRW+1    ;NO, INCREMENT COL. COUNT
3765 016754 000200      RTS      R0      ;AND EXIT
3766 016756 123727 017002 000067 1$:  CMPB     LNRW,#67      ;AT BOTTOM ROW?
3767 016764 001405      BEQ      2$      ;YES, NO INCREMENT POSSIBLE-EXIT.
3768 016766 105237 017002      INCB     LNRW      ;NO, INCREMENT ROW COUNT AND
3769 016772 112737 000040 017003      MOVB    #40,LNRW+1    ;SET COL. TO LEFT EDGE.
3770 017000 000200      2$:      RTS      R0
3771
3772 017002 000000      LNRW:    .WORD    0      ;CONTAINS UPDATED CURSOR POSITION.
3773      ;*****
3774
3775      ;SUBROUTINE TO XMIT, RECEIVE AND COMPARE. DATA ERRORS
3776      ;ARE REPORTED FROM SUBROUTINE. IF THE TRANSMIT OR
3777      ;RECEIVE LOOPS 'TIME OUT', EXIT FROM SUBROUTINE WILL
3778      ;BE NORMAL EXIT +2. SUBROUTINE ENTERED WITH (R1)=
3779      ;GOOD DATA BUFFER, (R2)-RECEIVE DATA BUFFER AND
3780      ;R3-COMPARE COUNT. IF THE VT61 DOES NOT HANG,THE ROUTINE
3781      ;WILL WAIT FOR END OF REC. MESSAGE(EOM).
3782
3783      ;*****
3784
3785
3786 017004      XRCMP:
3787 017004 010446      MOV      R4,-(SP)      ;PUSH R4 ON STACK
3788 017006 005004      CLR      R4      ;USE R4 A RECEIVE COUNTER.
3789 017010 012737 001001 002226      MOV      #1001,BLKM      ;SET UP FOR A SOM/EOM TRANSMIT.
3790 017016 042737 077577 002224      BIC      #77577,VSTAT      ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
3791 017024 052777 000100 162704      BIS      #TENA,@VXCSR      ;SET INTERRUPT ENABLES.
3792 017032 005037 017354      CLR      HDFLG      ;CLEAR ERROR 13 PRINT FLAG
3793 017036 012705 031552      MOV      #TCRLB+450,R5      ;R5 IS ERROR STORAGE POINTER
3794 017042 005037 002220      1$:      CLR      DLAY      ;SET UP TIME OUT DELAY
3795 017046 032737 000001 002224      BIT      #XMDNE,VSTAT      ;XMIT DONE?
3796 017054 001014      BNE      XREC      ;YES-GO RECEIVE
3797 017056 023737 015174 015200      2$:      CMP      RBBUF,RBUFP      ;HAS RECEIVE OPERATION BEGUN?
3798 017064 103410      BLO      XREC      ;YES-GO RECEIVE
3799 017066 032737 100000 002224      BIT      #RXOFF,VSTAT      ;XMIT XOFF SET?
3800 017074 001762      BEQ      1$      ;NO-KEEP LOOKING FOR XMIT DONE?
3801 017076 005337 002220      DEC      DLAY      ;YES RUN DELAY AND LOOK
3802 017102 001365      BNE      2$      ;FOR XON OR RECEIVED CH.
3803 017104 000432      BR       XRERR      ;TRANSMIT TIMEOUT-SET UP ERROR EXIT
3804
3805 017106 005037 002220      XRFC:    CLR      DLAY      ;SET UP TIME OUT DELAY
  
```

3806	017112	020237	015200	1\$:	CMP	R2,RBUF	;INSURE COMPARE POINTER	
3807	017116	103410			BLO	2\$	;LESS THAN RECEIVE POINTER	
3808	017120	032737	020000 002224		BIT	#REOM,VSTAT	;RECEIVE EOM?	
3809	017126	001030			BNE	XREXT	;YES-SET UP TO EXIT	
3810	017130	005337	002220		DEC	DLAY	;RUN TIMEOUT DELAY	
3811	017134	001416			BEQ	XRERR	;TIME OUT OCCURRED-ERROR EXIT	
3812	017136	000765			BR	1\$	;RETURN TO CHECK RECEIVE COUNT	
3813	017140	005204		2\$:	INC	R4	;ADD 1 TO RECEIVE COUNTER.	
3814	017142	122122			CMPB	(R1)+,(R2)+	;COMPARE CHARACTERS	
3815	017144	001407			BEQ	4\$	;EQUAL-COMPARE AGAIN	
3816	017146	020527	031602		CMP	R5,#TCRLB+500	;ALLREADY STORED 50 ERRORS?	
3817	017152	103004			BHIS	4\$	;YES-BYPASS STORAGE	
3818	017154	114125			MOVB	-(R1),(R5)+	;STORE GOOD DATA	
3819	017156	114225			MOVB	-(R2),(R5)+	;STORE BAD DATA	
3820	017160	010425			MOV	R4,(R5)+	;LOAD RECEIVE COUNT	
3821	017162	132122			BITB	(R1)+,(R2)+	;RESET POINTERS AND	
3822	017164	005303		4\$:	DEC	R3	;CHECK COMPARE COUNT	
3823	017166	001410			BEQ	XREXT	;ALL DONE-EXIT	
3824	017170	000746			BR	XREC	;COMPARE ANOTHER	
3825	017172	062700	000002	XRERR:	ADD	#2,R0	;SET UP ERROR EXIT	
3826	017176	005237	002200		INC	FTLCNT	;INCREMENT FATAL XMIT COUNT.	
3827	017202	004037	016520		JSR	R0,RESPTR	;RESET INTERRUPT POINTERS.	
3828	017206	000452			BR	XROUT		
3829	017210			XREXT:				
3830	017210	012746	020000		MOV	#REOM,-(SP)	::PUSH #REOM ON STACK	
3831	017214	005737	002232		TST	PTYPE	;PROCESSOR TYPE	::++C
3832	017220	001407			BEQ	3\$	;IF 0 THEN 11/45-11/70	::++C
3833	017222	100403			BMI	2\$	;IF -1 THEN 1135,40	::++C
3834	017224	012746	000002		MOV	#2,-(SP)	::PUSH #2. ON STACK	
3835	017230	000405			BR	4\$		::++C
3836	017232			2\$:				
3837	017232	012746	000004		MOV	#4,-(SP)	::PUSH #4. ON STACK	
3838	017236	000402			BR	4\$		::++C
3839	017240			3\$:				
3840	017240	012746	000010		MOV	#8,-(SP)	::PUSH #8. ON STACK	
3841	017244	004037	021076	4\$:	JSR	R0,WTBGND	;WAIT EOM	::++C
3842	017250	000431			BR	XROUT	;NO EOM-ISSUE ERROR AND EXIT.	
3843	017252	162705	031552		SUB	#TCRLB+450,R5	;NOW EXTRACT ERROR COUNT-IF ANY.	
3844	017256	010501			MOV	R5,R1	;AND STORE IT IN R1	
3845	017260	012705	031552		MOV	#TCRLB+450,R5	;RELOAD ERROR POINTER	
3846	017264	005701			TST	R1	;TEST FOR ERRORS	
3847	017266	001422			BEQ	XROUT	;NO-CHECK STATUS AND EXIT	
3848	017270	005737	017354		TST	HDFLG	;DATA ERROR HEADER PRINTED?	
3849	017274	001003			BNE	1\$	;YES-BYPASS HEADER PRINT	
3850	017276	104012			ERROR	12	;PRINT DATA ERROR HEADER	
3851	017300	005237	017354		INC	HDFLG	;SET HEADER PRINT FLAG	
3852	017304	004037	016076	1\$:	JSR	R0,CLREG	;ERROR WAS LEGITIMATE. LOAD	
3853	017310	112537	001124		MOVB	(R5)+,\$GDDAT	;ERROR MESSAGE AND ISSUE	
3854	017314	112537	001126		MOVB	(R5)+,\$BDDAT	;IT.	
3855	017320	012537	001120		MOV	(R5)+,\$GDADR	;LOAD RECEIVE COUNT	
3856	017324	104004			ERROR	4	;ISSUE DATA COMPARE ERROR	
3857	017326	162701	000004		SUB	#4,R1	;DECREMENT ERROR COUNT	
3858	017332	001364			BNE	1\$	;PRINT ANOTHER IF NOT AT ZERO	
3859	017334	004037	021202	XROUT:	JSR	R0,CKOFF	;SEE IS XOFF IS UP.	
3860	017340	012746	060001		MOV	#60001,-(SP)	::PUSH #60001 ON STACK	
3861	017344	004037	015734		JSR	R0,CKSFT	;CHECK FOR VSTAT /STATUS ERR.	

```

3862 017350 012604          MOV      (SP)+,R4          ;;POP STACK INTO R4
3863 017352 000200          RTS      R0              ;;EXIT SUBROUTINE
3864
3865 017354 000000          HDPLG: 0                  ;INHIBIT PRINT FLAG.
3866
3867                          ;;*****
3868
3869                          ;SLBROUTINE TO CREATE A 'RULER' IN LOCATIONS 200
3870                          ;TO 317.
3871
3872                          ;;*****
3873
3874 017356 012701 031302      CRRUL:  MOV      #TCRLB+200,R1      ;LOAD STARTING ADDRESS
3875 017362 012702 130461      MOV      #130461,R2              ;LOAD INITIAL RULER ASCII CODES.
3876 017366 110221          1$:  MOV      R2,(R1)+              ;STORE A RULER BYTE IN XMIT BUF.
3877 017370 022701 031422      CMP      #TCRLB+320,R1          ;RULER COMPLETE?
3878 017374 103001          BHS      2$                        ;NO
3879 017376 000200          RTS      R0                        ;AND EXIT.
3880 017400 105202          2$:  INCB     R2                    ;INCREMENT ASCII BYTE
3881 017402 122702 000272      CMP      #272,R2                ;END OF REVERSE VIDEO?
3882 017406 001003          BNE      3$                        ;NO-SEE IF END OF NORMAL.
3883 017410 012702 030660      MOV      #030660,R2              ;SET UP TO ISSUE REVERSE 0.
3884 017414 000405          BR       5$
3885 017416 122702 000072      3$:  CMP      #72,R2              ;END OF NORMAL VIDEO?
3886 017422 001361          BNE      1$                        ;NOT AT END OF A VIDEO STRING.
3887 017424 012702 130460      MOV      #130460,R2              ;YES-SET UP TO ISSUE NORMAL 0.
3888 017430 110221          5$:  MOV      R2,(R1)+              ;DO IT
3889 017432 105202          INCB     R2                        ;SET BYTE TO NEXT ASCII CODE
3890 017434 000302          SWAB     R2                        ;REVERSE VIDEO MODE.
3891 017436 000753          BR       1$                        ;BEGIN NEXT STRING
3892
3893                          ;;*****
3894
3895                          ;SUBROUTINE TO DELAY 10 M.S. TIME THE NUMBER INLOCATION
3896                          ;DCOUNT. THE PROCESSOR TYPE PRE-DETERMINES THE # OF LOOPS
3897                          ;REQUIRED TO DELAY 10 M.S. FOR ONE ITERATION. LOCATION
3898                          ;PMULT IS PRE-LOADED WITH : 11/45 4, 11/40 - 2
3899                          ;AND 11/10 =1.
3900                          ;;*****
3901
3902 017440          DELAY:
3903 017442 010146          MOV      R1,-(SP)                  ;;PUSH R1 ON STACK
3904 017444 013702 017500      MOV      R2,-(SP)                  ;;PUSH R2 ON STACK
3905 017450 012701 002570      1$:  MOV      PMULT,R2              ;LOAD PROCESSOR MULTIPLIER
3906 017454 005301          2$:  MOV      #1400.,R1              ;LOAD 10 M.S. DELAY
3907 017456 001376          DEC      R1                        ;RUN BASIC DELAY
3908 017460 005302          BNE      -2                          ;RUN MULTIPLIER DELAY
3909 017462 001372          DEC      R2
3910 017464 005337 017502      BNE      2$                      ;RUN MULTIPLIER DELAY
3911 017470 001365          DEC      DCOUNT                    ;RUN ITERATION COUNT
3912 017472 012602          BNE      1$
3913 017474 012601          MOV      (SP)+,R2                  ;;POP STACK INTO R2
3914 017476 000200          MOV      (SP)+,R1                  ;;POP STACK INTO R1
3915
3916 017500 000000          RTS      R0
3917 017502 000000          PMULT: 0                          ;PROCESSOR MULTIPLIER
                          DCOUNT: 0                          ;ITERATION COUNT

```

```

3918
3919
3920
3921
3922
3923
3924
3925
3926
3927 017504 012705 000041
3928 017510 110521
3929 017512 005303
3930 017514 001001
3931 017516 000200
3932 017520 105205
3933 017522 122705 000177
3934 017526 001766
3935 017530 000767
3936
3937
3938
3939
3940
3941
3942
3943 017532 042737 077577 002224
3944 017540 013701 015502
3945 017544 012703 000500
3946 017550 004037 017504
3947 017554 012737 003600 015510
3948 017562 052777 000100 162146
3949
3950 017570 032737 000001 002224
3951 017576 001774
3952
3953
3954
3955
3956
3957
3958
3959
3960 017600 004037 015512
3961 017604 042737 077577 002224
3962 017612 012737 000005 015510
3963 017620 013701 015502
3964 017624 012721 000002
3965 017630 013721 002054
3966 017634 013721 002010
3967 017640 005237 015510
3968 017644 112221
3969 017646 001374
3970 017650 112711 000004
3971 017654 052777 000100 162054
3972 017662 032737 000001 002224
3973 017670 001774

;*****
;SUBROUTINE TO GENERATE A INCREMENTING PATTERN AT
; (R1)+. ENTER WITH R3 EQUAL TO # OF CH. TO CREAT.
; R5 IS UTILIZED AS A WORK REGISTER.
;*****
BLDINC: MOV #41,R5 ;LOAD R5 WITH INITIAL CH.
BLDINA: MOVB R5,(R1)+ ;MOVE A CH. TO BUFFER
DEC R3 ;DECREMENT BYTE COUNT
BNE 2$ ;NOT DONE-UPDATE PATTERN
RTS R0 ;EXIT-DONE.
2$: INCB R5 ;UPDATE CH. PATTERN
CMPB #177,R5 ;PATTERN EXCEEDED MAX?
BEQ BLDINC ;YES-RESET IT.
BR BLDINA ;NO-ISSUE CURRENT PATTERN.
;*****
;SUBROUTINE TO FILL THE SCREFN WITH INCREMENTING DATA
;*****
DATSC: BIC #77577,VSTAT ;CLEAR INTERRUPT FLAGS.
MOV TBBUF,R1
MOV #320,R3 ;FILL XMIT BUFFER WITH INCRE-
JSR R0,BLDINC ;MENTING PATTERN
10$: MOV #TOTCH,XMCNT ;SET UP TO XMIT 1920 BYTES
BIS #TENA,@VXCSR
1$: BIT #XMDNE,VSTAT ;XMIT DONE?
BEQ .-6 ;NO
;*****
;SUBROUTINE TO RESET VT61 AND DISPLAY MESSAGE
;POINTED TO BY R2.
;*****
DSMES: JSR R0,RESETV ;RESET THE UNIT AND WAIT FOR XON.
BIC #77577,VSTAT ;CLEAR ALL FLAGS EXCEPT XOFF AND XMKIL.
MOV #5,XMCNT ;PRE-LOAD XMIT COUNT.
MOV TBBUF,R1 ;LOAD XMIT BUFFER WITH:
MOV #SOM,(R1)+ ;START OF MESSAGE
MOV ESCO,(R1)+
MOV DRECT,(R1)+ ;DISABLE RECTANGULAR MODE
1$: INC XMCNT ;INCREMENT TRANSMIT COUNT
MOVB (R2)+,(R1)+ ;DISPLAY MESSAGE
BNE 1$
MOVB #EOM,(R1) ;TERMINATE WITH END OF MESSAGE.
BIS #TENA,@VXCSR ;XMIT IT AND WAIT FOR
2$: BIT #XMDNE,VSTAT ;DONE
BEQ 2$

```

