

PDP11

INTER TEST PROGRAM
MD-11-DZITA-D

EP DZITA D DL B
COPYRIGHT 1977
FICHE 1 OF 1

MAR 1977
digital
MADE IN USA

The microfiche card contains a grid of frames. The leftmost column contains frames with text, likely program code or test instructions. The subsequent columns contain frames with data, including numerical values and possibly graphical plots or waveforms. The frames are arranged in a regular grid pattern, typical of a microfiche card.

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZITA-D-D
PRODUCT NAME: INTERPROCESSOR TEST PROGRAM (ITEP)
PROGRAM DATE: JANUARY 1977
MAINTAINER: DIAGNOSTICS

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

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

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

COPYRIGHT (C) 1973, 1977 BY DIGITAL EQUIPMENT CORPORATION

1.0 ABSTRACT.

THIS PROGRAM IS DESIGNED AS A MAINTENANCE AID FOR FIELD SERVICE PERSONEL. IT WILL VERIFY THE PROPER OPERATION OF A COMPLETE COMMUNICATION LINK FROM ONE PDP-11 SYSTEM TO ANOTHER OR TO A COMMUNICATION TEST CENTER.

2.0 REQUIREMENTS.

2.1 EQUIPMENT

A. PDP-11 SYSTEM WITH AT LEAST 4K OF CORE.

2.2 STORAGE.

4K OF CORE

3.0 LOADING PROCEDURE

THIS PROGRAM AND ALL OVERLAYS ARE ASSEMBLED IN ABSOLUTE FORMATS. THE ABS LOADER IS USED TO LOAD THE PROGRAM AND OVERLAYS.

LOAD THE ITEP PROGRAM AND THE APPROPRIATE OVERLAY FOR THE TYPE OF INTERFACE YOU WISH TO TEST.

4.0 OPERATING PROCEDURES.

IF RUNNING ITEP ON AN LSI-11 ENVIRONMENT :

1. IF THE LINE CLOCK IS TO BE USED IT SHOULD BE ENABLED PRIOR TO PROGRAM EXECUTION.

A. TWO METHODS OF ENTERING PARAMETERS ARE PROVIDED

1. LOAD ADDRESS 200 AND START TO ENTER PARAMS FROM CONSOLE TTY, PROCEED TO SECTION B.
 2. LOAD ADDRESS 200 AND SET SWITCH REGISTER BIT 15 BEFORE STARTING TO ENTER PARAMS FROM CONSOLE SWITCHES, PROCEED TO SECTION C.
- *THE PROGRAM MAY BE RESTARTED AT LOC 204 (ONCE PARAMETERS HAVE ALREADY BEEN SELECTED)

B. CONSOLE DIALOGUE PARAMETER INPUT (CURRENT VALUES FOR PARAMETERS ARE FOUND IN OVERLAY)
DN11 AND DM11BB PARAMETERS ARE DISCUSSED IN SECT. 10.0 OF THIS LISTING.

1. THE PROGRAM WILL TYPEOUT THE NAME OF THE VARIABLE OVERLAY.
 - A. IF YOU WISH TO SETUP JUST THE INDICATED OVERLAY, TYPE A CARAGE RETURN
 - B. IF YOU WISH TO SETUP A DN11, TYPE IN DN.
 - C. IF YOU WISH TO SETUP A DM11BB, TYPE IN DMB.

IF DN OR DMB WAS TYPED IN STEP 1 ABOVE THEN THE BUS ADDRESS, VECTOR, ETC. REFERED TO IN STEPS 2 THRU 7, PERTAIN TO THE DN11 OR DM11BB.

2. THE PROGRAM WILL TYPE THE DEFAULT BUS ADDRESS OF THE INTERFACE UNDER TEST.
 - A. TYPE A CAR. RETURN TO USE DEFAULT BUS ADDRESS
 - B. TYPEIN ACTUAL BUS ADDRESS
3. THE PROGRAM WILL TYPE OUT THE DEFAULT VECTOR ADDRESS
 - A. TYPE A CAR. RETURN TO USE DEFAULT ADDRESS

- B. TYPEIN ACTUAL VECTOR ADDRESS
4. THE PROGRAM WILL TYPE OUT THE DEFAULT INTERFACE PRIORITY
NOTE: 200=PRIO 4, 240=PRIO 5, 300=PRIO 6, ETC.
A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
B. TYPEIN ACTUAL VALUE
 5. THE PROGRAM WILL TYPEOUT THE DEFAULT VALUE OF PARAM#1
IF REQUIRED BY THE OVERLAY. (SEE SECT. 10.0 IN OVERLAY LISTING FOR PARAMETER DESCRIPTION)
A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
B. TYPEIN ACTUAL VALUE
 6. THE PROGRAM WILL TYPEOUT THE DEFAULT VALUE OF PARAM#2
IF REQUIRED BY THE OVERLAY.
A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
B. ENTER ACTUAL VALUE
 7. THE PROGRAM WILL TYPEOUT THE DEFAULT VALUE OF PARAM#3
IF REQUIRED BY THE OVERLAY.
A. TYPE A CAR. RETURN TO USE DEFAULT VALUE
THE DN-11 WILL USE PARAM #3 AS THE # TO DIAL.
IF USING A MODEM WITHOUT AUTOMATIC HANDSHAKING,
THE NUMBER MUST TERMINATE WITH A
"END-OF-NUMBER" CHARACTER (:).
B. ENTER ACTUAL VALUE.
 8. THE PROGRAM WILL RETURN TO STEP B1 IF THIS SETUP
WAS FOR DN11 OR DM11BB.
 9. THE PROGRAM WILL REQUEST THAT SWITCH REGISTER BE SET.
A. SETUP SWITCH REGISTER AS SPECIFIED IN STEP D.
AND TYPE A CAR. RETURN.
- NOTE: IF ANY OF THE ABOVE ITEMS 2 THRU 7 WERE CHANGED BY ENTERING
NEW VALUES, THE NEW VALUE BECOMES THE DEFAULT VALUE FOR SUBSEQUENT
RESTARTS OF THE PROGRAM.

- C. MANUAL PARAMETER INPUT FROM SWITCH REGISTER
1. THE PROGRAM HALTS FOR ISR (INTERFACE SERVICE ROUTINE) SPECIFICATION
 SWR14=1 SETUP DM-11BB ISR
 SWR13=1 SETUP DN-11 ISR
 SWR=000000=SETUP VARIABLE ISR (OVERLAY) (NOT DN-11 OR DM11BB)
 SET APPROPRIATE SWITCHES AND HIT CONTINUE.
 2. THE FOLLOWING HALTS ARE REPEATED FOR EACH ISR SPECIFIED.
 SETUP SEQUENCE IS: DN11, DM11-BB THEN VARIABLE ISR. (FOR EACH ENTRY SET SWITCHES AND THEN HIT CONT.)
 - A. HALT FOR BUS ADDRESS OF INTERFACE
 - B. HALT FOR VECTOR ADDRESS OF INTERFACE
 - C. HALT FOR PRIORITY OF INTERFACE (200=PRIO 4, 240=PRIO 5, 300=PRIO 6, ETC.)
 - D. HALT FOR INTERFACE PARAM #1 (SEE SECT. 10.0 IN OVERLAY LISTING FOR PARAMETER DESCRIPTION)
 - E. HALT FOR INTERFACE PARAM #2 (DN11 AND DMBB PARAMETERS ARE DISCUSSED IN SECT. 10.0 OF THIS LISTING)
 - F. GO BACK TO STEP A IF THIS SETUP WAS FOR DN OR DMB.
 3. HALT FOR OPERATIONAL SWITCH SETTINGS. (SEE STEP D.)
 - A. PRESS CONTINUE TO START TESTING

THIS PROGRAM HAS BEEN MODIFIED TO RUN ON A PROCESSOR WITH OR WITHOUT A HARDWARE SWITCH REGISTER. WHEN FIRST EXECUTED THE PROGRAM TESTS THE EXISTENCE OF A HARDWARE SWITCH REGISTER. IF NOT FOUND A SOFTWARE SWITCH REGISTER LOCATION (SWREG=LOC. 176) IS DEFAULTED TO. IF THIS IS THE CASE, UPON EXECUTION THE CONTENTS OF THE SWREG ARE DUMPED IN OCTAL ON THE CONSOLE TTY AND ANY CHANGES ARE REQUESTED

(IE) SWR=XXXXXX NEW=

POSSIBLE RESPONSES ARE:

1. <CR> IF NO CHANGES ARE TO BE MADE
2. 6 DIGITS 0-7 TO REPRESENT IN OCTAL THE NEW SWITCH REGISTER VALUE ;LAST DIGIT FOLLOWED BY <CR>.
3. ↑U TO ALLOW REENTERING VALUE IF ERROR IS COMMITTED KEYING IN SWREG VALUE.

BUILT INTO THE PROGRAM IS THE ABILITY TO DYNAMICALLY CHANGE THE CONTENTS OF SWREG DURING PROGRAM EXECUTION. BY STRIKING ↑G (CNTRL G) ON CONSOLE TTY THE OPERATOR SETS A REQUEST FLAG TO CHANGE THE CONTENTS OF SWREG, WHICH IS PROCESSED IN KEY AREAS OF THE PROGRAM CODE (IE) ERROR ROUTINES, AFTER HALTS END OF PASS, AND OTHER APPLICABLE AREAS.

D. OPERATIONAL SWITCH SETTINGS.

SW15=1 HALT ON ERROR
SW14=1 SINGLE PASS
SW14 HAS NO EFFECT IF SW04=0
SW13=1 INHIBIT ERROR TIMEOUTS
SW12=1 INHIBIT ALL TIMEOUTS EXCEPT ERRORS
IF SW12=0 AND SW04=1 END PASS IS TYPED
AND TRANSMITTED/RECEIVED DATA IS TYPED.
SW11=1 USE PREVIOUSLY SPECIFIED DATA
SW10=1 DATA SELECT (WITH SW09)
SW09=1 DATA SELECT (WITH SW10)
00=1 GET DATA FROM OPERATOR
01=1 TEST MESSAGE #1 (\$A QUICK BROWN FOX)
10=1 TEST MESSAGE #2 (\$B NUMERICS)
11=1 TEST MESSAGE #3 (\$C COMTEST/QUICK BROWN FOX/NUMERICS)
SW08=1 TRANSMIT RECEIVED DATA (INTERNAL LOOPBACK MODE)
SW07=1 DO NOT TEST RECEIVED DATA
SW06=1 MONITOR TRANSMITTED DATA ON CONSOLE TTY.*
SW05=1 MONITOR RECEIVED DATA ON CONSOLE TTY.*
* IN MANY CASES, NOT ALL DATA WILL APPEAR ON THE CONSOLE
TTY. THIS IS ESPECIALLY TRUE WHEN THE COMM INTERFACE IS
RUNNING AT A FASTER BAUD THAN THE CONSOLE, BUT EVEN AT EQUAL
OR SLOWER BAUDS, ALL CHARACTERS MAY NOT APPEAR ON THE CONSOLE.