```

3974 017672 000200          RTS      R0
3975
3976      ;*****
3977
3978      ;SUBROUTINE TO CONVERT A BINARY CHARACTER
3979      ;TO 3 OCTAL CHARACTERS. R1 CONTAINS BINARY
3980      ;NUMBER. RESULT IS STORED IN LOCATIONS SVER1,
3981      ;SVER2
3982
3983      ;*****
3984      BINOC:
3985      017674 010546          MOV      R5,-(SP)      ;:PUSH R5 ON STACK
3986      017676 012705 000002  MOV      #2,R5        ;:LOAD ITERATION COUNT
3987      017702 000403          BR       2$          ;:BYPASS SHIFTS FOR 1ST CONVERSION
3988      017704 106201          1$: ASRB      R1
3989      017706 106201          ASRB      R1          ;:SHIFT A CHAR INTO POSITION
3990      017710 106201          ASRB      R1
3991      017712 110165 002164  2$: MOVB     R1,SVER1(R5) ;:STORE THE BINARY OFFSET
3992      017716 142765 000370 002164 BICB     #370,SVER1(R5) ;:CLEAR NON ESSENTIAL BITS
3993      017724 152765 000060 002164 BISB     #60,SVER1(R5) ;:CONVERT OFFSET TO OCTAL
3994      017732 005305          DEC      R5          ;:DECREMENT CONVERSION COUNT
3995      017734 100363          BPL      1$          ;:NOT DONE CONVERT ANOTHER
3996      017736 112737 000040 002167  MOVB     #40,SVER2+1 ;:LOAD A SPACE
3997      017744 012605          MOV      (SP)+,R5    ;:POP STACK INTO R5
3998      017746 000200          RTS      R0
3999
4000      ;*****
4001      ;SUBROUTINE TO CONVERT AN OCTAL CHAR. TO BINARY. REG
4002      ;R1 CONTAINS OCTAL AND REG R2 IS BINARY ASSEMBLY AREA.
4003      ;*****
4004
4005      017750 042701 177770  OCTBIN: BIC      #177770,R1 ;:EXTRACT OCTAL COMPONENT
4006      017754 005702          TST      R2          ;:FIRST CONVERSION?
4007      017756 001403          BEQ      NOSHFT      ;:YES - DO NOT SHIFT
4008      017760 006302          ASL      R2          ;:NO - SHIFT PREVIOUS CHAR.
4009      017762 006302          ASL      R2
4010      017764 006302          ASL      R2
4011      017766 060102  NOSHFT: ADD      R1,R2          ;:ADD CURRENT CHAR.
4012      017770 000200          RTS      R0
4013
4014      ;*****
4015      ;ROUTINE TO WAIT FOR C/R FROM VT61 UNDER TEST
4016      ;*****
4017
4018      017772 032777 000200 161732 GTCR: BIT      #RECDN,@VRCSR ;:WAIT FOR REVEIVE DONE
4019      020000 001774          BEQ      -6
4020      020002 127737 161726 001752 CMPB     @VRBUF,CARRT ;:CHAR = CARRIAGE RETURN?
4021      020010 001370          BNE      GTCR        ;:NO-KEEP LOOKING
4022      020012 000200          RTS      R0          ;:YES-EXIT
4023
4024      ;*****
4025      ;SUBROUTINE TO GET A CHARACTER (NUMERIC) FROM THE
4026      ;CONSOLE. IF OTHER THAN A NUMERIC IS TYPED A
4027      ;"?" WILL BE ECHOED.
4028      ;*****
4029

```

```

4030
4031 020014 004037 014006          GTNUM: JSR      R0,CONRD      ;GET A CHAR
4032 020020 012601                MOV      (SP)+,R1      ;;POP STACK INTO R1
4033 020022 122701 000054          CMPB     #54,R1        ;CHAR. =COMMA?
4034 020026 001411                BEQ       1$            ;YES-GO PRINT IT
4035 020030 123701 001752          CMPB     CARRT,R1      ;CHAR. = CARRIAGE RETURN?
4036 020034 001406                BEQ       1$
4037 020036 120127 000060          CMPB     R1,#60
4038 020042 103421                BLO       QUST          ;IF CHAR. IS LESS THAN 60
4039 020044 120127 000067          CMPB     R1,#67        ;OR MORE THAN 67, TYPE
4040 020050 101016                BHI       QUST          ;A QUESTION MARK
4041 020052 110137 020122          1$:  MOVB     R1,TYPNUM
4042 020056 104401 020122          TYPE     ,TYPNUM
4043 020062 123701 001754          CMPB     LNFED,R1
4044 020066 001406                BEQ       GTEXT
4045 020070 123701 001752          CMPB     CARRT,R1      ;IF CHAR. - C/R SET UP TO ISSUE
4046 020074 001003                BNE       GTEXT        ;LINE FEED BEFORE EXITING.
4047 020076 113701 001754          MOVB     LNFED,R1
4048 020102 000763                BR        1$
4049 020104 000200                GTEXT:  RTS      R0        ;GOOD CHAR., EXIT
4050 020106 112737 000077 020122  QUST:  MOVB     #77,TYPNUM
4051 020114 104401 020122          TYPE     ,TYPNUM      ;TYPE QUESTION MARK AND
4052 020120 000735                BR        GTNUM        ;KEEP LOOKING.
4053 020122      000          TYPNUM: .BYTE     0
4054 020123      000          .BYTE     0
4055
4056
4057          ;;*****
4058          ;SUBROUTINE TO CALCULATE CHECKSUM ON THE LOWER
4059          ;BYTE OF R5. R4 IS STORAGE FOR THE CHECKSUM
4060          ;CHARACTER. ALGORITHM FOR CHECKSUM IS ROTATE
4061          ;CURRENT ONE PLACE LEFT AND XOR NEW CHAR. CHECKSUM

```

```

4062                                     ;IS THE LOWER 7 BITS OF R4
4063
4064                                     ;:*****
4065
4066 020124 042705 177400  CALCK: BIC      #177400,R5      ;CLEAR UPPER BYTE OF R5
4067 020130 120527 000021      CMPB     R5,#XON      ;CHAR. =XON?
4068 020134 001415      BEQ      NOCALC      ;YES DO NOT CALCULATE CHECKSUM
4069 020136 120527 000023      CMPB     R5,#XOFF     ;CHAR =XOFF?
4070 020142 001412      BEQ      NOCALC      ;YES DO NOT CALCULATE CHECKSUM
4071
4072 020144 000241      CLC                      ;INSURE CARRY BIT INITIALLY CLEAR
4073 020146 105704      TSTB     R4              ;SET UP TO ROTATE R4
4074 020150 100001      BPL      1$              ;A FULL 8 BYTES
4075
4076 020152 000261      1$: SEC                      ;R4 WAS NEG. SO ROTATE A ONE
4077 020154 106104      RCLB     R4              ;INTO LOW ORDER BIT.
4078 020156 010403      MOV      R4,R3
4079 020160 040503      BIC      R5,R3          ;NOT A AND B
4080 020162 040405      BIC      R4,R5          ;NOT B AND A
4081 020164 050305      BIS      R3,R5          ;ORED
4082 020166 010504      MOV      R5,R4          ;EQUAL NEW CHECKSUM
4083 020170 000200      NOCALC: RTS     R0
4084                                     ;:*****
4085
4086                                     ;SUBROUTINE TO LOAD XMIT BUFFER FROM R0 THRU R1
4087                                     ;:*****
4088
4089 020172 112021      LDBUF: MOVB     (R0)+,(R1)+      ;LOAD A BYTE
4090 020174 001376      BNE      -2              ; UNTIL ZERO BYTE FOUND.
4091 020176 000200      RTS      R0
4092                                     ;:*****
4093                                     ;SUBROUTINE TO CHECK THE VT61 FOR A PERIPHERAL ABORT.
4094                                     ;:*****
4095
4096 020200 032737 010000 002224  CKABRT: BIT      #PABRT,VSTAT      ;ABORT FLAG RECEIVED?
4097 020206 001445      BEQ      2$              ;NO-EXIT
4098 020210 010037 001124      MOV      R0,$GDDAT
4099 020214 162737 000004 001124      SUB      #4,$GDDAT      ;POINT ERR PC TO MAIN ROUTINE.
4100 020222 013737 002224 001126      MOV      VSTAT,$BDDAT
4101 020230 104020      ERROR     20              ;ISSUE PERIPHERAL ABORT ERROR
4102
4103 020232 013701 015502      MOV      TBBUF,R1
4104 020236 004037 020172      JSR      R0,LDBUF      ;LOAD THE XMIT BUFFER WITH.
4105 020242 033      117      137      .BYTE     .ESC,.O,.IABT,.ESC,.O,.RABT
4106 020245 033      117      140      .BYTE     .ESC,.O,.UNLKKB,0
4107 020250 033      117      145
4108 020253 000
4109 020254 012737 000011 015510      MOV      #9.,XMCNT      ;SET UP TO XMIT 9 BYTES.
4110 020262 004037 016120      JSR      R0,XMREC      ;XMIT AND RECEIVE.
4111 020266 000240
4112 020270 123727 015230 000170      NOP
4113 020276 001411      CMPB     STRO,#NABRT      ;ABORT FLAG CLEARED?
4114 020300 010037 001124      BEQ      2$              ;YES-EXIT
4115 020304 162737 000004 001124      MOV      R0,$GDDAT      ;NO-SET UP AND ISSUE A CANT
4116 020312 013737 002224 001126      SUB      #4,$GDDAT      ;CLEAR ABORT FLAG ERROR MESSAGE.
4117 020320 104021      MOV      VSTAT,$BDDAT
4118                                     ERROR     21

```

```

4118 020322 000200      2$:      RTS      R0
4119
4120      ;*****
4121      ;SUBROUTINE TO COMPARE RECEIVED KEYBOARD POSITION WITH
4122      ;EXPECTED KEYBOARD POSITION. ERRORS ARE REPORTED
4123      ;AS POSITIONAL ERRORS AND NOT DATA COMPARE ERRORS.
4124      ;*****
4125
4126
4127 020324 105077 011252      CKKBD:  CLRB      @ABUFP      ;CLEAR RECEIVE BYTE
4128 020330 005037 002162      CLR      CHR      ;CLEAR INPUT STORAGE.
4129 020334 105777 011242      KBDLP:  TSTB      @ABUFP      ;WAIT FOR A INPUT.
4130 020340 001775      BEQ      .-4
4131
4132 020342 117737 011234 002162      MOV      @ABUFP ,CHR      ;STORE IT AND
4133 020350 105077 011226      CLRB      @ABUFP      ;CLEAR THE INPUT AREA.
4134 020354 123714 002162      1$:      CMP      CHR      ,(R4)      ;RECEIVED EQUAL EXPECTED?
4135 020360 001500      BEQ      GDSTRK      ;NO-UPDATE POINTERS.
4136 020362 005237 002210      INC      BUBCT      ;INCREMENT ERROR COUNT.
4137 020366 023727 002210 000012      CMP      BUBCT,#10.      ;COUNT = 10?
4138 020374 103075      BHIS      CNTF      ;YES-EXIT SUBROUTINE.
4139 020376 010401      MOV      R4,R1
4140 020400 166501 011766      SUB      DTIBL(R5),R1      ;EXTRACT KEY POSITION FROM ROW LOC.
4141 020404 005201      INC      R1      ;CONVERT LOGICAL POS. TO ACTUAL.
4142 020406 004037 017674      JSR      R0,BINOCT      ;GET KEY POSITION IN OCTAL.
4143 020412 113737 002166 002164      MOV      SVER2,SVER1      ;RE-ASSEMBLE OCTAL BYTES.
4144 020420 123727 002165 000060      CMP      SVER1+1,#60      ;POSITION LESS THAN 8?
4145 020426 001413      BEQ      LDPOS      ;YES-GO LOAD IT.
4146 020430 123727 002164 000062      CMP      SVER1,#62      ;POSITION GREATER THAN 8 AND LESS THAN12?
4147 020436 103404      BLO      BOROW      ;YES-SET UP TO BORROW.
4148 020440 162737 000002 002164      SUB      #2,SVER1      ;NO-JUST SUBTRACT 2.
4149 020446 000403      BR      LDPOS
4150 020450 162737 000370 002164      BOROW:  SUB      #370,SVER1      ;SUBTRACT AND BORROW.
4151 020456 113737 002164 030213      LDPOS:  MOV      SVER1,KYSTRK+1      ;LOAD THE CONVERTED DECIMAL #.
4152 020464 113737 002165 030212      MOV      SVER1+1,KYSTRK
4153 020472 012703 030135      DMP OCT:  MOV      #DKBERR ,R3
4154 020476 004037 016460      JSR      R0 ,LDXMIT      ;ISSUE BODY OF KEYBOARD ERROR.
4155 020502 111401      MOV      (R4) ,R1
4156 020504 004037 017674      JSR      R0 ,BINOCT
4157 020510 012703 002164      MOV      #SVER1 ,R3
4158 020514 004037 016460      JSR      R0 ,LDXMIT      ;CONVERT AND ISSUE GOOD CHAR.
4159 020520 012703 030244      MOV      #DSPC6,R3
4160 020524 004037 016460      JSR      R0,LDXMIT      ;INSERT 6 SPACES IN MESSAGE.
4161 020530 113701 002162      MOV      CHR      ,R1
4162 020534 004037 017674      JSR      R0 ,BINOCT
4163 020540 012703 002164      MOV      #SVER1 ,R3
4164 020544 004037 016460      JSR      R0 ,LDXMIT      ;CONVERT AND ISSUE RECEIVED CHAR.
4165 020550 012703 001171      MOV      #CRLF ,R3
4166 020554 004037 016460      JSR      R0 ,LDXMIT      ;ISSUE C/R AND L/F.
4167 020560 000665      BR      KBDLP      ;LOOK FOR SAME KEY AGAIN.
4168
4169 020562 005204      GDSTRK:  INC      R4      ;INCREMENT KEYBOARD ROW COUNTER.
4170 020564 105714      TSTB      (R4)      ;REACHED END OF ROW?
4171 020566 001262      BNE      KBDLP      ;NO-LOOK FOR NEXT INPUT
4172 020570 000200      CNTF:   RTS      R0      ;YES-EXIT.
4173

```

```

4174      ;:*****
4175
4176      ;SUBROUTINE TO LOOP DATA THROUGH HOST COMPUTER. ALL
4177      ;FUNCTIONS ARE ALLOWED, BUT BLOCK TRANSMITS WHICH
4178      ;EXCEED 552 BYTES WILL RESULT IN THE TERMINATION
4179      ;OF THE OPERATION AFTER 552 RECEIVED BYTES.
4180
4181      ;:*****
4182
4183 020572 005237 021074 LOOP: INC XMZER ;SET UP TO XMIT NULLS.
4184 020576 012737 031602 015176 MOV #TCRLB+500,REBUF ;RESET BUFFER POINTERS
4185 020604 012737 030402 015502 MOV #RCRLB,TBBUF
4186 020612 004037 016520 JSR RO,RESPTR ;RELOAD ALL INTERRUPT POINTERS
4187 020616 042737 077577 002224 BIC #77577,VSTAT ;CLEAR ALL FLAGS BUT XOFF AND XMKIL.
4188 020624 013704 015200 LOOPT: MOV RBUF,R4 ;SET UP RECEIVE FLAG
4189 020630 032737 000001 002224 LOOPTA: BIT #XMDNE,VSTAT ;XMIT COMPLETE?
4190 020636 001407 BEQ LOOPR ;NO
4191 020640 042737 000001 002224 BIC #XMDNE,VSTAT ;YES RESET FLAG
4192 020646 013737 015174 015200 MOV RBBUF,RBUF ;RESET THE REC. BUFFER POINTER
4193 020654 000763 BR LOOPT
4194 020656 032737 001400 002224 LOOPR: BIT #EPL+ESC,VSTAT ;RECEIVED AN ESC OR EPL?
4195 020664 001004 BNE LPSTR ;YES-GO CHECK IT
4196 020666 023704 015200 CMP RBUF,R4 ;RECEIVED A DISPLAY CHAR?
4197 020672 001756 BEQ LOOPTA ;NO-LOOP
4198 020674 000426 BR BUMPCT
4199 020676 117777 010700 174274 LPSTR: MOVB @ABUF,@RBUF ;YES LOAD IT IN THE BUFFER
4200 020704 005237 015200 INC RBUF ;AND INCREMENT BUFFER POINTER
4201 020710 005037 015202 CLR ESAMB ;CLEAR ESC ASSEMBLY WORD
4202 020714 042737 001400 002224 BIC #EPL+ESC,VSTAT ;CLEAR THE FLAGS
4203 020722 005237 015510 INC XMCNT ;INCREMENT XMIT COUNT
4204 020726 123777 002132 010646 CMPB ESCN,@ABUF ;CHAR. A ESC(033)?
4205 020734 001733 BEQ LOOPT ;YES WAIT FOR NEXT PART OF FUNCTION
4206 020736 113777 001754 174234 MOVB LNFED,@RBUF ;CHAR. WAS EPL ADD A LINE FEED.
4207 020744 005237 015200 INC RBUF
4208 020750 000407 BR FRCECT ;AND ISSUE THEM.
4209 020752 023727 015510 000764 BUMPCT: CMP XMCNT,#500. ;BUFFER ABOUT FILLED?
4210 020760 103403 BLO FRCECT ;NO
4211 020762 005337 015200 1$: DEC RBUF ;YES-RESET THE RECEIVE POINTER
4212 020766 000716 BR LOOPT
4213 020770 005237 015510 FRCECT: INC XMCNT ;INCREMENT THE XMIT COUNT
4214 020774 023727 015510 000002 CMP XMCNT,#2 ;FIRST CHAR TO XMIT?
4215 021002 101003 BHI XMWT ;NO
4216 021004 052777 000100 160724 BIS #TENA,@VXCSR ;YES-SET THE XMIT ENABLE
4217 021012 004037 021022 XMWT: JSR RO,EXIST ;LOOK FOR END OF TEST COMMAND.
4218 021016 000702 BR LOOPT ;NONE FOUND.
4219 021020 000200 RTS RO ;AND EXIT
4220
4221      ;:*****
4222      ;SUBROUTINE TO CHECK FOR END OF TEST COMMAND. THE CONTROL
4223      ;C KEY EXITS ALL TESTS EXCEPT THE BLOCK MODE TEST
4224      ;WHICH IS EXITED ON A @ KEY.
4225      ;:*****
4226
4227 021022 127727 010554 000003 EXIST: CMPB @ABUF,#3 ;LOOK FOR CONTROL C.
4228 021030 001020 BNE NOROUT
4229

```

```

4230 021032 012737 031101 015176 ABSXT: MOV #RCRLB+477,REBUF ;RESET THE BUFFERS
4231 021040 012737 031102 015502 MOV #TCRLB,TBBUF
4232 021046 004037 016520 JSR RO,RESPTR ;RESET ALL POINTERS
4233 021052 012702 026715 MOV #DXT,R2
4234 021056 004037 017600 JSR RO,DSMES ;ISSUE EXIT MESSAGE
4235 021062 005037 021074 CLR XMZER ;CLEAR THE ZERO TRANSMIT FLAG.
4236 021066 062700 000002 ADD #2,RO ;SET UP TEST EXIT.
4237 021072 000200 NOROUT: RTS RO ;EXIT SUBROUTINE.
4238
4239 021074 000000 XMZER: .WORD 0
4240
4241 ;*****
4242 ;SUB-ROUTINE TO LOOK FOR VSTAT BIT ON THE STACK
4243 ;DELAY FACTOR IS FIRST WORD ON THE STACK AND VSTAT BIT
4244 ;IS THE SECOND. MIN. DELAY IS 4 U.S FOR A MOS 11/45.
4245 ;*****
4246
4247 021076 WTBGND:
4248 021076 012637 002222 MOV (SP)+,ROSVE ;POP STACK INTO ROSVE
4249 021102 012637 021200 MOV (SP)+,VDLAY ;POP STACK INTO VDLAY
4250 021106 012637 021176 MOV (SP)+,VBIT ;POP STACK INTO VBIT
4251 021112 012737 000002 002234 MOV #2,DLYCNT ;GO THROUGH DELAY TWICE ;:++C
4252 021120 005037 002220 1$: CLR DLAY
4253 021124 033737 021176 002224 2$: BIT VBIT,VSTAT ;SENSED THE CONDITION?
4254 021132 001015 BNE FNDBT ;YES-EXIT.
4255 021134 005337 002220 DEC DLAY ;NO-RUN DELAY.
4256 021142 005337 002234 BNE 2$ ;LOOPED TWICE? ;:++C
4257 021146 001364 BNE 1$ ;IF NO, GO AGAIN. ;:++C
4258 021150 005337 021200 DEC VDLAY ;DELAY FACTOR EXPIRED?
4259 021154 001361 BNE 1$ ;NO-LOOP
4260 021156 104011 ERROR 11 ;DELAY EXPIRED-ISSUE HUNG NIT
4261 021160 005237 002200 INC FTLCNT ;INCREMENT FATAL XMIT COUNT.
4262 021164 000401 BR TIMEXT
4263 021166 005720 FNDBT: TST (RO)+ ;SET UP FOR NORMAL EXIT
4264 021170 TIMEXT:
4265 021170 013746 002222 MOV ROSVE,-(SP) ;PUSH ROSVE ON STACK
4266 021174 000200 RTS RO
4267 021176 000000 VBIT: 0
4268 021200 000000 VDLAY: 0
4269
4270 ;*****
4271 ;SUBROUTINE TO LOOK FOR XOFF BEFORE EXITING A RECEIVE ROUTINE.
4272 ;*****
4273 021202 005037 002220 CKOFF: CLR DLAY
4274 021206 032737 100000 002224 1$: BIT #RXOFF,VSTAT ;IS XOFF SET?
4275 021214 001403 BEQ 2$ ;NO-EXIT
4276 021216 005337 002220 DEC DLAY ;RUN DELAY.
4277 021222 001371 BNE 1$
4278 021224 000200 2$: RTS RO
4279
4280 .SBTTL SCOPE HANDLER ROUTINE
4281
4282 ;*****
4283 ;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
4284 ;AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
4285 ;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>

```

```

4286      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4287      ;*SW14=1      LOOP ON TEST
4288      ;*SW11=1      INHIBIT ITERATIONS
4289      ;*SW09=1      LOOP ON ERROR
4290      ;*SW08=1      LOOP ON TEST IN SWR<7:0>
4291      ;*CALL
4292      ;*      SCOPE      ;;SCOPE=10T
4293
4294      $SCOPE:
4295      021226 004037 014040      JSR      RO,MONIT
4296      021232 032777 040000 157700 1$:      BIT      #BIT14,@SWR      ;;LOOP ON PRESENT TEST?
4297      021240 001111      BNE      $OVER      ;;YES IF SW14=1
4298      ;*****START OF CODE FOR THE XOR TESTER*****
4299      021242 000416      $XTSTR: BR      6$      ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
4300      ;THIS INSTRUCTION TO A 'NOP' (NOP=240)
4301      021244 013746 000004      MOV      @#ERRVEC,-(SP)      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
4302      021250 012737 021270 000004      MOV      #5$,@#ERRVEC      ;;SET FOR TIMEOUT
4303      021256 005737 177060      TST      @#177060      ;;TIME OUT ON XOR?
4304      021262 012637 000004      MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
4305      021266 000463      BR      $SVLAD      ;;GO TO THE NEXT TEST
4306      021270 022626      5$:      CMP      (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
4307      021272 012637 000004      MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
4308      021276 000423      BR      7$      ;;LOOP ON THE PRESENT TEST
4309      021300      6$:;*****END OF CODE FOR THE XOR TESTER*****
4310      021300 032777 000400 157632      BIT      #BIT08,@SWR      ;;LOOP ON SPEC. TEST?
4311      021306 001404      BEQ      2$      ;;BR IF NO
4312      021310 127737 157624 001102      CMPB     @SWR,$TSTNM      ;;ON THE RIGHT TEST?      SWR<7:0>
4313      021316 001462      BEQ      $OVER      ;;BR IF YES
4314      021320 105737 001103      2$:      TSTB     $ERFLG      ;;HAS AN ERROR OCCURRED?
4315      021324 001421      BEQ      3$      ;;BR IF NO
4316      021326 123737 001115 001103      CMPB     $ERMAX,$ERFLG      ;;MAX. ERRORS FOR THIS TEST OCCURRED?
4317      021334 101015      BHI      3$      ;;BR IF NO
4318      021336 032777 001000 157574      BIT      #BIT09,@SWR      ;;LOOP ON ERROR?
4319      021344 001404      BEQ      4$      ;;BR IF NO
4320      021346 013737 001110 001106      7$:      MOV      $LPERR,$LPADR      ;;SET LOOP ADDRESS TO LAST SCOPE
4321      021354 000443      BR      $OVER
4322      021356 105037 001103      4$:      CLRB     $ERFLG      ;;ZERO THE ERROR FLAG
4323      021362 005037 001160      CLR      $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
4324      021366 000415      BR      1$      ;;ESCAPE TO THE NEXT TEST
4325      021370 032777 004000 157542      3$:      BIT      #BIT11,@SWR      ;;INHIBIT ITERATIONS?
4326      021376 001011      BNE      1$      ;;BR IF YES
4327      021400 005737 001100      TST      $PASS      ;;IF FIRST PASS OF PROGRAM
4328      021404 001406      BEQ      1$      ;;      INHIBIT ITERATIONS
4329      021406 005237 001104      INC      $ICNT      ;;INCREMENT ITERATION COUNT
4330      021412 023737 001160 001104      CMP      $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
4331      021420 002021      BGE      $OVER      ;;BR IF MORE ITERATION REQUIRED
4332      021422 012737 000001 001104      1$:      MOV      #1,$ICNT      ;;REINITIALIZE THE ITERATION COUNTER
4333      021430 013737 021500 001160      MOV      $MXCNT,$TIMES      ;;SET NUMBER OF ITERATIONS TO DO
4334      021436 105237 001102      $SVLAD: INCB     $TSTNM      ;;COUNT TEST NUMBERS
4335      021442 011637 001106      MOV      (SP),$LPADR      ;;SAVE SCOPE LOOP ADDRESS
4336      021446 011637 001110      MOV      (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
4337      021452 005037 001162      CLR      $ESCAPE      ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
4338      021456 112737 000001 001115      MOV      #1,$ERMAX      ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
4339      021464 013777 001102 157450      $OVER:  MOV      $TSTNM,@DISPLAY      ;;DISPLAY TEST NUMBER
4340      021472 013716 001106      MOV      $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
4341      021476 000002      RTI      ;;FIXES PS
  
```

```

4342 021500 000005 $MXCNT: 5 ;:MAX. NUMBER OF ITERATIONS
4343 .SBTTL ERROR HANDLER ROUTINE
4344
4345 ;:*****
4346 ;:THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
4347 ;:SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
4348 ;:AND GO TO $ERRTYP ON ERROR
4349 ;:THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4350 ;:SW15=1 HALT ON ERROR
4351 ;:SW13=1 INHIBIT ERROR TYPEOUTS
4352 ;:SW10=1 BELL ON ERROR
4353 ;:SW09=1 LOOP ON ERROR
4354 ;:CALL
4355 ;:
4356 ;: ERROR N ;:ERROR=EMT AND N=ERROR ITEM NUMBER
4357 $ERROR:
4358 021502 105237 001103 7$: INCB $ERFLG ;:SET THE ERROR FLAG
4359 021506 001775 BEQ 7$ ;:DON'T LET THE FLAG GO TO ZERO
4360 021510 013777 001102 157424 MOV $TSTNM,@DISPLAY ;:DISPLAY TEST NUMBER AND ERROR FLAG
4361 021516 032777 002000 157414 BIT #BIT10,@SWR ;:BELL ON ERROR?
4362 021524 001402 BEQ 1$ ;:NO - SKIP
4363 021526 104401 001164 TYPE $BELL ;:RING BELL
4364 021532 005237 001112 1$: INC $ERTT ;:COUNT THE NUMBER OF ERRORS
4365 021536 011637 001116 MOV (SP),$ERRPC ;:GET ADDRESS OF ERROR INSTRUCTION
4366 021542 162737 000002 001116 SUB #2,$ERRPC
4367 021550 117737 157342 001114 MOVB @ERRPC,$ITEMB ;:STRIP AND SAVE THE ERROR ITEM CODE
4368 021556 032777 020000 157354 BIT #BIT13,@SWR ;:SKIP TYPEOUT IF SET
4369 021564 001004 BNE 20$ ;:SKIP TYPEOUTS
4370 021566 004737 022070 JSR PC,$ERRTYP ;:GO TO USER ERROR ROUTINE
4371 021572 104401 001171 TYPE $CRLF
4372 021576 20$:
4373 021576 005777 157336 2$: TST @SWR ;:HALT ON ERROR
4374 021602 100001 BPL 3$ ;:SKIP IF CONTINUE
4375 021604 000000 HALT ;:HALT ON ERROR!
4376 021606 032777 001000 157324 3$: BIT #BIT09,@SWR ;:LOOP ON ERROR SWITCH SET?
4377 021614 001402 BEQ 4$ ;:BR IF NO
4378 021616 013716 001110 MOV $LPERR,(SP) ;:FUDGE RETURN FOR LOOPING
4379 021622 005737 001162 4$: TST $ESCAPE ;:CHECK FOR AN ESCAPE ADDRESS
4380 021626 001402 BEQ 5$ ;:BR IF NONE
4381 021630 013716 001162 MOV $ESCAPE,(SP) ;:FUDGE RETURN ADDRESS FOR ESCAPE
4382 021634 5$:
4383 021634 022737 010776 000042 CMP #ENDAD,@#42 ;:ACT-11 AUTO-ACCEPT?
4384 021642 001001 BNE 6$ ;:BRANCH IF NO
4385 021644 000000 HALT ;:YES
4386 021646 6$:
4387 021646 000002 RTI ;:RETURN
4388 .SBTTL TYPE ROUTINE
4389
4390 ;:*****
4391 ;:ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
4392 ;:THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
4393 ;:NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
4394 ;:NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
4395 ;:NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
4396 ;:
4397 ;:CALL:

```