SW04=1 RETURN TO MONITOR FOR END PASS
WHEN SW04=0 PROGRAM LOOPS IN THE OVERLAY NEVER RETURNING TO THE MONITOR.
SW03=1 INTERNAL LOOPBACK MODE
SW02=1 EXTERNAL LOOPBACK MODE
SW01=1 ONE-WAY-IN MODE
SW00=1 ONE-WAY-OUT MODE

IF OPERATOR SPECIFIED DATA WAS INDICATED, THE PROGRAM WILL TYPE A
REQUEST FOR THE DATA. DATA MAY BE ENTERED AS ASCII CHARACTERS OR OCTAL CODE.
TYPE IN THE DATA TERMINATED WITH A CR. OCTAL CODE MAY BE ENTERED BY TYPING AN
↑(UP ARROW) FOLLOWED BY THE OCTAL CODE (IN THE RANGE 000 TO 377)
SEPERATED BY SPACES AND TERMINATED BY ↑(UP ARROW).
I.E. ABCD↑ 000 123 377↑ EFG (CAR.RETURN)

A TYPICAL SWITCH SETTING FOR HALF-DUPLEX=003150 THIS SETTING USES
INTERNAL LOOPBACK MODE, LOOPS IN OVERLAY, MONITORS TRANSMITTED AND RECEIVED
DATA ON THE CONSOLE TTY, AND TESTS RECEIVED DATA USING TEST MESSAGE #3.

A TYPICAL SWITCH SETTING FOR FULL-DUPLEX=003144 THIS SETTING
IS THE SAME AS ABOVE EXCEPT IT USES THE EXTERNAL LOOPBACK MODE.

ALL STANDARD MESSAGES(TEST MESSAGES 1-3) ARE PRECEDED BY 2 FILL CHARACTERS(177),
AND ARE FOLLOWED BY A CR(015), LF(012),RECEIVE TERMINATING CHARACTER(001),
4 FILLS(177), AND A TRANSMIT TERMINATING CHARACTER(000). DURING TRANSMISSION,
WHEN A 000 CHARACTER IS SEEN THE TRANSMISSION IS STOPPED.DURING RECEPTION,
WHEN A 001 CHARACTER IS RECEIVED, THE RECEIVER IS SHUT OFF.
IF THE MESSAGE WAS INPUTED BY THE OPERATER, THE TERMINATING CHARACTERS ARE ADDED.

TEST MODES

INTERNAL LOOPBACK MODE

1. THE OVERLAY WAITS TO RECEIVE A MESSAGE (TERMINATED BY <001>)
2. VERIFIES THE DATA AGAINST THE DATA SELECTED BY SW09 AND SW10 (SW7=0)
3. TRANSMIT THE DATA SELECTED BY SW09 AND SW10 (SW8=0) OR TRANSMIT THE RECEIVED DATA (SW8=1)
4. RETURNS TO MONITOR FOR "END PASS" (SW4=1) OR GO TO STEP 1. (SW4=0)

EXTERNAL LOOPBACK MODE

1. THE OVERLAY SETS REQUEST TO SEND
2. WAIT FOR CLEAR TO SEND
3. TRANSMITS THE SELECTED DATA
4. RESETS REQUEST TO SEND
5. WAIT FOR MESSAGE TO BE RECEIVED
6. VERIFIES THE DATA (SW07=0)
7. RETURNS TO MONITOR FOR "END PASS". (SW04=1) OR GO TO STEP 1 (SW04=0)

ONE-WAY-IN MODE

1. THE OVERLAY WAITS FOR MESSAGE TO BE RECEIVED.
2. VERIFIES THE DATA (SW07=0)
3. RETURNS TO MONITOR FOR "END PASS" (SW04=1) OR GO TO STEP 1 (SW04=0)

ONE-WAY-OUT MODE

1. THE OVERLAY SETS REQUEST TO SEND
2. WAITS FOR CLEAR TO SEND
3. TRANSMITS SELECTED DATA
4. RETURNS TO MONITOR FOR "END PASS". (SW04=1) OR GO TO STEP 1 (SW04=0)

- E. THE OVERLAY IS THEN ENTERED AND A CONNECTION ESTABLISHED EITHER MANUALLY OR AUTOMATICALLY.

IF ONE-WAY-IN OR INTERNAL LOOPBACK MODES ARE SELECTED.
 THE OVERLAY WILL SET DATA TERMINAL READY AND WAIT FOR DATA.

IF ONE-WAY-OUT OR EXTERNAL LOOPBACK MODES WERE SELECTED.
 THE OVERLAY WILL SET DATA TERMINAL READY AND REQUEST TO SEND.
 THE OVERLAY WILL THEN WAIT FOR CLEAR TO SEND BEFORE ATTEMPTING TO TRANSMIT DATA.

F. IF SW04=0 THE OVERLAY WILL CONTINUE TO TRANSMIT/RECEIVE DATA.

IF SW04=1 THE OVERLAY WILL RETURN TO THE MONITOR AND TYPE "END PASS".

IF BOTH SW04=1 AND SW14=1, THE PROGRAM WILL REQUEST NEW INTERFACE PARAMS AFTER ONE PASS OF THE SELECTED TEST MODE.

TEST EXECUTION MAY BE INTERRUPTED BY TYPING THE FOLLOWING CHARACTERS ON THE CONSOLE TTY.

LINE FEED = RESTART PROGRAM AT LOCATION 200.

QUESTION MARK = PRINTOUT FIRST 8 WORDS OF INPUT BUFFER. (ASCII)

THEN TYPE EITHER:

*WXXXXXX TO PRINTOUT THE 8 WORDS AT LOC XXXXXX.

*BXXXXXX TO PRINTOUT THE 16 BYTES AFTER LOC XXXXXX.

*C TO CONTINUE

PROGRAM MUST BE RESTARTED AT 200 AFTER PRINTING.

CARRIAGE RETURN = RESTART AT REQUEST FOR NEW OPERATIONAL SWITCHES.

5.0 PROGRAM AND/OR OPERATOR ACTION

IF THE OPERATOR WISHES TO MANUALLY EXAMINE THE TRANSMIT OR RECEIVE BUFFERS, DO THE FOLLOWING: TO FIND THE STARTING ADDRESS OF THE RECEIVE BUFFER, LOAD ADDRESS 11020 AND EXAMINE. TO FIND THE STARTING ADDRESS OF THE TRANSMIT BUFFER, LOAD ADDRESS 11022 AND EXAMINE.

5.1 NORMAL HALTS SEE SECTION 4.

6.0 ERRORS

6.1 ERROR REPORTING

THE ONLY ERROR REPORT FROM THE CONTROL PROGRAM OCCURS IF THE INTERFACE SPECIFIED IS NOT LOADED.

THE ERROR REPORTS FROM THE VARIOUS INTERFACE SERVICE ROUTINES ARE AS DEFINED IN THEIR DOCUMENTS

7.0 RESTRICTIONS

THE OPERATION OF THIS PROGRAM REQUIRES COORDINATION BETWEEN THE OPERATOR AND THE OPERATOR OF ANOTHER PDP-11 SYSTEM UNLESS ONE OF THE SYSTEMS IS ALWAYS OPERATING IN A FIXED MODE. THE FOLLOWING TABLE LISTS THE VALID COMBINATIONS:

NOTE: ONLY ONE MODE MAY BE SELECTED AT A TIME.

CPU #1	CPU #2
ONE-WAY-OUT	ONE-WAY-IN
ONE-WAY-IN	ONE-WAY-OUT
EXTERNAL-LOOPBACK	INTERNAL-LOOPBACK
INTERNAL-LOOPBACK	EXTERNAL-LOOPBACK
EXTERNAL-LOOPBACK	EXTERNAL-LOOPBACK (FULL-DUPLEX)

WHEN THE COMMUNICATION LINK INVOLVES MODEMS THE FOLLOWING RESTRICTIONS APPLY:

IF RUNNING IN FULL DUPLEX MODE BOTH SYSTEMS MUST BE IN EXTERNAL LOOP BACK MODE.

BOTH SYSTEMS SHOULD BE RUNNING IDENTICAL ROUTINES.

EXAMPLE:
SWITCHES 14,13,7,4 SHOULD BE THE SAME ON BOTH CPU'S

8.0 MISCELLANEOUS

ITEP WAS CHECKED OUT USING THE FOLLOWING BELL TELEPHONE MODEMS.
201A (HALF-DUPLEX SYNCHRONOUS 2000 BAUD)
202C (HALF-DUPLEX ASYNCHRONOUS 1200 BAUD)
103A (FULL-DUPLEX ASYNCHRONOUS 110 BAUD)

9.0 PROGRAM DESCRIPTION

THE INTERPROCESSOR TEST PROGRAM (ITP) PROVIDES THE LINKAGE BETWEEN THE OPERATOR AND THE VARIOUS INTERFACE SERVICE ROUTINES (OVERLAY) WHICH PERFORM THE ACTUAL DATA MOVEMENT AND VERIFICATION TO AND FROM THE COMMUNICATION LINK. IN ADDITION, ITP CONTAINS VARIOUS INTERRUPT AND SUB ROUTINES WHICH ARE USED BY THE OVERLAY'S.

9.1 TRAP CATCHER

THIS IS A SERIES OF JUMP AND HALT INSTRUCTIONS PLACED IN ALL UNUSED VECTORS TO CATCH UNEXPECTED INTERRUPTS.

9.2 SWITCH REGISTER INPUT ROUTINE (MANIN:)

THIS ROUTINE IS ENTERED ONLY WHEN SWITCH 15 IS SET WHEN PROGRAM IS STARTED AT LOCATION 200. IT ACCEPTS PARAMETERS FOR THE ISR'S FROM THE CONSOLE SWITCHES AT A SERIES OF HALTS. AS SPECIFIED IN OPERATING INSTRUCTIONS.

9.3 PARAMETER INPUT ROUTINE (GETIT:)

THIS ROUTINE SOLICITS PARAMETERS FROM THE OPERATOR ON THE CONSOLE DEVICE AND PLACES THEM IN THE SPECIFIED ISR'S PARAMETER TABLE.
NOT USED OPTIONAL PARAMETER WORDS ARE INDICATED BY THE PRESENCE OF A NEGATIVE VALUE IN THE ISR'S TABLE. THIS SECTION OF CODE UTILIZES SUB/ROUTINE 'GETANY' WHICH PRINTS OUT THE WORD POINTED TO BY THE ADDRESS IN REGISTER 0.
IT THEN INPUTS A WORD OR CARRIAGE RETURN FROM THE OPERATOR. IF ONLY A CARRIAGE RETURN IS TYPED, THE PARAMETER IS LEFT AS IT IS, OTHERWISE IT IS REPLACED BY THE OPERATORS TYPE IN AND THE POINTER IN REGISTER 0 IS INCREMENTED TO THE NEXT WORD.

9.4 TTY INTERRUPT (TTYINT:)

THE TTY INTERRUPT IS USED TO INTERRUPT THE EXECUTION OF A TEST IN ORDER TO RESTART (TYPE A LINE FEED) OR TO SPECIFY NEW OPERATIONAL SWITCHES (TYPE A CARRIAGE RETURN)

9.5 SET SWITCH OPTIONS (SWRSET:)

THE PROGRAM WILL HALT (MANUAL PARAMETER ENTRY) OR WAIT FOR A CARRIAGE RETURN (TTY CONTROL) AT THIS POINT TO PERMIT THE OPERATOR TO SETUP THE OPERATIONAL SWITCH SETTINGS. THE TEST MODE(SW00-SW03) AND TEST DATA(SW09-SW11) MAY BE CHANGED ONLY AT THIS POINT. ALL OTHER SWITCHES MAY BE CHANGED WHILE A TEST IS RUNNING. IF NEW VARIABLE DATA IS SPECIFIED, THIS ROUTINE WILL REQUEST THAT THE DATA BE ENTERED AND UTILIZES THE 'GETSTR' SUB/ROUTINE TO INPUT THE DATA FROM THE OPERATOR.

9.6 SETUP TIMER (SUTIME:)

THE PROGRAM LOOKS FOR AND UTILIZES EITHER THE LINE CLOCK OR REAL TIME CLOCK IF EITHER IS PRESENT ON THE SYSTEM. A BUS ERROR(NO RESPONSE) IS USED TO INDICATE THE ABSENCE OF A CLOCK. IF NEITHER EXISTS, THE PROGRAM WILL STILL RUN BUT IS SUBJECT TO WAITING IN UNENDING LOOPS.

9.7 THE INTERFACE SERVICE ROUTINES (ISR'S) ARE ENTERED AT THIS POINT.

9.8 END OF PASS (\$EOP:)

THIS SECTION OF CODE WILL PRINT "END OF PASS XXXXXX" AND THEN SENSE FOR SW14. IF SWITCH 14 IS RESET THE OVERLAY'S ARE REENTERED. IF SWITCH 14 IS SET THE PROGRAM CHECKS TO SEE IF IT WAS LOADED BY A MONITOR (LOCATION 42 NOT EQUAL 0) AND IF IT WAS, CONTROL IS RETURNED TO THE MONITOR. OTHERWISE THE PROGRAM REQUESTS NEW PARAMETERS.

9.10 HALT HANDLER (\$HLT:)

THIS ROUTINE IS USED TO SENSE THE OPERATIONAL SWITCHES AND PROVIDE ERROR CONTROL. IT WILL PRINTOUT THE ADDRESS OF THE ERROR HLT IF SWITCH 13 (DELETE ERROR TYPEOUTS) IS DOWN (NOT SET)

- 9.11 READ A CHARACTER ROUTINE (\$READC:)
THIS ROUTINE GETS A CHARACTER FROM THE TTY AND PLACES IT ON THE STACK
- 9.12 READ A STRING ROUTINE (\$READS)
THIS ROUTINE GETS A STRING OF CHARACTERS FROM THE TTY AND PLACES THEM IN A BUFFER SPECIFIED BY THE ADDRESS FOLLOWING THE SUB/ROUTINE CALL.
THE ROUTINE WILL ALSO ACCEPT OCTALLY REPRESENTED CHARACTERS WHEN THEY ARE PRECEDED AND FOLLOWED BY UP ARROWS, AND SPERATED BY SPACES OR COMMAS.
- 9.13 OCTAL INPUT ROUTINE(\$ACCEPT:)
THIS ROUTINE READS AN OCTALLY REPRESENTED WORD FROM THE TTY AND PLACES IT IN THE LOCATION INDICATED BY THE ADDRESS FOLLOWING THE SUB/ROUTINE CALL.
- 9.14 CLOCK INTERRUPT ROUTINE (TIMER:)
THIS ROUTINE IS ENTERED ON INTERUPTS FROM EITHER THE LINE CLOCK OR REAL TIME CLOCK EVERY 16 MILLISECONDS IF EITHER IS PRESENT.
IT WILL INCREMENT LOCATION 'TIME:' IN THE OVERLAY'S PARAMETER TABLE EVERY SECOND.
- 9.15 BINARY TO OCTAL ROUTINE (\$B2O16)
THIS ROUTINE WILL PRINTOUT THE OCTAL REPRESENTATION OF A WORD ON THE STACK.
- 9.16 POWER DOWN ROUTINE (\$PWRDN:)
THIS ROUTINE SAVES THE STATUS OF THE MACHINE WHEN POWER IS LOST.
- 9.17 POWER UP ROUTINE (\$PWRUP:)
THIS ROUTINE RESTORES THE STATE OF THE MACHINE WHEN POWER IS RESTORED AND RESTARTS AT ADDRESS 200.
- 9.18 VARIABLE INTERFACE SERVICE ROUTINE (VISR:)
THESE LOCATIONS ARE RESERVED FOR AND WILL BE OVERLAID BY THE VARIABLE ISR'S.
THE FIRST 2 WORDS CONTAIN A 3 CHARACTER ISR NEMONIC FOLLOWED BY A ZERO CHARACTER.
THE NEXT 3 WORDS CONTAIN THE BUS ADDRESS, VECTOR ADDRESS AND PRIORITY.
THE NEXT 2 WORDS MAY CONTAIN OPTIONAL PARAMETERS. THEY WILL CONTAIN ALL ONES IF THEY ARE NOT REQUIRED
THE NEXT WORD MAY CONTAIN THE ADDRESS OF AN INPUT BUFFER IF THE ISR REQUIRES AN ASCII PARAMETER. IT WILL CONTAIN

ALL ONES IF THE PARAMETER IS NOT REQUIRED.
LOCATION 'CLOCK:' WILL BE INCREMENTED EVERY SECOND WHILE
THE TEST IS BEING RUN IF THERE IS A LINE CLOCK OR REAL TIME CLOCK
ON THE SYSTEM. IT MAY BE USED AS A ELAPSED TIMER BY THE ISR.

10.0 PARAMETERS FOR THE DM11BB AND THE DN11

10.1 DM11BB PARAMETERS

PARAM#1 IS LOADED INTO THE CONTROL AND STATUS REGISTER OF THE DM11BB
TO SELECT THE LINE NUMBER IN OCTAL (BITS 0-3). ALL OTHER BITS MUST BE 0'S.
THIS IS THE ONLY PARAMETER USED BY THE DM11BB.

10.2 DN11 PARAMETERS

ONLY PARAM#3 IS USED BY THE DN11, IT CONTAINS THE NUMBER THE DN WILL DIAL.

NO1

MAINDEC-11-DZITA-D INTERPROCESSOR TEST PROGRAM MACY11 27(1006) 01-DEC-76 11:01 PAGE 13
DZITAD.F11 01-DEC-76 11:00

SEQ 0013

000000
000001
000002
000003
000004
000005
000006
000007
000000
000006
000007

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

000000
000040
000100
000140
000200
000240

:BASIC DEFINITIONS
:*****
:INITIAL ADDRESS OF THE STACK POINTER
STACK= 1070

001070

:*****
.EQUIV EMT,HLT

:BASIC DEFINITION OF ERROR CALL

:REGISTER DEFINITION

R0= %0
R1= %1
R2= %2
R3= %3
R4= %4
R5= %5
R6= %6
R7= %7
MODE= %0
R6=SP
R7=PC

:GENERAL REGISTER
:GENERAL REGISTER
:GENERAL REGISTER
:GENERAL REGISTER
:GENERAL REGISTER
:GENERAL REGISTER
:GENERAL REGISTER
:GENERAL REGISTER

:SWITCH DEFINITION

SW15= 100000
SW14= 40000
SW13= 20000
SW12= 10000
SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
SW07= 200
SW06= 100
SW05= 40
SW04= 20
SW03= 10
SW02= 4
SW01= 2
SW00= 1

.EQUIV SW09,SW9
.EQUIV SW08,SW8
.EQUIV SW07,SW7
.EQUIV SW06,SW6
.EQUIV SW05,SW5
.EQUIV SW04,SW4
.EQUIV SW03,SW3
.EQUIV SW02,SW2
.EQUIV SW01,SW1
.EQUIV SW00,SW0

PRTY0= 0
PRTY1= 40
PRTY2= 100
PRTY3= 140
PRTY4= 200
PRTY5= 240

000300
000340

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

000004
000010
000014
000014
000014
000020
000024
000030
000034

000000

000024
000026
000030
000032
000034
000036

PRTY6= 300
PRTY7= 340

:MISCELLANEOUS BIT ASSIGNMENT
BIT15= 100000
BIT14= 40000
BIT13= 20000
BIT12= 10000
BIT11= 4000
BIT10= 2000
BIT09= 1000
BIT08= 400
BIT07= 200
BIT06= 100
BIT05= 40
BIT04= 20
BIT03= 10
BIT02= 4
BIT01= 2
BIT00= 1
.EQUIV BIT09,BIT9
.EQUIV BIT08,BIT8
.EQUIV BIT07,BIT7
.EQUIV BIT06,BIT6
.EQUIV BIT05,BIT5
.EQUIV BIT04,BIT4
.EQUIV BIT03,BIT3
.EQUIV BIT02,BIT2
.EQUIV BIT01,BIT1
.EQUIV BIT00,BIT0

:VECTOR ADDRESSES
ERRVEC= 4
RESVEC= 10
TBITVEC= 14
TRTVEC= 14
BPTVEC= 14
IOTVEC= 20
PWRVEC= 24
EMTVEC= 30
TRAPVEC= 34
.EQUIV R4,CSR
.EQUIV R4,RCSR

.=0
:TRAP CATCHER IN UNUSED LOCATIONS FROM 0 - 776
:LOCATION 0 WILL CATCH IMPROPERLY LOADED VECTORS
.=24
\$PWRDN
340
\$HLT
340
\$TRAP
340

647		000100							
648	000100	006312							
649	000102	000340							
650									
651		000174							
652	000174	000000							
653	000176	000000							
654									
655		000200							
656									
657	000200	000137	003254		JMP	Q#BEGIN			:JUMP TO STARTING ADDRESS OF PROGRAM
658	000204	000137	004116		JMP	Q#SWPRT			:RESTART AT 204. DO THE RESTART.