```

4398      ;*1) USING A TRAP INSTRUCTION
4399      ;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
4400      ;*OR
4401      ;*      TYPE
4402      ;*      MESADR
4403      ;*
4404
4405 021650 105737 001157 $TYPE: TSTB $TPFLG      ;;IS THERE A TERMINAL?
4406 021654 100002      BPL 1$      ;;BR IF YES
4407 021656 000000      HALT      ;;HALT HERE IF NO TERMINAL
4408 021660 000407      BR 3$      ;;LEAVE
4409 021662 010046      1$: MOV RO,-(SP)      ;;SAVE RO
4410 021664 017600 000002      MOV @2(SP),RO      ;;GET ADDRESS OF ASCIZ STRING
4411 021670 112046      2$: MOVB (RO)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
4412 021672 001005      BNE 4$      ;;BR IF IT ISN'T THE TERMINATOR
4413 021674 005726      TST (SP)+      ;;IF TERMINATOR POP IT OFF THE STACK
4414 021676 012600      60$: MOV (SP)+,RO      ;;RESTORE RO
4415 021700 062716 000002      3$: ADD #2,(SP)      ;;ADJUST RETURN PC
4416 021704 000002      RTI      ;;RETURN
4417 021706 122716 000011      4$: CMPB #HT,(SP)      ;;BRANCH IF <HT>
4418 021712 001430      BEQ 8$      ;;BRANCH IF NOT <CRLF>
4419 021714 122716 000200      CMPB #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
4420 021720 001006      BNE 5$      ;;BRANCH IF NOT <CRLF>
4421 021722 005726      TST (SP)+      ;;POP <CR><LF> EQUIV
4422 021724 104401      TYPE      ;;TYPE A CR AND LF
4423 021726 001171      $CRLF
4424 021730 105037 022064      CLRB $CHARCNT      ;;CLEAR CHARACTER COUNT
4425 021734 000755      BR 2$      ;;GET NEXT CHARACTER
4426 021736 004737 022020      5$: JSR PC,$TYPEC      ;;GO TYPE THIS CHARACTER
4427 021742 123726 001156      6$: CMPB $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
4428 021746 001350      BNE 2$      ;;IF NO GO GET NEXT CHAR.
4429 021750 013746 001154      MOV $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
4430      ;;AND THE NULL CHAR.
4431 021754 105366 000001      7$: DECB 1(SP)      ;;DOES A NULL NEED TO BE TYPED?
4432 021760 002770      BLT 6$      ;;BR IF NO--GO POP THE NULL OFF OF STACK
4433 021762 004737 022020      JSR PC,$TYPEC      ;;GO TYPE A NULL
4434 021766 105337 022064      DECB $CHARCNT      ;;DO NOT COUNT AS A COUNT
4435 021772 000770      BR 7$      ;;LOOP
4436
4437      ;HORIZONTAL TAB PROCESSOR
4438
4439 021774 112716 000040      8$: MOVB #' ,(SP)      ;;REPLACE TAB WITH SPACE
4440 022000 004737 022020      9$: JSR PC,$TYPEC      ;;TYPE A SPACE
4441 022004 132737 000007 022064      BITB #7,$CHARCNT      ;;BRANCH IF NOT AT
4442 022012 001372      BNE 9$      ;;TAB STOP
4443 022014 005726      TST (SP)+      ;;POP SPACE OFF STACK
4444 022016 000724      BR 2$      ;;GET NEXT CHARACTER
4445 022020 105777 157124      $TYPEC: TSTB @STPS      ;;WAIT UNTIL PRINTER IS READY
4446 022024 100375      BPL $TYPEC
4447 022026 116677 000002 157116      MOVB 2(SP),@STPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
4448 022034 122766 000015 000002      CMPB #CR,2(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
4449 022042 001003      BNE 1$      ;;BRANCH IF NO
4450 022044 105037 022064      CLRB $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
4451 022050 000406      BR $TYPEX      ;;EXIT
4452 022052 122766 000012 000002      1$: CMPB #LF,2(SP)      ;;IS CHARACTER A LINE FEED?
4453 022060 001402      BEQ $TYPEX      ;;BRANCH IF YES

```

```

4454 022062 105227          INCB      (PC)+          ;;COUNT THE CHARACTER
4455 022064 000000          $CHARCNT: .WORD 0        ;;CHARACTER COUNT STORAGE
4456 022066 000207          $TYPEX: RTS      PC
4457
4458                          .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
4459
4460                          ;*****
4461                          ;*THIS ROUTINE USES THE 'ITEM CONTROL BYTE' ($ITEMB) TO DETERMINE WHICH
4462                          ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE 'ERROR TABLE' ($ERRTB),
4463                          ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
4464
4465                          $ERRTYP:
4466 022070 104401 001171      TYPE      , $CRLF          ;; 'CARRIAGE RETURN' & 'LINE FEED'
4467 022074 010046          MOV      R0, -(SP)          ;; SAVE R0
4468 022076 005000          CLR      R0                ;; PICKUP THE ITEM INDEX
4469 022100 153700 001114    BISB     @#$ITEMB, R0
4470 022104 001004          BNE      1$                ;; IF ITEM NUMBER IS ZERO, JUST
4471                          ;; TYPE THE PC OF THE ERROR
4472 022106 013746 001116    MOV      $ERRPC, -(SP)      ;; SAVE $ERRPC FOR TYPEOUT
4473                          ;; ERROR ADDRESS
4474 022112 104402          TYPDC     ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
4475 022114 000445          BR       10$                ;; GET OUT
4476 022116 005300 1$:      DEC      R0                ;; ADJUST THE INDEX SO THAT IT WILL
4477 022120 006300          ASL      R0                ;; WORK FOR THE ERROR TABLE
4478 022122 006300          ASL      R0
4479 022124 006300          ASL      R0
4480 022126 062700 001174    ADD      # $ERRTB, R0      ;; FORM TABLE POINTER
4481 022132 012037 022142    MOV      (R0)+, 2$        ;; PICKUP 'ERROR MESSAGE' POINTER
4482 022136 001404          BEQ      3$                ;; SKIP TYPEOUT IF NO POINTER
4483 022140 104401          TYPE     ;; TYPE THE 'ERROR MESSAGE'
4484 022142 000000 2$:      .WORD    0                ;; 'ERROR MESSAGE' POINTER GOES HERE
4485 022144 104401 001171    TYPE     , $CRLF          ;; 'CARRIAGE RETURN' & 'LINE FEED'
4486 022150 012037 022160 3$:      MOV      (R0)+, 4$        ;; PICKUP 'DATA HEADER' POINTER
4487 022154 001404          BEQ      5$                ;; SKIP TYPEOUT IF 0
4488 022156 104401          TYPE     ;; TYPE THE 'DATA HEADER'
4489 022160 000000 4$:      .WORD    0                ;; 'DATA HEADER' POINTER GOES HERE
4490 022162 104401 001171    TYPE     , $CRLF          ;; 'CARRIAGE RETURN' & 'LINE FEED'
4491 022166 010146 5$:      MOV      R1, -(SP)          ;; SAVE R1
4492 022170 012001          MOV      (R0)+, R1        ;; PICKUP 'DATA TABLE' POINTER
4493 022172 001415          BEQ      9$                ;; BR IF NO DATA TO BE TYPED
4494 022174 012000          MOV      (R0)+, R0        ;; PICKUP 'DATA FORMAT' POINTER
4495 022176 105720 6$:      TSTB     (R0)+            ;; 'OCTAL' OR 'DECIMAL'
4496 022200 001003          BNE      7$                ;; BR IF DECIMAL
4497 022202 013146          MOV      @ (R1)+, -(SP)    ;; SAVE @ (R1)+ FOR TYPEOUT
4498 022204 104402          TYPDC     ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
4499 022206 000402          BR       8$
4500 7$:      MOV      @ (R1)+, -(SP)          ;; SAVE @ (R1)+ FOR TYPEOUT
4501 022210 013146          TYPDS     ;; GO TYPE--DECIMAL ASCII WITH SIGN
4502 022212 104405 8$:      TST      (R1)              ;; IS THERE ANOTHER NUMBER?
4503 022214 005711          BEQ      9$                ;; BR IF NO
4504 022216 001403          TYPE     , 11$            ;; TYPE TWO(2) SPACES
4505 022220 104401 022240    BR       6$                ;; LOOP
4506 022224 000764
4507
4508 022226 012601 9$:      MOV      (SP)+, R1          ;; RESTORE R1
4509 022230 012600 10$:     MOV      (SP)+, R0          ;; RESTORE R0

```

```

4510 022232 104401 001171          TYPE      ,SCLRF          ;:'CARRIAGE RETURN' & 'LINE FEED'
4511 022236 000207          RTS      PC          ;:RETURN
4512 022240 020040      000      11$:      .ASCIIZ / /          ;:TWO(2) SPACES
4513          022244          .EVEN
4514          .SBTTL      BINARY TO OCTAL (ASCII) AND TYPE
4515
4516          ;*****
4517          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
4518          ;*OCTAL (ASCII) NUMBER AND TYPE IT.
4519          ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
4520          ;*CALL:
4521          ;*      MOV      NUM,-(SP)          ;:NUMBER TO BE TYPED
4522          ;*      TYPOS          ;:CALL FOR TYPEOUT
4523          ;*      .BYTE      N          ;:N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
4524          ;*      .BYTE      M          ;:M=1 OR 0
4525          ;*          ;:1=TYPE LEADING ZEROS
4526          ;*          ;:0=SUPPRESS LEADING ZEROS
4527
4528          ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
4529          ;*$TYPOS OR $TYPOC
4530          ;*CALL:
4531          ;*      MOV      NUM,-(SP)          ;:NUMBER TO BE TYPED
4532          ;*      TYPON          ;:CALL FOR TYPEOUT
4533
4534          ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
4535          ;*CALL:
4536          ;*      MOV      NUM,-(SP)          ;:NUMBER TO BE TYPED
4537          ;*      TYPOC          ;:CALL FOR TYPEOUT
4538
4539 022244 017646 000000          $TYPOS: MOV      @ (SP),-(SP)          ;:PICKUP THE MODE
4540 022250 116637 000001      022467      MOV      1(SP), $OFILL          ;:LOAD ZERO FILL SWITCH
4541 022256 112637 022471          MOV      (SP)+, $OMODE+1          ;:NUMBER OF DIGITS TO TYPE
4542 022262 062716 000002          ADD      #2, (SP)          ;:ADJUST RETURN ADDRESS
4543 022266 000406          BR      $TYPON
4544 022270 112737 000001      022467      $TYPOC: MOV      #1, $OFILL          ;:SET THE ZERO FILL SWITCH
4545 022276 112737 000006      022471          MOV      #6, $OMODE+1          ;:SET FOR SIX(6) DIGITS
4546 022304 112737 000005      022466      $TYPON: MOV      #5, $OCNT          ;:SET THE ITERATION COUNT
4547 022312 010346          MOV      R3, -(SP)          ;:SAVE R3
4548 022314 010446          MOV      R4, -(SP)          ;:SAVE R4
4549 022316 010546          MOV      R5, -(SP)          ;:SAVE R5
4550 022320 113704      022471          MOV      $OMODE+1, R4          ;:GET THE NUMBER OF DIGITS TO TYPE
4551 022324 005404          NEG      R4
4552 022326 062704 000006          ADD      #6, R4          ;:SUBTRACT IT FOR MAX. ALLOWED
4553 022332 110437 022470          MOV      R4, $OMODE          ;:SAVE IT FOR USE
4554 022336 113704 022467          MOV      $OFILL, R4          ;:GET THE ZERO FILL SWITCH
4555 022342 016605 000012          MOV      12(SP), R5          ;:PICKUP THE INPUT NUMBER
4556 022346 005003          CLR      R3          ;:CLEAR THE OUTPUT WORD
4557 022350 006105          1$:      ROL      R5          ;:ROTATE MSB INTO 'C'
4558 022352 000404          BR      3$          ;:GO DO MSB
4559 022354 006105          2$:      ROL      R5          ;:FORM THIS DIGIT
4560 022356 006105          ROL      R5
4561 022360 006105          ROL      R5
4562 022362 010503          MOV      R5, R3
4563 022364 006103          3$:      ROL      R3          ;:GET LSB OF THIS DIGIT
4564 022366 105337      022470          DECB      $OMODE          ;:TYPE THIS DIGIT?
4565 022372 100016          BPL      7$          ;:BR IF NO

```