DISPREG: 0
SWREG: 0

.=100
TIMER
340

.=174

.=200


```

659                                     ;*****
660      001100                          . =1100
661
662                                     ;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
663                                     ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
664                                     ;NOTE1: NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
665                                     ;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
666
667                                     ;CALL:
668                                     ;1) USING A TRAP INSTRUCTION
669                                     ;      TYPE      ,MESADR      ;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
670                                     ;OR
671                                     ;      TYPE
672                                     ;      MESADR
673
674      001100      010046      $TYPE:  MOV      RO, -(SP)      ;SAVE RO
675      001102      017600      000002      MOV      @2(SP), RO      ;GET ADDRESS OF ASCIZ STRING
676      001106      062766      000002      000002      ADD      #2, 2(SP)      ;ADJUST RETURN PC
677      001114      112046      1$:      MOVVB   (RO)+, -(SP)      ;PUSH CHARACTER TO BE TYPED ONTO STACK
678      001116      001003      BNE      2$      ;BR IF IT ISN'T THE TERMINATOR
679      001120      005726      TST      (SP)+      ;IF TERMINATOR POP IT OFF THE STACK
680      001122      012600      MOV      (SP)+, RO      ;RESTORE RO
681      001124      000002      RTI      ;RETURN
682      001126      004737      001160      2$:      JSR      PC, 5$      ;GO TYPE THIS CHARACTER
683      001132      122726      000012      3$:      CMPB    #12, (SP)+      ;CHECK IF THE CHAR. TYPED WAS A LINE FEED
684      001136      001366      BNE      1$      ;GO GET NEXT CHAR. IF NOT LINE FEED
685      001140      013746      001524      MOV      NULL, -(SP)      ;GET # OF FILLER CHARS. NEEDED
686                                     ;AND THE NULL CHAR.
687      001144      105366      000001      4$:      DECB   1(SP)      ;DOES A NULL NEED TO BE TYPED?
688      001150      002770      BLT      3$      ;BR IF NO--GO POP THE NULL OFF OF STACK
689      001152      004737      001160      JSR      PC, 5$      ;GO TYPE A NULL
690      001156      000772      BR       4$      ;LOOP
691      001160      105777      000334      5$:      TSTB   @TPS      ;WAIT UNTIL PRINTER IS READY
692      001164      100375      BPL      5$
693      001166      116677      000002      000326      MOVVB   2(SP), @TPB      ;LOAD CHAR TO BE TYPED INTO DATA REG.
694      001174      000207      RTS      PC
695

```

```

696                                     :*****
697                                     :      DEBUG DUMP ROUTINE
698                                     :*****
699                                     :=1200
700 001200 013746 011042 DMPHLT: MOV     FLAG,-(SP)
701 001204 042737 000070 011042 BIC     #70,FLAG      ;INIT FLAGS
702 001212 104400 001532          TYPE    ,ASTRK        ;TYPE *
703 001216 104402          GETCHR
704 001220 012637 001314          MOV     (SP)+,6$
705 001224 104400 001314          TYPE    6$
706 001230 122737 000127 001542 CMPS    #127,$CHAR    ;W? FOR WORD
707 001236 001004          BNE     1$
708 001240 052737 000010 011042 BIS     #BIT3,FLAG    ;SET FLAG BIT
709 001246 000430          BR      3$
710 001250 122737 000102 001542 1$:  CMPB   #102,$CHAR    ;B? FOR BYTE
711 001256 001004          BNE     2$
712 001260 052737 000020 011042 BIS     #BIT4,FLAG
713 001266 000420          BR      3$
714 001270 122737 000103 001542 2$:  CMPB   #103,$CHAR    ;C? FOR CONTINUE
715 001276 001014          BNE     3$
716 001300 012637 011042          MOV     (SP)+,FLAG
717 001304 052737 000040 011042 BIS     #BITS,FLAG
718 001312 000413          BR      DUMP
719 001314 000000          6$:  000000
720 001316 104400 001526          4$:  TYPE    ,QUES
721 001322 104400 001536          TYPE    ,CRLF
722 001326 000724          BR      DMPHLT
723 001330 005037 001544          3$:  CLR     WORK
724 001334 005726          TST    (SP)+
725 001336 104406 001544          ACCEPT ,WORK
726 001342 012700 001476 DUMP:  MOV    #DMPHLT,RO    ;INIT DUMP LIST
727 001346 062710 000020          ADD    #20,(RO)    ;BUMP ADDRESS
728 001352 032737 000040 011042 BIT     #BITS,FLAG
729 001360 001005          BNE    L1
730 001362 013737 001544 001476 MOV    WORK,DMPHLT
731 001370 001001          BNE    L1
732 001372 022020          CMP    (RO)+,(RO)+ ;SKIP 1ST TWO ENTRIES
733
734 001374 012001          L1:  MOV    (RO)+,R1    ;GET ADDR OF DATA FROM LIST
735 001376 001700          BEQ    DMPHLT      ;BR IF END OF LIST
736 001400 104400 001536          TYPE    ,CRLF
737 001404 010146          MOV    R1,-(SP)    ;PUSH ADDR ON STACK
738 001406 004037 006350          JSR    RO,$B2016  ;PRINT OUT ADDRESS
739
740 001412 032737 000010 011042 BIT     #BIT3,FLAG
741 001420 001014          BNE    L3
742 001422 012702 000020          MOV    #20,R2     ;SET WORD COUNTER = 8
743 001426 005046          L2:  CLR    -(SP)
744 001430 112116          MOVB   (R1)+,(SP)
745 001432 104400 007152          TYPE    ,MSG00
746 001436 004037 006336          JSR    RO,$B200T
747 001442          .BYTE 3
748 001443          .BYTE 1
749 001444 005302          DEC    R2         ;DECREMENT WORD COUNTER
750 001446 001367          BNE    L2        ;BR IF NOT = 0
751 001450 000751          BR     L1        ;GET NEXT ENTRY
    
```



```

752
753 001452 012702 000010      L3:      MOV      #10,R2
754 001456 012146      1$:      MOV      (R1)+,-(SP)
755 001460 104400 007152      TYPE    MSG00
756 001464 004037 006350      JSR     R0,$B2016
757 001470 005302      DEC     R2          ;DECREMENT THE WORD COUNT
758 001472 001371      BNE    1$
759 001474 000737      BR     L1          ;GET NEXT ENTRY
760 001476 000000      DMPST: 0          ;RESERVED FOR SW. REG
761 001500 000000      0          ;END OF TABLE FOR SW. REG
762 001502 000001      .RX:    .BLKW    1
763 001504 000001      .TX:    .BLKW    1
764 001506 000000      0
765 001510 000000      0
766 001512 000000      0
767 001514 177560      TKS:    177560     ;TTY KEYBOARD STATUS REG. ADDRESS
768 001516 177562      TKB:    177562     ;TTY KEYBOARD DATA BUFFER REG. ADDRESS
769 001520 177564      TPS:    177564     ;TTY PRINTER STATUS REG. ADDRESS
770 001522 177566      TPB:    177566     ;TTY PRINTER BUFFER REG. ADDRESS
771 001524 000000      NULL:   .WORD    0 ;CONTAINS NULL CHARACTER FOR FILLS

```

772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807

808
809
810
811
812
813

001526 037440 000040
001532 005015 000052
001536 015
001537 012 000
001542
001544
000000
001544 000000
001546 000000
001550 000000
001552 000000
001554 000000
001556 000000
001560 000000
001562 000000
001564 002604
001566 003004
001570 003076
001572 003126
001574 002604
001576 003004
001600 003076
001602 003126
001604 000400
002604
002604 000100
003004 177 177
003006 040444 052040 042510
003076 177 177
003100 041044 030040 031061
003126 003126
003126 177 177
003130 041444 041440 046517
003230 030040 031061 032063
003254
002604
003004
003076
003126

```
*****  
: CONSTANTS AND WORKING STORAGE  
*****  
QUES: .ASCIZ " ? "  
ASTRK: .ASCIZ <15><12>"*"  
CRLF: .ASCII <15>  
LF: .ASCIZ <12>  
 .EVEN  
$WORK=WORK  
$CHAR: 0  
WORK: 0  
WORK1: 0  
WORK2: 0  
WORK3: 0  
WORK4: 0  
WORK5: 0  
WORK6: 0  
FLAGS: 0  
:DATA PATTERN ADDRESS TABLE AND PATTERNS  
DAT: .WORD VDB ;ADDRESS OF VARIABLE DATA BUFFER  
 .WORD DP1 ;ADDRESS OF MESSAGE 1  
 .WORD DP2 ;ADDRESS OF MESSAGE 2  
 .WORD DP3 ;ADDRESS OF MESSAGE 3  
 .WORD DP4 ;ADDRESS OF MESSAGE 4  
 .WORD DP5 ;ADDRESS OF MESSAGE 5  
 .WORD DP6 ;ADDRESS OF MESSAGE 6  
 .WORD DP7 ;ADDRESS OF MESSAGE 7  
:RECEIVER DATA BUFFER STARTS HERE..  
IBUF: .BLKW 400  
VDB: .BLKW 100 ;VARIABLE DATA BUFFER  
DP1: .BYTE 177,177  
.ASCIZ '$A THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.'<15><12><001><177><177><177><1  
.EVEN  
DP2: .BYTE 177,177  
.ASCIZ '$B 0123456789'<15><12><001><177><177><177><177>  
.EVEN  
DP3: .BYTE 177,177  
.ASCII '$C COM-TEST MAYNARD THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG'  
.ASCIZ ' 0123456789'<15><12><001><177><177><177><177>  
.EVEN  
DP4=VDB ;SPARE  
DP5=DP1 ;SPARE  
DP6=DP2 ;SPARE  
DP7=DP3 ;SPARE
```



```

014 *****
015 START OF PROGRAM
016 *****
017 BEGIN:
018 MOV #STACK,SP ;SETUP THE STACK POINTER
019 STPS,PRTY7
020 SUSWR
021 CLR $PASS ;CLEAR THE PASS COUNT
022 MOV #BEGIN,$LPADR ;INITILIZE THE LOOP ADDRESS FOR SCOPE
023 CLR FLAGS ;RESET FLAGS
024 RESET
025 *****
026 GET PARAMETERS FROM OPERATOR
027 *****
028 CMP #100000,$SWR ;MANUAL INPUT??
029 BNE GETIT ;BR IF NO
030 *****
031 SWITCH REG INPUT ROUTINE
032 *****
033 MANIN: MOV #DN+4,R1 ;PRESET POINTER FOR DN-11
034 CLR RO ;CLEAR DISPLAY
035 HALT ;HALT FOR ISR REQUEST
036 MOV $SWR, FLAGS ;SAVE ISR REQUEST INDICATORS
037 BIT #20000,$SWR ;IS DN11 SETUP REQUESTER?
038 BEQ .+6 ;BR IF NO
039 JSR PC,MANBA ;GO SETUP DN11
040 *****
041 MOV #DMB+4,R1 ;PRESET DMB ISR ADDRESS
042 BIT #40000,FLAGS ;IS DMB SETUP REQUESTED?
043 BEQ .+6 ;BR IF NO
044 JSR PC,MANBA ;GO SETUP DM11-B
045 *****
046 MOV #VISR+4,R1 ;PRESET VARIABLE ISR ADDRESS
047 JSR PC,MANBA ;GO SETUP VARIABLE ISR
048 *****
049 JMP SWRSET ;GO GET OPERATIONAL SWITCHES
050 *****
051 MANBA: MOV (R1), RO ;DISPLAY BUS ADDR
052 HALT ;HALT FOR BUS ADDR
053 MOV $SWR, (R1)+
054 MOV (R1), RO ;DISPLAY VECTOR ADDR
055 HALT ;HALT FOR VECTOR ADDR
056 MOV $SWR, (R1)+
057 MOV (R1), RO ;DISPLAY PRIORITY
058 HALT ;HALT FOR PRIORITY
059 MOV $SWR, (R1)+
060 MOV (R1), RO ;DISPLAY PARAM #1
061 BMI MANINX ;BR IF PARAM NOT REQUIRED
062 HALT ;HALT FOR PARAM #1
063 MOV $SWR, (R1)+
064 MOV (R1), RO ;DISPLAY PARAM #2
065 BMI MANINX ;BR IF PARAM NOT REQUIRED
066 HALT ;HALT FOR PARAM #2
067 MOV $SWR, (R1)+
068 MOV PC ;MANINX: RTS
069
003254 012706 001070
003254 104414 000340
003260 104420
003264 005037 004774
003266 012737 003254 003462
003272 005037 001562
003300 000005
003306 022777 100000 005530
003314 001063
003316 012701 010164
003322 005000
003324 000000
003326 017737 005512 001562
003334 032777 020000 005502
003342 001402
003344 004737 003404
003350 012701 010406
003354 032737 040000 001562
003362 001402
003364 004737 003404
003370 012701 011004
003374 004737 003404
003400 000137 004200
003404 011100
003406 000000
003410 017721 005430
003414 011100
003416 000000
003420 017721 005420
003424 011100
003426 000000
003430 017721 005410
003434 011100
003436 100410
003440 000000
003442 017721 005376
003446 011100
003450 100403
003452 000000
003454 017721 005364
003460 000207