```

4566 022374 042703 177770      BIC      #177770,R3      ;;GET RID OF JUNK
4567 022400 001002      BNE      4$      ;;TEST FOR 0
4568 022402 005704      TST      R4      ;;SUPPRESS THIS 0?
4569 022404 001403      BEQ      5$      ;;BR IF YES
4570 022406 005204      4$: INC      R4      ;;DON'T SUPPRESS ANYMORE 0'S
4571 022410 052703 000060      BIS      #'0,R3      ;;MAKE THIS DIGIT ASCII
4572 022414 052703 000040      5$: BIS      #'1,R3      ;;MAKE ASCII IF NOT ALREADY
4573 022420 110337 022464      MOV      R3,8$      ;;SAVE FOR TYPING
4574 022424 104401 022464      TYPE     8$      ;;GO TYPE THIS DIGIT
4575 022430 105337 022466      7$: DECB     $OCNT      ;;COUNT BY 1
4576 022434 003347      BGT      2$      ;;BR IF MORE TO DO
4577 022436 002402      BLT      6$      ;;BR IF DONE
4578 022440 005204      INC      R4      ;;INSURE LAST DIGIT ISN'T A BLANK
4579 022442 000744      BR       2$      ;;GO DO THE LAST DIGIT
4580 022444 012605      6$: MOV      (SP)+,R5      ;;RESTORE R5
4581 022446 012604      MOV      (SP)+,R4      ;;RESTORE R4
4582 022450 012603      MOV      (SP)+,R3      ;;RESTORE R3
4583 022452 016666 000002 000004      MOV      2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
4584 022460 012616      MOV      (SP)+,(SP)
4585 022462 000002      RTI      ;;RETURN
4586 022464 000      8$: .BYTE     0      ;;STORAGE FOR ASCII DIGIT
4587 022465 000      .BYTE     0      ;;TERMINATOR FOR TYPE ROUTINE
4588 022466 000      $OCNT: .BYTE     0      ;;OCTAL DIGIT COUNTER
4589 022467 000      $OFILL: .BYTE     0      ;;ZERO FILL SWITCH
4590 022470 000000      $OMODE: .WORD     0      ;;NUMBER OF DIGITS TO TYPE
4591      .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
4592
4593      ;*****
4594      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
4595      ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
4596      ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
4597      ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
4598      ;*REPLACED WITH SPACES.
4599      ;*CALL:
4600      ;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
4601      ;*      TYPDS      ;;GO TO THE ROUTINE
4602
4603      $TYPDS:
4604      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
4605      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
4606      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
4607      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
4608      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
4609      MOV      #20200,-(SP)      ;;SET BLANK SWITCH AND SIGN
4610      MOV      20(SP),R5      ;;GET THE INPUT NUMBER
4611      BPL      1$      ;;BR IF INPUT IS POS.
4612      NEG      R5      ;;MAKE THE BINARY NUMBER POS.
4613      MOV      #'-,1(SP)      ;;MAKE THE ASCII NUMBER NEG.
4614      1$: CLR      R0      ;;ZERO THE CONSTANTS INDEX
4615      MOV      #5DBLK,R3      ;;SETUP THE OUTPUT POINTER
4616      MOV      #'',(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
4617      2$: CLR      R2      ;;CLEAR THE BCD NUMBER
4618      MOV      $DTBL(R0),R1      ;;GET THE CONSTANT
4619      3$: SUB      R1,R5      ;;FORM THIS BCD DIGIT
4620      BLT      4$      ;;BR IF DONE
4621      INC      R2      ;;INCREASE THE BCD DIGIT BY 1

```

```

4622 022554 000774          BR      3$
4623 022556 060105      4$: ADD     R1,R5          ;;ADD BACK THE CONSTANT
4624 022560 005702          TST     R2          ;;CHECK IF BCD DIGIT=0
4625 022562 001002          BNE     5$          ;;FALL THROUGH IF 0
4626 022564 105716          TSTB    (SP)        ;;STILL DOING LEADING 0'S?
4627 022566 100407          BMI     7$          ;;BR IF YES
4628 022570 106316      5$: ASLB     (SP)        ;;MSD?
4629 022572 103003          BCC     6$          ;;BR IF NO
4630 022574 116663 000001 177777  MOVB   1(SP),-1(R3)  ;;YES--SET THE SIGN
4631 022602 052702 000060      6$: BIS     #'0,R2    ;;MAKE THE BCD DIGIT ASCII
4632 022606 052702 000040      7$: BIS     #' ,R2    ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
4633 022612 110223          MOVB   R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
4634 022614 005720          TST     (R0)+    ;;JUST INCREMENTING
4635 022616 020027 000010      CMP     R0,#10    ;;CHECK THE TABLE INDEX
4636 022622 002746          BLT     2$          ;;GO DO THE NEXT DIGIT
4637 022624 003002          BGT     8$          ;;GO TO EXIT
4638 022626 010502          MOV     R5,R2      ;;GET THE LSD
4639 022630 000764          BR      6$          ;;GO CHANGE TO ASCII
4640 022632 105726      8$: TSTB    (SP)+    ;;WAS THE LSD THE FIRST NON-ZERO?
4641 022634 100003          BPL     9$          ;;BR IF NO
4642 022636 116663 177777 177776  MOVB   -1(SP),-2(R3)  ;;YES--SET THE SIGN FOR TYPING
4643 022644 105013      9$: CLRB     (R3)        ;;SET THE TERMINATOR
4644 022646 012605          MOV     (SP)+,R5    ;;POP STACK INTO R5
4645 022650 012603          MOV     (SP)+,R3    ;;POP STACK INTO R3
4646 022652 012602          MOV     (SP)+,R2    ;;POP STACK INTO R2
4647 022654 012601          MOV     (SP)+,R1    ;;POP STACK INTO R1
4648 022656 012600          MOV     (SP)+,R0    ;;POP STACK INTO R0
4649 022660 104401 022706          TYPE    $DBLK    ;;NOW TYPE THE NUMBER
4650 022664 016666 000002 000004  MOV     2(SP),4(SP)  ;;ADJUST THE STACK
4651 022672 012616          MOV     (SP)+,(SP)
4652 022674 000002          RTI              ;;RETURN TO USER
4653 022676 023420      $DTBL: 10000.
4654 022700 001750          1000.
4655 022702 000144          100.
4656 022704 000012          10.
4657 022706 000004      $DBLK: .BLKW    4
4658          .SBTTL  POWER DOWN AND UP ROUTINES
4659
4660
4661          ;*****
4662          ;POWER DOWN ROUTINE
4663 $PWRDN: MOV     # $ILLUP,@#PWRVEC  ;;SET FOR FAST UP
4664          MOV     #340,@#PWRVEC+2  ;;PRIO:7
4665          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
4666          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
4667          MOV     R2,-(SP)          ;;PUSH R2 ON STACK
4668          MOV     R3,-(SP)          ;;PUSH R3 ON STACK
4669          MOV     R4,-(SP)          ;;PUSH R4 ON STACK
4670          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
4671          MOV     @SWR,-(SP)        ;;PUSH @SWR ON STACK
4672          MOV     SP,$SAVR6         ;;SAVE SP
4673          MOV     # $PWRUP,@#PWRVEC  ;;SET UP VECTOR
4674          HALT
4675          BR      .-2              ;;HANG UP
4676
4677          ;*****
4678          ;POWER UP ROUTINE

```