```

```

870 003462 000000 $LPADR: 0
871
872 ;*****
873 ;      ISR PARAM INPUT ROUTINE
874 ;*****
875 003464 012700 011000 GETIT: MOV #VISR, R0 ;PRESET ISR ADDR IN R0
876 003470 104400 007154 TYPE ,MSG01 ;<15><12> INTERFACE TYPE
877 003474 104400 011000 TYPE ,VISR ;PRINTOUT ISR NAME
878 003500 104400 001526 TYPE ,QUES
879
880 003504 104404 001544 GETSTR ,WORK ;READIN OPERATOR'S RESPONSE
881 003510 123727 001544 000015 CMPB WORK, #15 ;IS IT CAR. RET?
882 003516 001431 BEQ GETBA ;BR IF YES
883
884 003520 012700 010160 MOV #DN, R0 ;PRESET DN ISR ADDR
885 003524 023737 010160 001544 CMP DN, WORK ;IS IT DN?
886 003532 001004 BNE DMBTST ;BR IF NO
887 003534 052737 020000 001562 BIS #20000, FLAGS ;SET DN11 FLAG
888 003542 000417 BR GETBA ;GO GET DN11 PARAMS
889 003544
890 003544 012700 010402 DMBTST: MOV #DMB, R0 ;PRESET DM11B ISR
891 003550 023737 010402 001544 CMP DMB, WORK ;IS IT DM?
892 003556 001004 BNE NOISR ;BR IF NO
893 003560 052737 040000 001562 BIS #40000, FLAGS ;SET DM11-B FLAG
894 003566 000405 BR GETBA ;GO GET DM11-B PARAMS
895
896 003570
897 003570 104400 007177 NOISR: TYPE ,MSG02 ;<15><12> ISR NOT LOADED!
898
899 003574 000000 HALT ;HALT
900 003576 000137 003254 JMP BEGIN ;TRY AGAIN
901 003602 010004 GETBA: MOV R0, R4 ;SAVE POINTER
902 003604 022020 CMP (R0)+, (R0)+ ;INCREMENT ISR POINTER
903 003606 104400 007222 TYPE ,MSG03 ;<15><12> BUS ADDRESS=
904 003612 004737 004776 JSR PC,GETANY ;GET THE BUS ADDR
905
906 003616 104400 007241 GETVA: TYPE ,MSG04 ;<15><12> VECTOR ADDRESS=
907 003622 004737 004776 JSR PC,GETANY ;GET THE VECTOR ADDR.
908
909 003626 104400 007263 GETPRI: TYPE ,MSG05 ;<15><12> PRIORITY=
910 003632 004737 004776 JSR PC,GETANY ;GET THE PRIORITY
911
912 003636 005710 GETPRM: TST (R0) ;PARAM #1 REQUIRED?
913 003640 100412 BMI GETP3 ;BR IF NO
914 003642 104400 007277 TYPE ,MSG06 ;<15><12> PARAMS #1=
915 003646 004737 004776 JSR PC,GETANY ;GET PARAM
916
917 003652 005710 TST (R0) ;PARAM #2 REQUIRED?
918 003654 100404 BMI GETP3 ;BR IF NO
919 003656 104400 007314 TYPE ,MSG07 ;<15><12> PARAMS #2=
920 003662 004737 004776 JSR PC,GETANY ;GET PARAM
921
922 003666 016437 000016 003704 GETP3: MOV 16(R4), ARIA ;IS ASCII PARAM REQUIRED
923 003674 100424 BMI GETEX ;BR IF NO
924 003676 104400 007331 TYPE ,MSG08 ;<15><12> ASCII PARAM=
925 003702 104400 TYPE ;PRINTOUT ASCII PARAM

```



```

926 003704 000000          ARIA:  0
927 003706 104400 001526      TYPE      ,QUES
928
929 003712 104404          GETSTR          :GET ASCII INPUT AND
930 003714 001604          IBUF          :PUT IT HERE
931 003716 012702 001604      MOV      #IBUF, R2      :SETUP POINTER
932 003722 122712 000015      CMPB     #15, (R2)     :WAS NEW DATA EN(ERED)?
933 003726 001407          BEQ      GETEX        :BR IF NO
934
935 003730 013703 003704      MOV      ARIA, R3      :SETUP DEST. POINTER
936 003734 112223          MOVB     (R2)+, (R3)+  :MOV INPUT TO DEST.
937 003736 122712 000015      CMPB     #15, (R2)     :LAST DIGIT?
938 003742 001374          BNE     -5            :LOOP IF NO
939 003744 105013          CLRB    (R3)         :INSERT ALL ZERO CHAR
940
941
942 003746 020427 011000      GETEX:  CMP      R4, #VISR :WAS THIS THE VARIABLE ISR
943 003752 001461          BEQ      SWPRNT      :BR IF YES
944 003754 000137 003464          JMP      GETIT       :GET ANOPHER
945
946          ;TTY INTERRUPTS HERE WHEN MODULE IS RUNNING.
947
948 003760 017701 175532      TTYINT: MOV      @TKB,R1   :CLEAR TTY BUFFER
949 003764 042777 000100 175522      BIC      #100,@TKS    :RESET INT. ENABLE
950 003772 042701 177700          BIC      #177700,R1   :STRIP JUNK
951 003776 022701 000007          CMP      #7,R1
952 004002 001013          BNE     2$
953 004004 022737 000176 011044      CMP      #SWREG,SWR
954 004012 001003          BNE     1$
955 004014 052737 000001 011042      BIS      #BIT0,FLAG
956 004022 052777 000100 175464 1$:  BIS      #100,@TKS
957 004030 000002          RTI
958 004032 022701 000077      2$:  CMP      #'?', R1     :IS IT ?
959 004036 001014          BNE     NOQ          :BR IF NO
960 004040 012700 001476          MOV      #DMPNST,R0   :SETUP DUMP LIST
961 004044 012710 001604          MOV      #IBUF, (R0)  :TO PRINTOUT INPUT BUFFER
962 004050 017737 004744 001502      MOV      @IRDA,.RX    :IF SWITCH REG. =0 PRINT RX BUFFER.
963 004056 017737 004740 001504      MOV      @IXDA,.TX    : " " " " PRINT TX BUFFER
964 004064 000137 001374          JMP      LI           :AND GO PRINT IT
965
966 004070 022701 000012      NOQ:  CMP      #12, R1   :IS IT LINE FEED?
967 004074 001002          BNE     RSTART      :BR IF NO
968 004076 000137 003254          JMP      BEGIN       :RESTART
969
970 004102 104400 010155      RSTART: TYPE      ,MFILL
971 004106 000005          RESET
972 004110 105227 000000          INCB     #0          :DELAY HERE FOR AWILE
973 004114 001375          BNE     -4
974 004116 022737 000176 011044      SWPRNT: CMP      #SWREG,SWR
975 004124 001007          BNE     XSWPNT
976 004126 052737 000002 011042      BIS      #BIT1,FLAG
977 004134 104400 007350          TYPE      ,MSG09
978 004140 104422          SETSWI
979 004142 000417          BR      REST
980 004144 104400 007350      XSWPNT: TYPE      ,MSG09 :<15><12> SET SWITCHES
981 004150 105777 175340          TSTB    @TKS        :WAIT FOR TTY INPUT

```

```

982 004154 100375 BPL -4 ;LOOP
983 004156 017702 175334 MOV @TKB,R2 ;RESET DONE FLAG
984 004162 017746 004656 MOV @SWR, -(SP)
985 004166 004037 006350 JSR R0,$B2016 ;PRINTOUT SWITCHES
986 004172 104400 001536 TYPE ,CALF
987 004176 000401 BR .+4 ;SKIP OVER HALT
988
989 *****
990 SET SWITCH OPTIONS
991 *****
992 004200 000000 SWRSET: HALT ;HALT FOR SWITCH SETUP
993 ;SW00=ONE WAY OUT
994 ;SW01=ONE WAY IN
995 ;SW02=EXTERNAL LOOPBACK
996 ;SW03=INTERNAL LOOPBACK
997 ;SW04=LOOP ON DATA
998 ;SW05=MONITOR INPUT
999 ;SW06=MONITOR OUTPUT
1000 ;SW07=NO DATA COMPARE
1001 ;SW08=EXTERNAL DATA
1002 ;SW09=DATA SELECT
1003 ;SW10=DATA SELECT
1004 ;SW11=DATA SELECT
1005 ;SW12=
1006 ;SW13=INHIBIT ERROR TYPEOUTS
1007 ;SW14=LOOP ON TEST
1008 ;SW15=HALT ON ERROR
1009
1010 004202 012737 004210 003462 REST: MOV #RESTR,$LPADR ;SETUP LOOP
1011 004210 017701 004630 RESTR: MOV @SWR,R1
1012 004214 000301 SWAB R1
1013 004216 032777 000017 004620 BIT #17,@SWR ;WAS SOME MODE SELECTED?
1014 004224 001003 BNE .+10 ;BR IF YES
1015 004226 104400 007637 TYPE ,MSG2 ;<15><12>NO MODE SELECTED.
1016 004232 000723 BR RSTART ;GO GET SWITCH REGISTER.
1017 004234 042701 177761 BIC #177761,R1 ;STRIP JUNK
1018 004240 016137 001564 011022 MOV DAT(1),IXDA ;SETUP INIT DATA ADDR FROM TABLE
1019 004246 005701 TST R1 ;VARIABLE DATA SPECIFIED?
1020 004250 001010 BNE SUXCC ;BR IF NO
1021 004252 032777 000400 004564 BIT #400,@SWR ;USE EXTERNAL DATA?
1022 004260 001004 BNE SUXCC ;BR IF YES
1023 *****
1024 GET VARIABLE DATA
1025 *****
1026 004262 104400 007372 TYPE ,MSG10 ;<15><12> ENTER DATA <15><12>
1027 004266 104404 GETSTR ;GET DATA
1028 004270 002604 VDB ;AND PUT IT HERE
1029 004272 012737 001604 011020 SUXCC: MOV #IBUF,IRDA ;SETUP READ BUFFER ADDR
1030
1031 004300 032777 000400 004536 BIT #400,@SWR ;EXTERNAL DATA?
1032 004306 001403 BEQ SWRNXT ;BR IF NO
1033 004310 012737 002604 011022 MOV #VDB,IXDA ;SETUP BUFFER ADDRESS
1034 004316 SWRNXT:
1035 004316 012737 003760 000060 MOV #TTYINT,@#60 ;SETUP TTY VECTOR
1036 004324 012737 000340 000062 MOV #340,@#62
1037 004332 012777 000100 175154 MOV #100,@TKS ;AND ENABLE INTERRUPTS

```


1038	004340	012702	001604
1039	004344	005022	
1040	004346	022702	002004
1041	004352	001374	

CLRIB:	MOV	#IBUF, R2	
	CLR	(R2)+	;CLEAR INPUT BUFFER
	CMP	#IBUF+200,R2	
	BNE	CLRIB	

```

1042 ;*****
1043 ;          SETUP TIMER          *
1044 ;*****
1045
1046 004354 012737 000060 006334 SUTIME: MOV    #60,    MSECS ;PRESET COUNTER
1047 004362 012737 006312 000100      MOV    #TIMER, 100 ;SETUP LINE CLOCK VECTOR
1048 004370 012737 000340 000102      MOV    #340,   102 ;AND PRIORITY
1049 004376 012737 004422 000004      MOV    #NOLC,  4 ;SETUP BUS ERROR VECTOR
1050 004404 012737 000340 000006      MOV    #340,   6 ;SET UP PRIORITY TO 7
1051 004412 052737 000100 177546      BIS    #100,   177546 ;ENABLE LINE CLOCK
1052 004420 000441
1053
1054 ;BUS ERROR RETURNS HERE IF NO LINE CLOCK
1055
1056 004422 012737 006312 000104 NOLC:  MOV    #TIMER, 104 ;SETUP RTC VECTOR
1057 004430 012737 000340 000106      MOV    #340,   106 ;AND PRIORITY
1058 004436 012737 004472 000004      MOV    #1$,    4 ;SETUP BUS ERROR VECTOR
1059 004444 012737 000340 000006      MOV    #340,   6 ;SET PRIORITY
1060 004452 012737 003100 172542      MOV    #1600., 172542 ;SET COUNTER BUFFER.
1061 004460 012737 000111 172540      MOV    #111,   172540 ;ENABLE REAL TIME TIME CLOCK
1062 004466 000240
1063 004470 000415
1064 004472 012737 004516 000100 1$:    MOV    #2$, 100 ;TRY LSI CLOCK
1065 004500 104414 000000      STPS,PRTY0 ;LOWER PSW TO 0
1066 004504 005227 000000      3$:    INC    #0
1067 004510 001375      BNE    3$
1068 004512 104400 007570      TYPE,MSG19 ;NO CLOCK AVAILABLE
1069 004516 012737 006312 000100 2$:    MOV    #TIMER, 100
1070
1071 004524 000137 004530 NORTC: JMP    .+4 ;SPARE JUMP
1072 004530 012737 000006 000004      MOV    #6, 2#4 ;SET TRAP VECTOR
1073 004536 005037 000006      CLR    2#6 ;SET BUS ERROR VECTOR
1074 004542 012706 001070      MOV    #STACK, SP ;SETUP STACK
1075 004546 104414 000000      STPS,PRTY0
1076 004552 012737 006350 011030      MOV    #B2016, B2016 ;SETUP BIN TO OCT ADDR
1077
1078
1079 ;*****
1080 ;          DO TESTING NOW          *
1081 ;*****
1082 004560 032737 020000 001562      BIT    #20000, FLAGS ;WAS A DN11 SETUP
1083 004566 001402      BEQ    DMCHK ;BR IF NO
1084 004570 004737 010200      JSR    PC,DNGO ;GO TO DN11 ISR
1085
1086 004574 032737 040000 001562 DMCHK: BIT    #40000, FLAGS ;WAS A DM11-B SETUP?
1087 004602 001402      BEQ    VIGO ;BR IF NO
1088 004604 004737 010422      JSR    PC,DMGO ;GO TO DM11-B ISR
1089
1090 ;*****
1091 ;          GOTO THE MODULE AND RUN          *
1092 ;*****
1093
1094 004610 004777 004222      VIGO:  JSR    PC,2ISR+36 ;GO TO ISR

```



```

1095 :*****
1096 :      END OF PASS ROUTINE
1097 :*****
1098 004614 005237 004774 EOP:  INC  $PASS      ;INCREMENT PASS COUNTER
1099 004620 005746      TST  -(SP)      ;PUSH DOWN AND PROTECT STACK.
1100 004622 104416      KBDIN
1101 004624 032777 010000 004212 BIT  #SW12,DSWR  ;INHIBIT TYPEOUTS?
1102 004632 001034      BNE  2$        ;BR IF YES
1103 004634 104400 007411      TYPE  MSG11    ;' 5<<12>' END OF PASS
1104 004640 013746 004774      MOV  $PASS, -(SP)
1105 004644 004037 006350      JSR  RD,$B2016 ;PRINTOUT PASS COUNT
1106 004650 104400 001536      TYPE  ,CRLF
1107 004654 032700 000002      BIT  #OWI,MODE ;SKIP TRANSMIT TYPEOUT IF OWI
1108 004660 001012      BNE  3$        ;BR IF YES
1109 004662 104400 010060      TYPE  MSG26    ;TRANSMITTED DATA=
1110 004666 013737 011022 004676 MOV  IXDA,4$   ;SET POINTER TO TXBUF
1111 004674 104400      TYPE          ;TYPE TXBUFFER
1112 004676 000000      4$:  0
1113 004700 032700 000001      BIT  #OWO,MODE ;SKIP RECEIVE TYPEOUT IF OWO
1114 004704 001007      BNE  2$        ;BR IF YES
1115 004706 104400 010106      3$:  TYPE  MSG27    ;RECEIVED DATA=
1116 004712 013737 011020 004722 MOV  IRDA,5$   ;SET POINTER TO RXBUF
1117 004720 104400      TYPE          ;TYPE RXBUFFER
1118 004722 000000      5$:  0
1119 004724 032777 040000 004112 2$:  BIT  #BIT14,DSWR ;LOOP ON TEST?
1120 004732 001005      BNE  1$        ;BR IF NO...
1121 004734 016600 000002      MOV  2(SP),RO ;GET RETURN ADDRESS
1122 004740 104414 000000      STPS,PRTYD
1123 004744 000110      JMP  (RO)      ;GO BACK TO MODULE.
1124 004746 012706 001070      1$:  MOV  #STACK,SP ;RESET THE STACK POINTER.
1125 004752 013700 000042      MOV  D#42, RO ;GET MONITOR ADDRESS
1126 004756 001404      BEQ  $DOAGN    ;BR IF NONE
1127 004760 004710      JSR  PC,(RO)  ;GO TO MONITOR
1128 004762 000240      NOP
1129 004764 000240      NOP          ;SAVE ROOM FOR
1130 004766 000240      NOP          ;ACT-11
1131 004770 000137 000200 $DOAGN: JMP  D#20C   ;RESTART TEST
1132 004774 000000 $PASS:  0
1133
1134
1135 :*****
1136 :      SUBROUTINE TO INPUT OCTAL WORD FROM OPERATOR
1137 :*****
1138 004776 011046 GETANY: MOV  (RO), -(SP) ;PUT WORD ON STACK
1139 005000 004037 006350      JSR  RD,$B2016 ;AND TYPE IT
1140 005004 104400 001526      TYPE  ,QUES
1141 005010 011037 001544      MOV  (RO),WORK ;PRESET FOR DEFAULT (CR)
1142 005014 104406 001544      ANYMOR: ACCEPT WORK ;OCTAL READIN
1143 005020 013710 001544      MOV  WORK,(RO) ;MOVE IT TO ISR
1144 005024 005720      ANYEX: TST  (RO)+ ;BUMP POINTER
1145 005026 000240      NOP
1146 005030 000207      RTS  PC      ;SUB/ROUTINE EXIT
1147
1148 :*****
1149 :      ERROR HLT HANDLER
1150 :*****

```


1151	005032			\$HLT:		
1152	005032	104414	000140		STPS,PRTY3	;LOWER PSW PRIOTITY TO 3
1153	005036	005237	005730	\$HLOT:	INC \$ERTTL	;INCREMENT ERROR COUNTER
1154	005042	001775			BEQ \$HLOT	;MAKE SURE ITS NOT ZERO
1155	005044	011637	005726		MOV (SP) \$HLTAD	;SAVE ADDRESS OF HLT
1156	005050	162737	000002	005726	SUB #2, \$HLTAD	;AND BACK IT UP
1157	005056	010146			MOV R1, -(SP)	;SAVE R1
1158						
1159	005060	032777	020000	003756	BIT #BIT13, @SWR	;INHIBIT ERR TYPEOUTS?
1160	005066	001070			BNE TRX	;BR IF YES
1161						
1162	005070	104400	001536		TYPE ,CRLF	
1163	005074	117701	000626		MOVB \$HLTAD,R1	;EXTRACT HLT CODE
1164	005100	006301			ASL R1	;AND ALIGN IT
1165	005102	016137	005270	005112	MOV EMTAB(R1),.+10	;GET HEADER ADDRESS
1166	005110	104400	005320		TYPE ,EMD	;AND PRINT HEADER
1167	005114	104400	007430		TYPE ,MSG12	; < AT LOC >
1168	005120	013746	005726		MOV \$HLTAD,-(SP)	;GET HLT ADDRESS
1169	005124	004037	006350		JSR RO,\$B2016	;AND PRINT IT
1170	005130	005701			TST R1	;HLT CODE = 0?
1171	005132	001446			BEQ TRX	;BR IF YES
1172						
1173						
1174	005134	022701	000016		CMP #16,R1	;IS IT HLT+7?
1175	005140	001023			BNE 1\$;BR IF NO
1176	005142	005702			TST R2	;PRINTOUT BAD DATA?
1177	005144	001406			BEQ 2\$;BR IF NO
1178	005146	104400	007767		TYPE ,MSG23	; < BAD DATA= >
1179	005152	110246			MOVB R2,-(SP)	;GET DATA
1180	005154	004037	006336		JSR RO,\$B20CT	;AND PRINT IT
1181	005160	003			.BYTE 3	
1182	005161	001			.BYTE 1	
1183	005162	005703		2\$:	TST R3	;PRINT OUT GOOD DATA?
1184	005164	001410			BEQ 3\$;BR IF NO
1185	005166	104400	010003		TYPE ,MSG24	; < GOOD DATA= >
1186	005172	110346			MOVB R3,-(SP)	;GET DATA
1187	005174	004037	006336		JSR RO,\$B20CT	;AND PRINT IT
1188	005200	003			.BYTE 3	
1189	005201	001			.BYTE 1	
1190	005202	104400	001536		TYPE ,CRLF	
1191	005206	000420		3\$:	BR †TRX	
1192						
1193						
1194	005210	005702		1\$:	TST R2	;PRINTOUT RCV CSR?
1195	005212	001405			BEQ TR3	;BR IF NO
1196	005214	104400	007441		TYPE ,MSG13	; < RCV CSR=>
1197	005220	010246			MOV R2,-(SP)	;GET DATA
1198	005222	004037	006350		JSR RO,\$B2016	;AND PRINT IT
1199						
1200	005226	005703		TR3:	TST R3	;PRINTOUT XMIT CSR?
1201	005230	001407			BEQ TRX	;BR IF NO
1202	005232	104400	007453		TYPE ,MSG14	; < XMIT CSR=>
1203	005236	010346			MOV R3,-(SP)	;GET DATA
1204	005240	004037	006350		JSR RO,\$B2016	;AND PRINT IT
1205	005244	104400	001536		TYPE ,CRLF	
1206						


```

1207 005250 032777 100000 003566 TRX: BIT #BIT15, @SWR ;HALT ON ERROR?
1208 005256 001401 BEQ HLTX ;BR IF NO
1209 005260 000000 HALT
1210
1211 005262 104416 HLTX: KBDIN
1212 005264 012601 MOV (SP)+, R1 ;RESTORE R1
1213 005266 000002 RTI ;AND RETURN TO PROGRAM
1214
1215 005270 005320 EMTAB: EMO
1216 005272 005333 EM1
1217 005274 005343 EM2
1218 005276 005363 EM3
1219 005300 005407 EM4
1220 005302 005422 EM5
1221 005304 005446 EM6
1222 005306 005542 EM7
1223 005310 005565 EM10
1224 005312 005642 EM11
1225 005314 005667 EM12
1226 005316 005711 EM13
1227