```

4678 022770 012737 023056 000024 $PWRUP: MOV    $SILLUP,@#PWRVEC ;;SET FOR FAST DOWN
4679 022776 013706 023062          MOV    $SAVR6,SP      ;;GET SP
4680 023002 005037 023062          CLR    $SAVR6        ;;WAIT LOOP FOR THE TTY
4681 023006 005237 023062 1$:      INC    $SAVR6        ;;WAIT FOR THE INC
4682 023012 001375          BNE    1$                ;;OF WORD
4683 023014 012677 156120          MOV    (SP)+,@SWR      ;;POP STACK INTO @SWR
4684 023020 012605          MOV    (SP)+,R5          ;;POP STACK INTO R5
4685 023022 012604          MOV    (SP)+,R4          ;;POP STACK INTO R4
4686 023024 012603          MOV    (SP)+,R3          ;;POP STACK INTO R3
4687 023026 012602          MOV    (SP)+,R2          ;;POP STACK INTO R2
4688 023030 012601          MOV    (SP)+,R1          ;;POP STACK INTO R1
4689 023032 012600          MOV    (SP)+,R0          ;;POP STACK INTO R0
4690 023034 012737 022716 000024  MOV    $PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
4691 023042 012737 000340 000026  MOV    #340,@#PWRVEC+2 ;;PRIO:7
4692 023050 104401          TYPE                    ;;REPORT THE POWER FAILURE
4693 023052 023064          $PWRMG: .WORD    $POWER      ;;POWER FAIL MESSAGE POINTER
4694 023054 000002          RTI
4695 023056 000000          $SILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
4696 023060 000776          BR      -2                ;; BEFORE THE POWER DOWN WAS COMPLETE
4697 023062 000000          $SAVR6: 0                    ;;PUT THE SP HERE
4698 023064 005015 047520 042527 $POWER: .ASCII <15><12>'POWER'
4699 023072 000122
4700          .EVEN
4701          .SBTTL TRAP DECODER
4702
4703          ;*****
4704          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
4705          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
4706          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
4707          ;*GO TO THAT ROUTINE.
4708
4709 023074 010046          $TRAP: MOV    R0,-(SP)          ;;SAVE R0
4710 023076 016600 000002          MOV    2(SP),R0        ;;GET TRAP ADDRESS
4711 023102 005740          TST    -(R0)              ;;BACKUP BY 2
4712 023104 111000          MOVB   (R0),R0          ;;GET RIGHT BYTE OF TRAP
4713 023106 006300          ASL    R0                ;;POSITION FOR INDEXING
4714 023110 016000 023130          MOV    $TRPAD(R0),R0      ;;INDEX TO TABLE
4715 023114 000200          RTS    R0                ;;GO TO ROUTINE
4716
4717
4718          ;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO
4719
4720 023116 011646          $TRAP2: MOV    (SP),-(SP)       ;;MOVE THE PC DOWN
4721 023120 016666 000004 000002  MOV    4(SP),2(SP)     ;;MOVE THE PSW DOWN
4722 023126 000002          RTI                    ;;RESTORE THE PSW
4723
4724          .SBTTL TRAP TABLE
4725
4726          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
4727          ;*BY THE 'TRAP' INSTRUCTION.
4728
4729          : ROUTINE
4730          :-----
4731 023130 023116          $TRPAD: .WORD    $TRAP2
4732 023132 021650          $TYPE  ;;CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
4733 023134 022270          $TYPOC ;;CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)

```

4734	023136	022244							
4735	023140	022304							
4736	023142	022472							
4737									
4738									
4739									

						\$TYPOS	::CALL=TYPOS	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
						\$TYPON	::CALL=TYPON	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
						\$TYPDS	::CALL=TYPDS	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)

4740 023144 003100 003466 003674 TSTADD: TST1,TST2,TST3

4741 023152 004070 004222 004462 TST4,TST5,TST6

4742 023160 004712 005466 006216 TST7,TST10,TST11

4743 023166 006430 006604 007032 TST12,TST13,TST14

4744 023174 007254 010024 010242 TST15,TST16,TST17

4745 023202 010410 011032 011336 TST20,TST21,TST22

4746 023210 011534 011606 TST23,TST24

4747 023214 042523 020124 052126

4748 023222 030466 020123 047524 STUPM: .ASCII /SET VT61S TO FULL DUPLEX, /<15><12>

4749 023230 020040 052506 046114

4750 023236 042040 050125 042514

4751 023244 026130 006440 012

4752 023251 122 046505 052117 .ASCII2 /REMOTE, PARITY MATCHED TO INTERFACE/<15><12> ;:♦♦C

4753 023256 026105 050040 051101

4754 023264 052111 020131 040515

4755 023272 041524 042510 020104

4756 023300 047524 044440 052116

4757 023306 051105 040506 042503

4758 023314 005015 000

4759 023317 015 040412 042104

4760 023324 042522 051523 051505

4761 023332 053440 052111 020110

4762 023340 042522 050123 047117

4763 023346 044523 042526 053040

4764 023354 033124 051461 040440

4765 023362 042522 006472 000012

4766 023370 005015 047516 053040

4767 023376 033124 020061 042522 NOVT: .ASCII2 <15><12>/NO VT61 RESPONDED TO ESCZ SEQ. AUTO RETRY IN 30 SEC./<15><12>

4768 023404 050123 047117 042504

4769 023412 020104 047524 042440

4770 023420 041523 020132 042523

4771 023426 027121 040440 052125

4772 023434 020117 042522 051124

4773 023442 020131 047111 031440

4774 023450 020060 042523 027103

4775 023456 005015 000

4776 023456 005015 000

4777 023456 005015 000

4778 023456 005015 000

4779 023456 005015 000

4780 023456 005015 000

4790	023461	015	042012	030514	DLERR: .ASCII2 <15><12>/DL11 FAILED AT ADDRESS/
4791	023466	020061	040506	046111	
4792	023474	042105	040440	020124	
4793	023502	042101	051104	051505	
4794	023510	000123			
4795					
4796	023512	040515	052516	046101	DMANA: .ASCII /MANUAL TEST SELECTED -/<15><12>
4797	023520	052040	051505	020124	
4798	023526	042523	042514	052103	
4799	023534	042105	026440	005015	
4800	023542	047105	042524	020122	.ASCII2 /ENTER ADDRESSES OF VT61S TO BE TESTED/<15><12>
4801	023550	042101	051104	051505	
4802	023556	042523	020123	043117	
4803	023564	053040	033124	051461	
4804	023572	052040	020117	042502	
4805	023600	052040	051505	042524	
4806	023606	006504	000012		
4807					
4808	023612	047105	042524	020122	DMANB: .ASCII2 /ENTER TESTS TO BE RUN/<15><12>
4809	023620	042524	052123	020123	
4810	023626	047524	041040	020105	
4811	023634	052522	006516	000012	
4812					
4813	023642	047101	042440	041523	EM1: .ASCII2 /AN ESC SEQ. TO THE VT61 FAILED - OCTAL EQUIV. IS:/<15><12>
4814	023650	051440	050505	020056	
4815	023656	047524	052040	042510	
4816	023664	053040	033124	020061	
4817	023672	043040	044501	042514	
4818	023700	020104	020055	041517	
4819	023706	040524	020114	050505	
4820	023714	044525	027126	044440	
4821	023722	035123	005015	000	
4822	023727	124	051505	021524	DH1: .ASCII2 /TEST# ERR PC BYTE 1+2 BYTE 3+4/<15><12>
4823	023734	020040	051105	020122	
4824	023742	041520	020040	054502	
4825	023750	042524	030440	031053	
4826	023756	041040	052131	020105	
4827	023764	025463	005464	000012	
4828					
4829	023772	042522	042503	053111	EM2: .ASCII2 /RECEIVE STATUS ERROR./<15><12>
4830	024000	020105	052123	052101	
4831	024006	051525	042440	051122	
4832	024014	051117	006456	000012	
4833	024022	042101	027104	020040	DH2: .ASCII2 /ADD. STAT. ERR.BITS CHAR./<15><12>
4834	024030	052123	052101	020056	
4835	024036	042440	051122	041056	
4836	024044	052111	020123	041440	
4837	024052	040510	027122	005015	
4838	024060	000			
4839					
4840	024061	123	043117	053524	EM3: .ASCII2 /SOFTWARE (VSTAT) STATUS ERROR./<15><12>
4841	024066	051101	020105	053050	
4842	024074	052123	052101	020051	
4843	024102	052123	052101	051525	
4844	024110	042440	051122	051117	
4845	024116	006456	000012		

4846	024122	050040	051501	021523	DH3:	.ASCIIZ / PASS#, TEST#, EXP.STAT, ACT.STAT/<15><12>
4847	024130	020054	052040	051505		
4848	024136	021524	020054	042440		
4849	024144	050130	051456	040524		
4850	024152	026124	020040	041501		
4851	024160	027124	052123	052101		
4852	024166	005015	000			
4853						
4854	024171	107	027104	042040	EM4:	.ASCIIZ /GD. DATA DOES NOT MATCH REC. DATA/<15><12>
4855	024176	052101	020101	047504		
4856	024204	051505	047040	052117		
4857	024212	046440	052101	044103		
4858	024220	051040	041505	020056		
4859	024226	040504	040524	005015		
4860	024234	000				
4861	024235	124	051505	021524	DH4:	.ASCIIZ /TEST# ,REC.CNT.,GD. DATA, REC. DATA/<15><12>
4862	024242	026040	042522	027103		
4863	024250	047103	027124	043454		
4864	024256	027104	042040	052101		
4865	024264	026101	051040	041505		
4866	024272	020056	040504	040524		
4867	024300	005015	000			
4868		024304				.EVEN
4869						
4870	024304	054502	042524	020123	EM5:	.ASCIIZ /BYTES EXPECTED DOES NOT EQUAL BYTES RECEIVED/<15><12>
4871	024312	054105	042520	052103		
4872	024320	042105	042040	042517		
4873	024326	020123	047516	020124		
4874	024334	050505	040525	020114		
4875	024342	054502	042524	020123		
4876	024350	042522	042503	053111		
4877	024356	042105	005015	000		
4878	024363	102	052131	051505	DH5:	.ASCIIZ /BYTES EXP., BYTES REC./<15><12>
4879	024370	042440	050130	026056		
4880	024376	041040	052131	051505		
4881	024404	051040	041505	006456		
4882	024412	000012				
4883						
4884	024414	052503	051522	051117	EM6:	.ASCIIZ /CURSOR POSITIONING ERROR/<15><12>
4885	024422	050040	051517	052111		
4886	024430	047511	044516	043516		
4887	024436	042440	051122	051117		
4888	024444	005015	000			
4889	024447	107	020104	044514	DH6:	.ASCIIZ /GD LINE GD COL. BD LINE BD COL/<15><12>
4890	024454	042516	020040	042107		
4891	024462	041440	046117	020056		
4892	024470	020040	042102	046040		
4893	024476	047111	020105	041040		
4894	024504	020104	047503	006514		
4895	024512	000012				
4896						
4897	024514	044504	042522	052103	EM7:	.ASCIIZ /DIRECT CURSOR ADDRESSING FAILURE/<15><12>
4898	024522	041440	051125	047523		
4899	024530	020122	042101	051104		
4900	024536	051505	044523	043516		
4901	024544	043040	044501	052514		

4902	024552	042522	005015	000	
4903	024557	120	051501	021523	DH7: .ASCIZ /PASS# TEST# ERROR PC /<15><12>
4904	024564	020040	042524	052123	
4905	024572	021440	020040	051105	
4906	024600	047522	020122	041520	
4907	024606	020040	006440	000012	
4908	024614	040520	051523	020043	DH10: .ASCIZ /PASS# TEST# BD.ROW BD.COL/<15><12>
4909	024622	052040	051505	021524	
4910	024630	020040	042102	051056	
4911	024636	053517	020040	042102	
4912	024644	041456	046117	005015	
4913	024652	000			
4914					
4915	024653	114	051501	020124	EM11: .ASCIZ /LAST TRANSMISSION TO VT61 CAUSED UNIT TO FAIL-HANG./<15><12>
4916	024660	051124	047101	046523	
4917	024666	051511	044523	047117	
4918	024674	052040	020117	052126	
4919	024702	030466	041440	052501	
4920	024710	042523	020104	047125	
4921	024716	052111	052040	020117	
4922	024724	040506	046111	044055	
4923	024732	047101	027107	005015	
4924	024740	000			
4925					
4926	024741	126	033124	020061	EM12: .ASCIZ /VT61 UNDER TEST FAILED- ERROR DATA FOLLOWS/<15><12>
4927	024746	047125	042504	020122	
4928	024754	042524	052123	043040	
4929	024762	044501	042514	026504	
4930	024770	042440	051122	051117	
4931	024776	042040	052101	020101	
4932	025004	047506	046114	053517	
4933	025012	006523	000012		
4934					
4935	025016	052126	030466	043040	EM10: .ASCIZ /VT61 FAILED SELF TEST FUNCTION/<15><12>
4936	025024	044501	042514	020104	
4937	025032	042523	043114	052040	
4938	025040	051505	020124	052506	
4939	025046	041516	044524	047117	
4940	025054	005015	000		
4941					
4942					
4943	025057	120	051501	021523	DH12: .ASCIZ /PASS#, TEST#, GD.CKSUM, BD.CKSUM/<15><12>
4944	025064	020054	052040	051505	
4945	025072	021524	020054	042107	
4946	025100	041456	051513	046525	
4947	025106	020054	042102	041456	
4948	025114	051513	046525	005015	
4949	025122	000			
4950					
4951	025123	124	051505	044524	DABRT: .ASCIZ /TESTING ABORTED-TOO MANY FATAL XMITs/<15><12>
4952	025130	043516	040440	047502	
4953	025136	052122	042105	052055	
4954	025144	047517	046440	047101	
4955	025152	020131	040506	040524	
4956	025160	020114	046530	052111	
4957	025166	006523	000012		

4958					
4959	025172	052126	030466	051040	EM13: .ASCIZ /VT61 RECEIVER CHECKSUM COMPARE ERROR/<15><12>
4960	025200	041505	044505	042526	
4961	025206	020122	044103	041505	
4962	025214	051513	046525	041440	
4963	025222	046517	040520	042522	
4964	025230	042440	051122	051117	
4965	025236	005015	000		
4966					
4967	025241	126	033124	020061	EM14: .ASCIZ /VT61 TRANSMITTER CHECKSUM COMPARE ERROR/<15><12>
4968	025246	051124	047101	046523	
4969	025254	052111	042524	020122	
4970	025262	044103	041505	051513	
4971	025270	046525	041440	046517	
4972	025276	040520	042522	042440	
4973	025304	051122	051117	005015	
4974	025312	000			
4975					
4976		025314			
4977	025314	047125	052111	052440	DVUNIT: .EVEN .ASCII /UNIT UNDER TEST /<15><12>
4978	025322	042116	051105	052040	
4979	025330	051505	020124	005015	
4980	025336	041522	051123	020040	.ASCIZ /RCSR VECT. IDENT/<15><12>
4981	025344	053040	041505	027124	
4982	025352	020040	044440	042504	
4983	025360	052116	005015	000	
4984	025365	040	041522	051123	DH11: .ASCIZ / RCSR VECT./<15><12>
4985	025372	020040	053040	041505	
4986	025400	027124	005015	000	
4987	025405	120	044522	052116	DPRTR: .ASCIZ /PRINTER IS ATTACHED/<15><12>
4988	025412	051105	044440	020123	
4989	025420	052101	040524	044103	
4990	025426	042105	005015	000	
4991	025433	103	050117	042511	DCOPYR: .ASCIZ /COPIER IS ATTACHED/<15><12>
4992	025440	020122	051511	040440	
4993	025446	052124	041501	042510	
4994	025454	006504	000012		
4995	025460	047530	043106	052040	EM15: .ASCIZ /XOFF TO VT61 FAILED TO HALT BLOCK XMIT/<15><12>
4996	025466	020117	052126	030466	
4997	025474	043040	044501	042514	
4998	025502	020104	047524	044040	
4999	025510	046101	020124	046102	
5000	025516	041517	020113	046530	
5001	025524	052111	005015	000	
5002	025531	130	047117	052040	EM16: .ASCIZ /XON TO VT61 FAILED TO RESTART BLOCK XMIT/<15><12>
5003	025536	020117	052126	030466	
5004	025544	043040	044501	042514	
5005	025552	020104	047524	051040	
5006	025560	051505	040524	052122	
5007	025566	041040	047514	045503	
5008	025574	054040	044515	006524	
5009	025602	000012			
5010	025604	047516	054040	047117	EM17: .ASCIZ /NO XON RECEIVED WITHIN 3 SEC. AFTER A RESET/<15><12>
5011	025612	051040	041505	044505	
5012	025620	042526	020104	044527	
5013	025626	044124	047111	031440	

5014	025634	051440	041505	020056	
5015	025642	043101	042524	020122	
5016	025650	020101	042522	042523	
5017	025656	006524	000012		
5018	025662	040514	052123	050040	EM20: .ASCIIZ /LAST PERIPHERAL OPERATION ABORTED/<15><12>
5019	025670	051105	050111	042510	
5020	025676	040522	020114	050117	
5021	025704	051105	052101	047511	
5022	025712	020116	041101	051117	
5023	025720	042524	006504	000012	
5024	025726	047503	046125	020104	EM21: .ASCIIZ /COULD NOT CLEAR LAST ABORT FLAG./<15><12>
5025	025734	047516	020124	046103	
5026	025742	040505	020122	040514	
5027	025750	052123	040440	047502	
5028	025756	052122	043040	040514	
5029	025764	027107	005015	000	
5030	025771	123	046517	047440	EM22: .ASCIIZ /SOM OR EOM NOT RECEIVED DURING MAINT. MODE TRANSMIT/<15><12>
5031	025776	020122	047505	020115	
5032	026004	047516	020124	042522	
5033	026012	042503	053111	042105	
5034	026020	042040	051125	047111	
5035	026026	020107	040515	047111	
5036	026034	027124	046440	042117	
5037	026042	020105	051124	047101	
5038	026050	046523	052111	005015	
5039	026056	000			
5040	026057	114	047111	020105	EM23: .ASCIIZ /LINE FEED OR CURSOR RIGHT ISSUED FROM ROW 23 DID NOT CAUSE SCREEN TO SC
5041	026064	042506	042105	047440	
5042	026072	020122	052503	051522	
5043	026100	051117	051040	043511	
5044	026106	052110	044440	051523	
5045	026114	042525	020104	051106	
5046	026122	046517	051040	053517	
5047	026130	031040	020063	044504	
5048	026136	020104	047516	020124	
5049	026144	040503	051525	020105	
5050	026152	041523	042522	047105	
5051	026160	052040	020117	041523	
5052	026166	047522	046114	005015	
5053	026174	000			
5054	026175	120	051501	020123	DH13: .ASCIIZ /PASS , TEST , VSTAT/<15><12>
5055	026202	020054	020040	042524	
5056	026210	052123	026040	020040	
5057	026216	053040	052123	052101	
5058	026224	005015	000		
5059	026227	120	051501	026123	DH14: .ASCIIZ /PASS, TEST, ERR PC, VSTAT/<15><12>
5060	026234	020040	052040	051505	
5061	026242	026124	020040	042440	
5062	026250	051122	050040	026103	
5063	026256	020040	053040	052123	
5064	026264	052101	005015	000	
5065					
5066	026271	105	041523	000040	DESC: .ASCIIZ /ESC /
5067					
5068					
5069					

5070	026276	042513	041131	040517	DKYBD: .ASCII /KEYBOARD TEST/<15><12>
5071	026304	042122	052040	051505	
5072	026312	006524	012		
5073	026315	113	054505	052123	.ASCII /KEYSTROKES ECHO:<15><12>
5074	026322	047522	042513	020123	
5075	026330	041505	047510	006472	
5076	026336	012			
5077	026337	101	042040	051511	.ASCII /A DISPLAY CHAR. = A DISPLAY CHAR.<15><12>
5078	026344	046120	054501	041440	
5079	026352	040510	027122	036440	
5080	026360	040440	042040	051511	
5081	026366	046120	054501	041440	
5082	026374	040510	027122	005015	
5083	026402	031463	036440	042440	.ASCII /33 = ESC<15><12>
5084	026410	041523	005015		
5085	026414	032461	036440	041440	.ASCII /15 = C-R<15><12>
5086	026422	051055	005015		
5087	026426	031061	036440	046040	.ASCII /12 = L-F<15><12>
5088	026434	043055	005015		
5089	026440	033460	036440	041040	.ASCII /07 = BELL<15><12>
5090	026446	046105	006514	012	
5091	026453	061	020060	020075	.ASCII /10 = TAB<15><12>
5092	026460	040524	006502	012	
5093	026465	116	047117	042055	.ASCIIZ /NON-DISPLAY CHAR.= OCTAL EQUIV<15><12>
5094	026472	051511	046120	054501	
5095	026500	041440	040510	027122	
5096	026506	020075	041517	040524	
5097	026514	020114	050505	044525	
5098	026522	006526	000012		
5099					
5100	026526	040524	020102	006	DTAB: .ASCIIZ /TAB /
5101	026533	103	051055	000040	DCR: .ASCIIZ /C-R /
5102	026540	026514	020106	000	DLF: .ASCIIZ /L-F /
5103	026545	102	046105	020114	DBELL: .ASCIIZ /BELL /
5104	026552	000			
5105					
5106	026553	114	047517	020120	DLOOP: .ASCII /LOOP TEST - LOOP COMMANDS AND DATA THRU<15><12>
5107	026560	042524	052123	026440	
5108	026566	046040	047517	020120	
5109	026574	047503	046515	047101	
5110	026602	051504	040440	042116	
5111	026610	042040	052101	020101	
5112	026616	044124	052522	005015	
5113	026624	047510	052123	041040	.ASCII /HOST BACK TO VT61 UNDER TEST. <15><12>
5114	026632	041501	020113	047524	
5115	026640	053040	033124	020061	
5116	026646	047125	042504	020122	
5117	026654	042524	052123	020056	
5118	026662	005015			
5119	026664	047503	052116	047522	DCNTZ: .ASCIIZ /CONTROL C EXITS TEST.<15><12>
5120	026672	020114	020103	042440	
5121	026700	044530	051524	052040	
5122	026706	051505	027124	005015	
5123	026714	000			
5124					
5125	026715	105	044530	020124	DEXT: .ASCIIZ /EXIT TEST./

5126	026722	042524	052123	000056		
5127						
5128	026730	051120	047111	042524	DPRNT: .ASCII /PRINTER TEST -/<15><12>	
5129	026736	020122	042524	052123		
5130	026744	026440	005015			
5131	026750	031461	020062	047503	.ASCII /132 COLUMNS OF A SLIDING PATTERN WILL BE/	
5132	026756	052514	047115	020123		
5133	026764	043117	040440	051440		
5134	026772	044514	044504	043516		
5135	027000	050040	052101	042524		
5136	027006	047122	053440	046111		
5137	027014	020114	042502			
5138	027020	047503	052116	047111	.ASCII /CONTINUOUSLY OUTPUTTED TO PRINTER/<15><12>	
5139	027026	052517	046123	020131		
5140	027034	052517	050124	052125		
5141	027042	042524	020104	047524		
5142	027050	050040	044522	052116		
5143	027056	051105	005015			
5144	027062	040503	027122	051040	.ASCII /CAR. RET. WILL START TEST. ANOTHER/<15><12>	::**C
5145	027070	052105	020056	044527		
5146	027076	046114	051440	040524		
5147	027104	052122	052040	051505		
5148	027112	027124	040440	047516		
5149	027120	044124	051105	005015		
5150	027126	040503	027122	051040	.ASCIIZ /CAR. RET. WILL CAUSE TEST EXIT./<15><12>	::**C
5151	027134	052105	020056	044527		
5152	027142	046114	041440	052501		
5153	027150	042523	052040	051505		
5154	027156	020124	054105	052111		
5155	027164	006456	000012			
5156	027170	040503	027122	051040	DCRST: .ASCIIZ /CAR. RET. TO START/<15><12>	
5157	027176	052105	020056	047524		
5158	027204	051440	040524	052122		
5159	027212	005015	000			
5160						
5161	027215	114	051501	020124	DEVERR: .ASCIIZ /LAST XMIT CAUSED VT61 HANG/<15><12>	
5162	027222	046530	052111	041440		
5163	027230	052501	042523	020104		
5164	027236	052126	030466	044040		
5165	027244	047101	006507	000012		
5166	027252	000077			QMRK: .ASCIIZ /?/	
5167	027254	051120	042117	041525	DKBD: .ASCII /PRODUCTION KEYBOARD TEST. 10 ERRORS CAUSES TEST EXIT./<15><12>	
5168	027262	044524	047117	045440		
5169	027270	054505	047502	051101		
5170	027276	020104	042524	052123		
5171	027304	020056	030061	042440		
5172	027312	051122	051117	020123		
5173	027320	040503	051525	051505		
5174	027326	052040	051505	020124		
5175	027334	054105	052111	006456		
5176	027342	012				
5177	027343	104	050105	042522	.ASCIIZ /DEPRESS KEYS FROM LEFT TO RIGHT/<15><12>	
5178	027350	051523	045440	054505		
5179	027356	020123	051106	046517		
5180	027364	046040	043105	020124		
5181	027372	047524	051040	043511		

5182	027400	052110	005015	000	
5183	027405	104	050105	042522	DLSHFT: .ASCIZ /DEPRESS LEFT SHIFT KEY AND THE 'A' KEY /<15><12>
5184	027412	051523	046040	043105	
5185	027420	020124	044123	043111	
5186	027426	020124	042513	020131	
5187	027434	047101	020104	044124	
5188	027442	020105	040442	020042	
5189	027450	042513	020131	005015	
5190	027456	000			
5191	027457	104	050105	042522	DTOP: .ASCIZ /DEPRESS KEYS IN TOP ROW/<15><12>
5192	027464	051523	045440	054505	
5193	027472	020123	047111	052040	
5194	027500	050117	051040	053517	
5195	027506	005015	000		
5196					
5197	027511	104	050105	042522	DRSHFT: .ASCIZ /DEPRESS RIGHT SHIFT KEY AND THE 'A' KEY /<15><12>
5198	027516	051523	051040	043511	
5199	027524	052110	051440	044510	
5200	027532	052106	045440	054505	
5201	027540	040440	042116	052040	
5202	027546	042510	021040	021101	
5203	027554	045440	054505	006440	
5204	027562	000012			
5205	027564	042504	051120	051505	DSEC: .ASCIZ /DEPRESS KEYS IN SECOND ROW/<15><12>
5206	027572	020123	042513	051531	
5207	027600	044440	020116	042523	
5208	027606	047503	042116	051040	
5209	027614	053517	005015	000	
5210					
5211	027621	104	050105	042522	DTHRD: .ASCIZ /DEPRESS KEYS IN THIRD ROW BEGINNING WITH 'A' /<15><12>
5212	027626	051523	045440	054505	
5213	027634	020123	047111	052040	
5214	027642	044510	042122	051040	
5215	027650	053517	041040	043505	
5216	027656	047111	044516	043516	
5217	027664	053440	052111	020110	
5218	027672	040447	006447	000012	
5219	027700	042504	051120	051505	DCONT: .ASCIZ /DEPRESS CONTROL KEY ,AND THE 'A' KEY /<15><12>
5220	027706	020123	047503	052116	
5221	027714	047522	020114	042513	
5222	027722	020131	040454	042116	
5223	027730	052040	042510	021040	
5224	027736	021101	045440	054505	
5225	027744	006440	000012		
5226	027750	042504	051120	051505	DBOT: .ASCIZ /DEPRESS KEYS IN FORTH ROW EXCEPT SHIFT KEYS/<15><12>
5227	027756	020123	042513	051531	
5228	027764	044440	020116	047506	
5229	027772	052122	020110	047522	
5230	030000	020127	054105	042503	
5231	030006	052120	051440	044510	
5232	030014	052106	045440	054505	
5233	030022	006523	000012		
5234	030026	042504	051120	051505	DSPCE: .ASCIZ /DEPRESS SPACE BAR/<15><12>
5235	030034	020123	050123	041501	
5236	030042	020105	040502	006522	
5237	030050	000012			

5238					
5239	030052	042504	051120	051505	DKPD: .ASCII /DEPRESS KEYPAD KEYS,LEFT TO RIGHT, TOP TO BOTTOM/<15><12>
5240	030060	020123	042513	050131	
5241	030066	042101	045440	054505	
5242	030074	026123	042514	052106	
5243	030102	052040	020117	044522	
5244	030110	044107	026124	052040	
5245	030116	050117	052040	020117	
5246	030124	047502	052124	046517	
5247	030132	005015	000		
5248					
5249	030135	113	054505	047502	DKBERR: .ASCII /KEYBOARD ERROR,KEY POSITION IN ROW SHOULD BE /
5250	030142	051101	020104	051105	
5251	030150	047522	026122	042513	
5252	030156	020131	047520	044523	
5253	030164	044524	047117	044440	
5254	030172	020116	047522	020127	
5255	030200	044123	052517	042114	
5256	030206	041040	020105		
5257	030212	020040	005015		KYSTRK: .ASCII / /<15><12>
5258	030216	041517	040524	020114	.ASCII /OCTAL GD, OCTAL BAD/<15><12>
5259	030224	042107	020054	041517	
5260	030232	040524	020114	040502	
5261	030240	006504	000012		
5262	030244	020040	020040	020040	DSPC6: .ASCII / /
5263	030252	000			
5264					
5265	030253	036	076	020	ROW1: .BYTE 36,76,20,13,32,12,54,44,14,41,71,57,63,64,3,114,0
5266	030256	013	032	012	
5267	030261	054	044	014	
5268	030264	041	071	057	
5269	030267	063	064	✓3	
5270	030272	114	000		
5271					
5272	030274	026	056	030	ROW2: .BYTE 26,56,30,73,52,22,55,34,24,31,51,77,62,61,2,0
5273	030277	073	052	022	
5274	030302	055	034	024	
5275	030305	031	051	077	
5276	030310	062	061	002	
5277	030313	000			
5278					
5279	030314	046	040	053	ROW3: .BYTE 46,40,53,23,72,42,45,74,11,21,47,27,66,0
5280	030317	023	072	042	
5281	030322	045	074	011	
5282	030325	021	047	027	
5283	030330	066	000		
5284					
5285	030332	016	070	060	ROW4: .BYTE 16,70,60,50,33,43,25,35,75,65,37,115,67,0
5286	030335	050	033	043	
5287	030340	025	035	075	
5288	030343	065	037	115	
5289	030346	067	000		
5290					
5291	030350	001	000		CNTRA: .BYTE 01,0
5292					
5293	030352	101	000		SHFTA: .BYTE 101,0