```

```

005320 051105 047522 020122 EMO: .ASCIZ "ERROR HALT"
005333 127 044501 044524 EM1: .ASCIZ "WAITING"
005343 127 044501 044524 EM2: .ASCIZ "WAITING TO XMIT"
005363 104 026516 030461 EM3: .ASCIZ "DN-11 NOT AVAILABLE"
005407 104 030516 026461 EM4: .ASCIZ "DN11-ERROR"
005422 047104 030461 041440 EM5: .ASCIZ "DN11 CALL ABANDONED"
005446 041522 020126 052502 EM6: .ASCIZ "RCV BUFFER FULL,END OF MESSAGE CHARACTER(001) WAS NOT FOUND"
005542 040504 040524 041440 EM7: .ASCIZ "DATA COMPARE ERROR"
005565 105 051122 051117 EM10: .ASCIZ "ERROR RCV CSR=CONTENTS OF SELECT 0 REGISTER"
005642 047125 054105 042520 EM11: .ASCIZ "UNEXPECTED INTERRUPT"
005667 116 046530 050040 EM12: .ASCIZ "NXM PRINCIPAL CAR"
005711 116 046530 040440 EM13: .ASCIZ "NXM ALT CAR"
005726 005726 .EVEN
1228 005726 000000 $HLTAD: 0
005730 000000 $ERTTL: 0

```

```

1229 :*****
1230 : READ A CHAR. ROUTINE
1231 :*****
1232 :
1233 : CALL= GETCHR ;INPUT A CHAR FROM TTY
1234 : RETURNS HERE WITH CHAR ON STACK
1235 005732 011646 $READC: MOV (SP),-(SP) ;PUSH THE PC
1236 005734 016666 MOV 4(SP),2(SP) ;SAVE THE PS
1237 005742 105777 000004 000002 TSTB @TKS ;IS RECEIVE DONE
1238 005746 100375 BPL -4 ;LOOP IF NO
1239 005750 017737 173542 001542 MOV @TKB,$CHAR ;SAVE THE CHAR.
1240 005756 042737 177600 001542 BIC #177600,$CHAR ;STRIP JUNK
1241 005764 013766 001542 000004 MOV $CHAR,4(SP) ;PUT CHAR ON STACK
1242 005772 000002 RTI ;EXIT
1243 :*****
1244 : READ A STRING ROUTINE
1245 :*****
1246 :
1247 : CALL= GETSTR ;INPUT A STRING OF CHARS FROM TTY
1248 : ADDR ;TO THIS ADDRESS
1249 : TERMINATE INPUT WITH LINE FEED
1250 005774 011602 $READS: MOV (SP), R2 ;SETUP ADDRESS OF INPUT BUFFER
1251 005776 012201 MOV (R2)+, R1 ;INCREMENT RETURN ADDRESS
1252 006000 010216 MOV R2,(SP) ;AND PUT BACK ON STACK
1253 :
1254 006002 104402 GETIC: GETCHR ;GET A CHAR
1255 006004 104400 001542 TYPE $CHAR
1256 006010 122726 000136 CMPB #136,(SP)+ ;IS IT BINARY DELIMITER
1257 006014 001011 BNE GOTIC ;BR IF NO
1258 :
1259 006016 104406 001544 OCT: ACCEPT $WORK ;GET OCT. CHAR
1260 006022 113721 001544 MOVB $WORK,(R1)+ ;STORE OCT. CHAR.
1261 006026 122737 000136 001542 CMPB #136,$CHAR ;TERMINATOR=BIN. DELIMITER?
1262 006034 001370 BNE OCT ;BR IF NO
1263 006036 000761 BR GETIC
1264 :
1265 006040 113721 001542 GOTIC: MOVB $CHAR,(R1)+ ;STORE CHAR. IN BUFFER
1266 006044 022737 000015 001542 CMP #15,$CHAR ;IS IT END OF INPUT (CAR. RETURN)
1267 006052 001353 BNE GETIC ;BR IF NO
1268 006054 112721 000012 MOVB #12,(R1)+
1269 006060 104400 001537 TYPE LF ;TYPE A LINE FEED
1270 006064 112721 000001 MOVB #001,(R1)+ ;INSERT RX TERM.
1271 006070 112721 000177 MOVB #177,(R1)+ ;AND A FILL
1272 006074 112721 000177 MOVB #177,(R1)+ ;INSERT ANOTHER FILL
1273 006100 112721 000177 MOVB #177,(R1)+ ;INSERT 3RD FILL
1274 006104 112721 000177 MOVB #177,(R1)+ ;INSERT 4TH FILL
1275 006110 105011 CLAB (R1) ;PUT ZEROS AT END
1276 006112 000002 RTI
1277

```



```

1278
1279
1280
1281
1282
1283 006114
1284 006114 010046
1285 006116 010146
1286 006120 010246
1287 006122 010346
1288 006124 016600 000010
1289 006130 005001 1$: CLR R1 ;GET ADDRESS OF WHERE TO PUT NUMBER
1290 006132 012702 000006 1$: MOV #6,R2 ;CLEAR PARTIAL NUMBER
1291 006136 104402 2$: GETCHR ;MAX. # OF DIGITS ALLOWED
1292 006140 112603 ;GET ONE CHARACTER
1293 006142 110337 006310 ;AND PUT IT IN R3
1294 006146 104400 006310 MOV R3,6$
1295 006152 022703 000025 TYPE .6$ ;ECHO THE CHARACTER
1296 006156 001451 CMP #25,R3
1297 006160 022703 000015 BEQ 8$
1298 006164 001427 CMP #15,R3 ;WAS THIS CHARACTER A "CR"?
1299 006166 022703 000040 BEQ 5$ ;BR IF YES
1300 006172 001433 CMP #40,R3 ;WAS "SPACE" HIT?
1301 006174 022703 000136 BEQ 7$ ;BR IF YES
1302 006200 001430 CMP #136,R3 ;WAS "↑" HIT?
1303 006202 032703 000110 BEQ 7$ ;BR IF YES
1304 006206 001011 BIT #110,R3 ;INSURE THE CHARACTER IS
1305 ;A DIGIT BETWEEN 0 AND 7.
1306 006210 005302 DEC R2 ;CHECK NUMBER OF CHARACTERS
1307 006212 002407 BLT 4$ ;BR IF TOO MANY
1308 006214 006301 ASL R1 ;POSITION PARTIAL NUMBER
1309 006216 006301 ASL R1 ;FOR THIS DIGIT
1310 006220 006301 ASL R1
1311 006222 042703 177770 BIC #1C<7>,R3 ;GET RID OF THE ASCII JUNK
1312 006226 050301 BIS R3,R1 ;COMBINE THIS DIGIT WITH PARTIAL
1313 006230 000742 BR 2$ ;GO GET ANOTHER DIGIT
1314 006232 104400 001526 4$: TYPE .QUES ;TYPE "?"
1315 006236 104400 001536 TYPE .CRLF ;TYPE CARRAGE RETURN AND LINE FEED.
1316 006242 000732 BR 1$ ;GO START OVER
1317 006244 104400 001537 5$: TYPE .LF ;FOLLOW "CR" WITH A "LF"
1318 006250 022702 000006 CMP #6,R2 ;WERE ANY DIGITS INPUT
1319 006254 001002 BNE .+6 ;BR IF YES
1320 006256 013001 MOV @ (R0)+,R1 ;USE OLD DATA
1321 006260 005740 TST -(R0) ;BACKUP R0--
1322 006262 010130 7$: MOV R1,@ (R0)+ ;PASS THE NUMBER TO THE USER
1323 006264 010066 000010 MOV R0,10(SP) ;SET FOR RETURN
1324 006270 012603 MOV (SP)+,R3 ;RESTORE R3
1325 006272 012602 MOV (SP)+,R2 ;RESTORE R2
1326 006274 012601 MOV (SP)+,R1 ;RESTORE R1
1327 006276 012600 MOV (SP)+,R0 ;RESTORE R0
1328 006300 000002 RTI
1329 006302 104400 010150 8$: TYPE .CTLU
1330 006306 000710 BR 1$
1331 006310 000 000 6$: .BYTE 0,0 ;STORAGE FOR ASCII CHAR. AND TERMINATOR

```

```

1233
1234
1235 006312 005337 006334
1236 006316 001005
1237 006320 012737 000060 006334
1238 006326 005237 011032
1239 006332
1240 006332 000002
1241 006334 000000
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253 006336 112037 006543
1254 006342 112037 006541
1255 006346 000406
1256 006350 112737 000001 006541
1257 006356 112737 000006 006543
1258 006364 112737 000005 006540
1259 006372 010346
1270 006374 010446
1271 006376 010546
1272 006400 113704 006543
1273 006404 005404
1274 006406 062704 000006
1275 006412 110437 006542
1276 006416 113704 006541
1277 006422 016605 000010
1278 006426 005003
1279 006430 006105
1280 006432 000404
1281 006434 006105
1282 006436 006105
1283 006440 006105
1284 006442 010503
1285 006444 006103
1286 006446 105337 006542
1287 006452 100016

```

```

*****
:      CLOCK INTERRUPT ROUTINE
*****
TIMER:  DEC      MSECS      :COUNT 60 CYCLES
        BNE      TIMEX      :BR IF NOT 60
        MOV      #60, MSECS  :RESTORE COUNT
        INC      TIME        :INCREMENT SECONDS
TIMEX:
MSECS:  RTI          :RETURN FROM INTERRUPT
        0
*****
:      BINARY TO OCTAL (ASCII) AND TYPE
:      $B20CT---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
:      CALL:
:      MOV      NUM, -(SP)      :NUMBER TO BE TYPED
:      JSR      RD, $B20CT     :CALL FOR TYPEOUT
:      .BYTE   N                :N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
:      .BYTE   M                :M=1 OR 0
:                                  :1=TYPE LEADING ZEROS
:                                  :0=SUPPRESS LEADING ZEROS
:
:      $B201---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST $B20CT OR $B2016
:      CALL:
:      MOV      NUM, -(SP)
:      JSR      RD, $B201
:
:      $B2016---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
:      CALL:
:      MOV      NUM, -(SP)
:      JSR      RD, $B2016
:
$B20CT:  MOV      (RD)+, $OMODE+1 :PICKUP THE NUMBER OF DIGITS TO TYPE
        MOV      (RD)+, $OFILL    :GET THE ZERO FILL SWITCH
        BR      $B201
$B2016:  MOV      #1, $OFILL      :SET THE ZERO FILL SWITCH
        MOV      #6, $OMODE+1    :SET FOR SIX(6) DIGITS
$B201:   MOV      #5, $CNT        :SET THE ITERATION COUNT
        MOV      R3, -(SP)       :SAVE R3
        MOV      R4, -(SP)       :SAVE R4
        MOV      R5, -(SP)       :SAVE R5
        MOV      $OMODE+1, R4    :GET THE NUMBER OF DIGITS TO TYPE
        NEG      R4
        ADD      #6, R4          :SUBTRACT IT FOR MAX. ALLOWED
        MOV      R4, $OMODE      :SAVE IT FOR USE
        MOV      $OFILL, R4      :GET THE ZERO FILL SWITCH
        MOV      10(SP), R5     :PICKUP THE INPUT NUMBER
        CLR      R3             :CLEAR THE OUTPUT WORD
1$:      ROL      R5             :ROTATE MSB INTO "C"
        BR      2$
2$:      ROL      R5             :GO DO MSB
        ROL      R5             :FORM THIS DIGIT
        ROL      R5
        ROL      R5
        MOV      R5, R3
3$:      ROL      R5             :GET LSB OF THIS DIGIT
        DECB    $OMODE         :TYPE THIS DIGIT?
        BPL     7$            :BR IF NO