```
5294
5295 030354      015      000      SPCB:  .BYTE  15,0
5296
5297 030356      113      004      103  KYPD:  .BYTE  113,04,103,104,1,112,101,102,6,7,106,100,5
5298 030361      104      001      112
5299 030364      101      102      006
5300 030367      007      106      100
5301 030372      005
5302 030373      010      105      107      .BYTE  10,105,107,110,17,111,0
5303 030376      110      017      111
5304 030401      000
5305
5306
5307 030402 000500      RCRLB:  .EVEN
5308                                .BLKB  500      ;RECEIVE CIRCULAR BUFFER
5309 031102 000500      TCRLB:  .BLKB  500      ;TRANSMIT CIRCULAR BUFFER
5310 031602 000000      ABUFP:  .WORD  0
5311 031604 000062      ABBUF:  .BLKB  50.
5312 031666 000000      0
5313      000001      .END
```

ABBUF	031604	1482	3279	3281	5311#										
ABSXT	021032	4230#													
ABUFP	031602	1482*	1767	1799	1802	1814	1817	2046	2047	2054	2627	2630	2634*	3278*	
		3279	3281*	3282*	3633	3637	4127*	4129	4132	4133*	4199	4204	4227	5310#	
AESCO	015106	3350	3382#												
AESCP	015100	3348	3379#												
ALEXT	012460	2884	2888#												
ALWCNT	002202	1315#	1584	1855	2314	3204									
AOUT	013102	2940	2990#												
ASESC	014722	3285	3347#												
ASTRT	003100	1485	1498#	2564	2633	2713	2739	2776							
AUTO	002236	805	1339#												
AUTOA	002246	1341#	1345	1426	2872										
BASC3	004240	1730	1731#												
BBLUP	013614	3037	3040	3133#											
BDEXT	012450	2880	2886#												
BDEXTA	012446	2881	2885#												
BEL	001750	1106#	1504	2649											
BINOCT	017674	2665	3984#	4142	4156	4162									
BIT0	= 000001	770#													
BIT00	= 000001	760#	770	1474	2555	2968	3284								
BIT01	= 000002	759#	769	1477	1548	3284	3353								
BIT02	= 000004	758#	768	2919											
BIT03	= 000010	757#	767	3284	3349	3359									
BIT04	= 000020	756#	766												
BIT05	= 000040	755#	765												
BIT06	= 000100	754#	764												
BIT07	= 000200	753#	763	1894	3248										
BIT08	= 000400	752#	762	4310											
BIT09	= 001000	751#	761	4318	4376										
BIT1	= 000002	769#													
BIT10	= 002000	750#	4361												
BIT11	= 004000	749#	4325												
BIT12	= 010000	748#													
BIT13	= 020000	747#	4368												
BIT14	= 040000	746#	4296												
BIT15	= 100000	745#													
BIT2	= 000004	768#													
BIT3	= 000010	767#													
BIT4	= 000020	766#													
BIT5	= 000040	765#													
BIT6	= 000100	764#													
BIT7	= 000200	763#													
BIT8	= 000400	762#													
BIT9	= 001000	761#													
BLDADA	002320	1363#	1385												

[illegible]

[illegible]

DLVCNT	002234	1331#	4250*	4256*						
DMAIN	002002	1144#								
DMAA	023512	1358	4796#							
DMAAB	023612	1387	4808#							
DMPDCT	020472	4153#								
DOAROW	011642	2763#	2773	2780						
DPNT	002116	1237#								
DPRNT	026730	2689	5128#							
DPRTR	025405	1479	4987#							
DRECT	002010	1152#	1711	3509	3966					
DRSHFT	027511	2789	5197#							
DSCHNG=	100000	1058#								
DSEC	027564	2787	5205#							
DSMES	017600	2621	2690	2735	2759	2775	3960#	4234		
DSPCE	030026	2789	5234#							
DSPC6	030244	4159	5262#							
DSWR =	177570	691#	834	2829						
DTAB	026526	2655	5100#							
DTHRD	027621	2787	5211#							
DTOP	027457	2787	5191#							
DTTBL	011766	2763	2793#	4140						
DT0	001424	870	1002#							
DT1	001436	899	921	1004#						
DT2	001454	877	906	1011#						
DT3	001466	927	934	1013#						
DT4	001502	885	913	941	948	978	985	992	1017#	
DT5	001514	892	1019#							
DT6	001526	957	964	971	999	1021#				
DUNTST	023317	3051	4771#							
DVUNIT	025314	1457	4977#							
EAPNT	002026	1174#								
EEMP	002016	1162#	3382							
EINST	002032	1179#								
EMAIN	002000	1142#	3507							
EMTVEC=	000030	780#	2813*	2814*						
EM1	023642	868	4813#							
EM10	025016	919	4935#							
EM11	024653	925	4915#							
EM12	024741	932	4926#							
EM13	025172	939	4959#							
EM14	025241	946	4967#							
EM15	025460	955	4995#							
EM16	025531	962	5002#							
EM17	025604	969	5010#							
EM2	023772	875	4829#							
EM20	025662	976	5018#							
EM21	025726	983	5024#							
EM22	025771	990	5030#							
EM23	026057	997	5040#							
EM3	024061	883	4840#							
EM4	024171	890	4854#							
EM5	024304	897	4870#							
EM6	024414	904	4884#							
EM7	024514	911	4897#							
ENDPS	010702	2563	2565#							
ENDSEL	010670	2548	2556	2562#						

ENDTAB	001724	1086#	2910															
ENSRT	007024	2217	2224#															
EOM	= 000004	1304#	1611	1858	1993	2054	2317	3111	3174	3309	3445	3699	3970					
EOS	001774	1133#	2481															
EPI	= 001000	1043#	3319	4194	4202													
EPNT	002114	1235#	2696															
ERPL	002034	1181#																
ERRVEC	= 000004	773#	2827	2828*	2839*	4301	4302*	4304*	4307*									
ERSE	010260	2469	2472#															
ERSXT	010406	2488	2490	2495#														
ESAMB	015202	2637*	3284	3289*	3347	3349	3353*	3359*	3376*	3403#	3712*	4201*						
ESC	= 000400	1044#	2635	3290	4194	4202												
ESCN	002132	1270#	1556	1863	1987	2115	2149	2353	2480	2559	2697	3286	4204					
ESCO	002054	1201#	1262	1559	1709	1737	1792	1866	1989	1990	2355	2520	3497	3508				
		3510	3965															
ESCOI	= 002054	1262#	1652	1670	2151	2160	2441											
ESCP	002120	1251#	1263	1562	2212													
ESCP1	= 002120	1263#																
ESCV1	= 002040	1254#																
ESCZ	002126	1264	1266#	1607	1609	3540												
ESCZ1	= 002126	1264#	1865															
ESJEQ	002216	1321#	1520*	1522	1578	1579												
ESTEX	003454	1514	1587#															
ESTST	003116	1502	1504#															
EXINT	013354	3044	3050#															
EXIT3	004456	1766	1768	1773#														
EXMAIN	011332	2624	2672#	2691	2736													
EXMNT	003672	1629	1634#															
EXTST	021022	2631	2711	4217	4227#													
FADD	012542	2909#	2920															
FADD1	012560	2914#	2917															
FEXIT	011704	2765	2771	2774#														
FNDBT	021166	4253	4263#															
FRCECT	020770	4208	4210	4213#														
FTEXT	003460	1585	1589#															
FTLCNT	002200	1314#	1481*	1583*	1584	1855	1887*	2051*	2314	3204	3212*	3502*	3642*	3826*				
		4261*																
FTLEXT	016452	3683	3685#															
GCMD	003154	1512#	1553	1557	1560	1563	1566	1573										
GDAD	013070	2973	2987#															
GDCURP	006416	2109	2128#															
GDSCRL	007252	2259	2276	2278	2283#													
GDSTRK	020562	4135	4169#															
GETON	015630	2522	3500	3520#														
GNS	= ***** U	791	2849	4732	4733	4734	4735	4736										
GOTON	015672	3522	3528#															
GTCR	017772	2693	4018#	4021														
GTEXT	020104	4044	4046	4049#														
GTNUM	020014	1363	1391	4031#	4052													
HDFLG	017354	3513*	3792*	3848	3851*	3865#												
HT	= 000011	683#	4417	4458														
IABT	002022	1167#																
IDENT	002124	1261#	1450*	1452*	1468	1474	1477	1538	2555									
INAG	006660	2195#	2219															
INITA	013164	1343	1381	3011#														
INRPL	006622	2184	2186#															

[illegible]

NOROUT	021072	4228	4237#											
NORXT	016456	3684	3686#											
NOSHFT	017766	4007	4011#											
NOSOM	015274	3439	3443#											
NOVT	023370	3045	4778#											
NWLN	010042	2418	2420#											
OCTBIN	017750	1384	1406	4005#										
OFFLP	004562	1799#	1805											
ONE	002204	1316#												
ONLP	004640	1812#	1820											
ONOF A	004622	1803	1809#											
ONOF LP	004532	1794#	1818											
ONOF XT	004710	1813	1823#											
ONOF 61	004500	1786	1788#											
PABRT =	010000	1040#	3395	4096										
PATGN	005412	1852	1891	1936#										
PDKBD	011624	2755	2758#											
PIRQ =	177772	690#												
PIRQVE=	000240	784#												
PMULT	017500	2854*	2861*	2866*	3904	3916#								
POPIT	003264	1540#	1586											
POP2SP=	022626	1030#	2858	2865	2885	2916	2955	3211						
PRABRT=	000172	1259#	3393											
PRESC	002214	1320#	1507*	1511	1525	1556*	1559*	1562*	1576					
PRO =	000000	707#												
PR1 =	000040	708#												
PR2 =	000100	709#												
PR3 =	000140	710#												
PR4 =	000200	711#												
PR5 =	000240	712#												
PR6 =	000300	713#												
PR7 =	000340	714#												
PS =	177776	687#	688											
PSW =	177776	688#	3228*											
PTYPE	002232	1327#	1755	1999	2033	2330	2360	2537	2855*	2862*	2867*	3676	3831	
PUSH2S=	024646	1031#												
PWRVEC=	000024	779#	2817*	2818*	4662*	4663*	4672*	4678*	4690*	4691*				
QMRK	027252	2886	5166#											
QUST	020106	4038	4040	4050#										
RABT	002106	1223#												
RBBUF	015174	1443*	1707	1878	1880	2011	2015	2059	2063	2158	2167	2260	2277	2280*
		2382	2383	2399*	2432	2448	2473*	2483*	2489	2493	3312	3324	3400#	3649
		3653	3708	3732	3797	4192								
RBUFP	015200	1880*	1881	2372	2374	2382*	3312*	3322	3324*	3332*	3333*	3402#	3708*	3797
		3806	4188	4192*	4196	4199*	4200*	4206*	4207*	4211*				
RCRLB	030402	1441	1443	1660	1677	2119*	4185	4230	5307#					
RCUR	002046	1194#	1700											
RDCUR	002050	1197#	1653	1671	1701	2442	3361							
RDENA =	000100	1059#	1483	1505	1589	2565	2970	2978	3221	3244				
REBUF	015176	1441*	3322	3401#	4184*	4230*								
RECAD	014214	2964	3239#											
RECDN =	000200	1057#	3093	3188	3213	4018								
RECEX	013602	3098	3114	3118#										
RECEXA	013606	3112	3120#											
RECITT	016366	2103*	2195*	2263*	3654*	3663#	3713*							
RECTM	013424	3087#	3543											

[illegible]

STUPM	023214	2851	4756#											
SVER1	002164	1308#	2666	3556*	3587	3991*	3992*	3993*	4143*	4144	4146	4148*	4150*	4151
		4152	4157	4163										
SVER2	002166	1309#	3557*	3588	3996*	4143								
SWR	001140	834#	2807	2829*	2831	2837*	4296	4310	4312	4318	4325	4361	4368	4373
		4376	4670	4683*										
SWREG	000176	793#	2837											
SW0	= 000001	742#												
SW00	= 000001	732#	742											
SW01	= 000002	731#	741											
SW02	= 000004	730#	740											
SW03	= 000010	729#	739											
SW04	= 000020	728#	738											
SW05	= 000040	727#	737											
SW06	= 000100	726#	736											
SW07	= 000200	725#	735											
SW08	= 000400	724#	734											
SW09	= 001000	723#	733											
SW1	= 000002	741#												
SW10	= 002000	722#												
SW11	= 004000	721#												
SW12	= 010000	720#												
SW13	= 020000	719#												
SW14	= 040000	718#												
SW15	= 100000	717#												
SW2	= 000004	740#												
SW3	= 000010	739#												
SW4	= 000020	738#												
SW5	= 000040	737#												
SW6	= 000100	736#												
SW7	= 000200	735#												
SW8	= 000400	734#												
SW9	= 001000	733#												
TAB	001756	1112#	2653											
TBBUF	015502	1444*	1650	1667	1697	1731	1862	1984	2024	2090	2146	2187	2192	2197
		2246	2265	2321	2350	2392	2394	2421	2439	2475	2531	2699	2702	3454
		3466	3477	3482#	3709	3944	3963	4103	4185*	4231*				
		775#												
TBITVE-	000014	2911	2918#											
TBLCK	012572	1605*	1607*	1609*	1611*	1796*	1809*	1847*	1853*	1858*	2307*	2312*	2317*	2644*
TBUFP	015506	2646*	3448	3452	3454*	3456*	3461	3463	3464	3466*	3468*	3477*	3484#	3693*
		3695*	3699*	3709*										
TCOMB	= 000104	1067#	2971	2977	3243									
TCRLB	031102	1442	1444	1877	1900	1906	1908	2104	2110	2196	2209	2264	2280	2387
		3793	3816	3843	3845	3874	3877	4184	4231	5309#				
TCUCH	002064	1208#	2152	2163										
TDATA	007020	2209	2222#											
TDEV	012722	2932	2951#											
TEBUF	015504	1442*	3452	3464	3483#									
TENA	= 000100	1065#	1742	1997	2031	2328	2358	2535	2708	2947	2951	3246	3253	3301
		3478	3619	3673	3707	3791	3948	3971	4216					
TESC	013666	1527	1793	2521	2560	2698	3155#	3498	3511	3541				
TFUNCT	007012	2211	2221#											
TIMEXT	021170	4262	4264#											
TKVEC	= 000060	782#												
TMNAD	012424	1366	1373	2879#										

[illegible]

[illegible]

\$CMTAG	001100	814#	2804	2805	2813	2819	2820	2821						
\$CM3 =	000000	844#												
\$CRLF	001171	848#	1456	1473	2985	3058	4165	4371	4388	4423	4458	4466	4485	4490
		4510												
\$DBLK	022706	4615	4649	4657#										
\$DOAGN	011006	2583	2592	2598#										
\$DTBL	022676	4618	4653#											
\$ENDAD	010776	800	2594#	2845	4383									
\$ENDCT	010744	2585#	2819											
\$ENDMG	011015	2587	2602#											
\$ENULL	011012	2590	2601#											
\$EOP	010710	2575#												
\$EOPCT	010736	2582#	2586	2819*										
\$ERFLG	001103	817#	4285	4314	4316	4322*	4343	4358*	4388					
\$ERMAX	001115	823#	2822*	4316	4338*	4343								
\$ERROR	021502	2813	4357#											
\$ERRPC	001116	824#	1002	1013	4365*	4366*	4367	4388	4472					
\$ERRTB	001174	865#	4480											
\$ERRTY	022070	4370	4465#											
\$ERTTL	001112	821#	4364*	4388										
\$ESCAP	001162	845#	2821*	4337*	4379	4381	4388							
\$FILLC	001156	842#	4427	4458										
\$FILLS	001155	841#	4458											
\$GDADR	001120	825#	1011	1019	1021	1806*	1821*	1912*	3503*	3581*	3598*	3731*	3855*	
\$GDDAT	001124	827#	1002	1004	1011	1017	1019	1576*	1577*	1630*	1709*	1710*	1910*	1919*
		2014*	2062*	2115*	2116*	2160*	2161*	2166*	2212*	2434*	2450*	2480*	2481*	2524*
		3566*	3582*	3600*	3730*	3853*	4098*	4099*	4114*	4115*				
		2591#												
\$GET42	010766	4738												
\$GTSWR=	***** U	664												
\$HD =	000000	818#	4329*	4330	4332*	4342								
\$ICNT	001104	4662	4678	4695#										
\$ILLUP	023056	832#												
\$INTAG	001135	822#	4367*	4388	4469									
\$ITEMB	001114	849#	4388	4458										
\$LF	001172	819#	1502*	1602*	1648*	1694*	1730*	1786*	1839*	1976*	2088*	2144*	2184*	2243*
\$LPADR	001106	2302*	2418*	2469*	2515*	2617*	2687*	2731*	2755*	2823*	4320*	4335*	4340	4342
		820#	2824*	4320	4336*	4342	4378							
\$LPERR	001110	2841	4335	4373	4411									
\$MAIL =	***** U	4333	4342#											
\$MXCNT	021500	840#	4429	4458										
\$NULL	001154	1499#	1599#	1645#	1691#	1727#	1783#	1836#	1973#	2085#	2141#	2181#	2240#	

[illegible]

[illegible]

J 9
CZVTHCO DL11 VT61 EXER MACY11 30A(1052) 10-APR-80 15:42 PAGE 115
CZVTHC.P11 10-APR-80 15:32 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0113

.XMTAL= 000126	1202#	2026	2477	
.Y = 000131	1195#	2248	2267	2423

COMMEN	1#	785#													
ENDCOM	1#	785#													
ERROR	679#	1582	1632	1658	1663	1675	1681	1705	1712	1750	1770	1807	1822	1872	1888
	1913	1921	2016	2052	2064	2107	2118	2156	2164	2168	2207	2213	2258	2262	2275
	2279	2380	2397	2430	2436	2446	2452	2487	2491	2494	2526	3504	3568	3585	3736
	3850	3856	4101	4117	4260										
ESCAPE	1#	785#													
GETPRI	1#	785#													
GETSWR	1#	785#	2848#												
MULT	1#	785#													
NEWTST	1#	785#	1498	1599	1645	1691	1727	1783	1836	1973	2085	2141	2181	2240	2299
	2415	2466	2512	2614	2684	2728	2752								
POP	1#	785#	1450	1453	1536	1540	1587	3019	3026	3030	3087	3089	3133	3135	3136
	3144	3155	3186	3344	3537	3554	3559	3661	3725	3727	3862	3912	3997	4032	4246
	4248	4249	4644	4683	4684										
PUSH	1#	785#	1509	1521	1525	1664	1682	1713	1751	1753	1758	1760	1763	1771	1790
	1791	1792	1923	1998	2002	2004	2007	2017	2032	2036	2038	2041	2065	2120	2281
	2329	2333	2335	2338	2359	2363	2365	2368	2398	2437	2453	2517	2519	2520	2536
	2540	2542	2545	2557	2558	2559	2694	2696	2697	3035	3036	3038	3039	3090	3115
	3118	3120	3137	3146	3178	3192	3270	3495	3496	3497	3505	3507	3508	3509	3510
	3539	3542	3544	3590	3613	3657	3674	3678	3680	3737	3786	3829	3834	3836	3839
	3860	3901	3984	4264	4603	4664	4670								
REPORT	1#	785#													
SCOPE	680#	1500	1600	1646	1692	1728	1784	1837	1974	2086	2142	2182	2241	2300	2416
	2467	2513	2576	2615	2685	2729	2753								
SETPRI	1#	785#													
SETTRA	4724#	4733	4734	4735	4736										
SETUP	1#	785#	2802												
SKIP	1#	785#													
SLASH	1#	785#													
SPACE	785#														
STARS	1#	785#	796	809	850	1033	1035	1053	1055	1071	1073	1089	1091	1101	1103
	1113	1115	1136	1138	1169	1171	1184	1186	1226	1228	1230	1232	1245	1247	1299
	1302	1334	1337	1347	1354	1409	1415	1488	1489	1496	1499	1591	1597	1599	1635
	1642	1645	1685	1689	1691	1718	1725	1727	1775	1781	1783	1825	1834	1836	1959
	1971	1973	2075	2083	2085	2134	2139	2141	2171	2179	2181	2228</			

\$\$ESCA	1#	785#												
\$\$NEWT	1#	785#	1499	1599	1645	1691	1727	1783	1836	1973	2085	2141	2181	2240
	2415	2466	2512	2614	2684	2728	2752							2299
\$\$SET	4724#	4733	4734	4735	4736									
\$\$SKIP	1#	785#												
.EQUAT	1#	654#	675											
.HEADE	1#	654#												
.KT11	1#													
.SETUP	1#	654#	1334											
.SWRHI	1#	654#	664											
.SWRLO	654#	675#												
.\$ACT1	1#	654#	794											
.\$APT8	1#													
.\$APTH	1#													
.\$APTY	1#													
.\$ASTA	1#													
.\$CATC	1#	654#	785											
.\$CMTA	1#	654#	807											
.\$DB2D	1#													
.\$DB2O	1#													
.\$DIV	1#													
.\$EOP	1#	654#	2566											
.\$ERRO	1#	654#	4343											
.\$ERRT	1#	654#	4458											
.\$MULT	1#													
.\$POWE	1#	654#	4658											
.\$RAND	1#													
.\$RDDE	1#													
.\$RDOC	1#													
.\$READ	1#													
.\$R2AZ	1#													
.\$SAVE	1#	654#												
.\$SB2D	1#													
.\$SB2O	1#													
.\$SCOP	1#	654#	4280											
.\$SIZE	1#													
.\$SUPR	1#													
.\$TRAP	1#	654#	4701											
.\$TYP8	1#													
.\$TYPD	1#	654#	4591											
.\$TYPE	1#	654#	4388											
.\$TYPO	1#	654#	4514											
.\$40CA	1#													
.1170	1#													

. ABS. 031670 000

ERRORS DETECTED: 0

CZVTHC.BIN,CZVTHC.LST/CRF/SOL/NL:TOC=SYSMAC.SML,CZVTHC.P11
 RUN-TIME: 66 65 5 SECONDS
 RUN-TIME RATIO: 228/138=1.6
 CORE USED: 33K (65 PAGES)

CZVTMCO DL11 VT61 EXER MACV11 30A(1052) 10-APR-80 15:42 PAGE 119
CZVTMC.P11 10-APR-80 15:32 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0116