```



```

1388 006454 042703 177770      BIC      #177770,R3      :GET RID OF JUNK
1389 006460 001002      BNE      4$             :TEST FOR 0
1390 006462 005704      TST      R4             :SUPPRESS THIS 0?
1391 006464 001403      BEQ      5$             :BR IF YES
1392 006466 005204      4$: INC      R4             :DON'T SUPPRESS ANYMORE 0'S
1393 006470 052703 000060      BIS      #'0,R3        :MAKE THIS DIGIT ASCII
1394 006474 052703 000040      5$: BIS      #' ,R3        :MAKE ASCII IF NOT ALREADY
1395 006500 110337 006536      MOVB     R3,8$          :SAVE FOR TYPING
1396 006504 104400 006536      TYPE     8$            :GO TYPE THIS DIGIT
1397 006510 105337 006540      7$: DECB    $OCNT        :COUNT BY 1
1398 006514 003347      BGT      2$            :BR IF MORE TO DO
1399 006516 002402      BLT      6$            :BR IF DONE
1400 006520 005204      INC      R4             :INSURE LAST DIGIT ISN'T A BLANK
1401 006522 000744      BR       2$            :GO DO THE LAST DIGIT
1402 006524 012605      6$: MOV      (SP)+,R5     :RESTORE R5
1403 006526 012604      MOV      (SP)+,R4     :RESTORE R4
1404 006530 012603      MOV      (SP)+,R3     :RESTORE R3
1405 006532 012616      MOV      (SP)+,(SP)   :SET THE STACK FOR RETURNING
1406 006534 000200      RTS      R0            :RETURN
1407 006536      .BYTE   0              :STORAGE FOR ASCII DIGIT
1408 006537      .BYTE   0              :TERMINATOR FOR TYPE ROUTINE
1409 006540      .BYTE   0              :OCTAL DIGIT COUNTER
1410 006541      .BYTE   0              :ZERO FILL SWITCH
1411 006542 000000      $OMODE: 0              :NUMBER OF DIGITS TO TYPE

```

```

1412 006544 010046      :TRAP HANDLER
1413 006546 016600 000002      $TRAP: MOV      R0,-(SP)   :SAVE R0
1414 006552 005740      MOV      2(SP),R0      :GET TRAP ADDRESS
1415 006554 111000      TST      -(R0)         :BACKUP BY 2
1416 006556 016000 006564      MOVB     (R0),R0       :GET RIGHT BYTE OF TRAP
1417 006562 000200      MOV      $TRPAD(R0),R0 :INDEX TO TABLE
1418      RTS      R0           :GO TO ROUTINE

```

:TRAP TABLE
:ROUTINE
:-----

```

1419 006564 001100      $TRPAD: $TYPE
1420      104400      TYPE=TRAP+0
1421 006566 005732      $READC  GETCHR=TRAP+2
1422      104402
1423 006570 005774      $READS  GETSTR=TRAP+4
1424      104404
1425 006572 006114      $ACCEPT ACCEPT=TRAP+6
1426      104406
1427 006574 006610      $RWAIT  RWAIT=TRAP+10
1428      104410
1429 006576 006620      $XWAIT  XWAIT=TRAP+12
1430      104412
1431 006600      .STPS   STPS=TRAP+14
1432      104414
1433 006602      .KBDIN  KBDIN=TRAP+16
1434      104416
1435 006604      .SUSWR

```

```

1444 006606 104420          .SETSWI      SUSWR=TRAP+20
1445          006750          SETSWI=TRAP+22
1446          104422
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500

```

```

*****
SPECIAL PRINTOUT ROUTINES
*****

$RWAIT: TYPE      ,MSG15      ;<15><12> WAITING AT LOC <SP>
          JMP      WAITPO
$XWAIT: TYPE      ,MSG16      ;<15><12> WAITING FOR CLEAR TO SEND AT LOC <SP><SP>
WAITPO: MOV      (SP), -(SP)   ;SETUP ADDRESS OF CALL FOR PRINTOUT
          JSR      RO,$B2016   ;PRINTOUT ADDRESS
          TYPE      ,MSG25      ; <DMBB LINE STATUS REG= >
          MOV      4(SP), -(SP) ;MOV CSR TO BOTTOM OF STACK
          MOV      4(SP), 6(SP) ;MOVE PSW UP A WORD
          MOV      2(SP), 4(SP) ;MOVE PC UP A WORD
          MOV      (SP)+, (SP)  ;MOVE CSR UP A WORD
          JSR      RO,$B2016   ;GO PRINT CSR
          TYPE      ,MSG18      ;<.><15><12>
          ;GIVE IT LINE FEED
          RTI                  ;AND RETURN TO CALLER

*****
POWER DOWN ROUTINE.
SINCE INIT IS ISSUED IN A PWR DN/UP SEQUENCE.
PROGRAM MUST BE RESTARTED AGAIN.
*****
$PWRDN: MOV      #$PWRUP,3#24
          HALT
          BR      .-2

*****
POWER UP ROUTINE.
MESSAGE "POWER HAS FAILED..." WILL BE PRINTED OUT.
PROGRAM WILL BE RESTARTED.
*****
$PWRUP: MOV      #$PWRDN,3#24
          MOV      #STACK,SP
          TYPE      ,MSG20      ;<15><12> POWER FAILED..
          STPS,PRTYO
          JMP      3#200
.KBDIN: BIT      #BIT0,FLAG    ;TEST IG FLAG
          BEQ      OUT          ;NO EXIT
          BIC      #100,3TKS    ;CLEAR TTY IE

```



```

1500 006750 104400 010131      .SETSWI:TYPE      SWEQ      ;TYPE SWR=
1501 006754 017746 002064      MOV        @SWR,-(SP) ;SET UP OCTAL TYPEOUT
1502 006760 004037 006350      JSR        RD,$B2016 ;DO IT
1503 006764 104400 010141      TYPE      NEQ      ;TYPE NEW=
1504 006770 017737 002050 001544      MOV        @SWR,WORK ;SET UP FOR CR DEFAULT
1505 006776 104406 001544      ACCEPT    ,WORK    ;GET VALUE
1506 007002 013777 001544 002034      MOV        WORK,@SWR ;REPLACE IT
1507 007010 032737 000002 011042      BIT        #BIT1,FLAG ;SEE HOW WE GOT HERE
1508 007016 001005      BNE       1$      ;WRONG WAY?
1509 007020 005777 172472      TST       @TKB    ;CLEAR BUFFER
1510 007024 052777 000100 172462      BIS        #100,@TKS ;RESET TTY IE
1511 007032 042737 000003 011042 1$:      BIC        #BIT0+BIT1,FLAG ;CLEAR FLAG BITS
1512 007040 000002      OUT:      RTI      ;EXIT
1513
1514 007042 042766 000340 000002 .STPS:    BIC        #PTY7,2(SP) ;CLEAR OUT PRIORITY BITS
1515 007050 057666 000000 000002      BIS        @SP,2(SP) ;SET NEW PRIORITY
1516 007056 062716 000002      ADD       #2,(SP)  ;SETUP EXIT
1517 007062 000002      RTI      ;EXIT
1518
1519 007064 013746 000006      .SUSWR: MOV        6,-(SP)  ;SAVE 6 ON STACK
1520 007070 013746 000004      MOV        4,-(SP)  ;SAVE 4 ON STACK
1521 007074 012737 007122 000004      MOV        #1$,4    ;SETUP TIMEOUT
1522 007102 012737 000340 000006      MOV        #340,6   ;SET PRIORITY
1523 007110 022777 177777 001726      CMP        #-1,@SWR ;TEST FOR 177570
1524 007116 001402      BEQ       2$      ;NOT ALL 1'S
1525 007120 000407      BR        3$      ;IT'S THERE - EXIT
1526 007122 022626 1$:      CMP        (SP)+,(SP)+ ;ADJUST STACK AFTER TRAP
1527 007124 012737 000176 011044 2$:      MOV        #SWREG,SWR ;REPLACE HARDWARE REGISTERS
1528 007132 012737 000174 011046      MOV        #DISPREG,DISPLAY ;WITH SOFTWARE REGISTERS
1529 007140 012637 000004 3$:      MOV        (SP)+,4   ;RESTORE 4
1530 007144 012637 000006      MOV        (SP)+,6   ;RESTORE 6
1531 007150 000002      RTI      ;EXIT
    
```



```

(1)
1541
1542
1543 010160 047104 000040
1544 010164 175200
1545 010166 000350
1546 010170 000200
1547 010172 177777
1548 010174 177777
1549 010176 010356
1550
1551
1552
1553 100000
1554 040000
1555 010000
1556 000040
1557 000020
1558 000002
1559 000001

```

```

:*****
: DN-11 INTERFACE SERVICE PARAMS
:*****
DN: .ASCIZ "DN "
DNBA: 175200 ;BUS ADDRESS
DNIV: 350 ;INTERRUPT VECTOR
DNPRI: 200 ;PRIORITY
DNPAR1: 177777 ;NOT USED
DNPAR2: 177777 ;NOT USED
DNPAR3: DIALNO ;ADDRESS OF DIAL #

```

```

PWI=100000 ;POWER INDICATOR
ACR=40000 ;ABANDON CALL AND RETRY
DLO=10000 ;DATA LINE OCCUPIED
DSS=40 ;DATA SET STATUS
PND=20 ;PRESENT NEXT DIGIT
DP=2 ;DIGIT PRESENT
CRQ=1 ;CALL REQUEST

```

```

1561 ;*****
1562 ; START OF DN-11 CODE
1563 ;*****
1564 010200 013704 010164 DNGO: MOV DNBA, R4 ;SETUP BUS ADDR
1565 010204 005014 CLR (R4) ;RESET DN-11
1566 010206 005037 011032 CLR TIME ;RESET TIMER
1567 010212 032737 000002 011032 BIT #2, TIME ;AND WAIT 2 SECS
1568 010220 001774 BEQ -.6
1569 010222 012703 010356 MOV #DIALNO,R3 ;SETUP DIAL # ADDRESS
1570 010226 012714 000001 MOV #CRQ, @CSR ;SET CALL REQUEST
1571 010232 032714 100000 BIT #PWI, @CSR ;IS DN AVAILABLE
1572 010236 001425 BEQ DNL1X ;BR IF YES
1573
1574 010240 011402 MOV @CSR, R2 ;SETUP CSR FOR PRINTOUT
1575 010242 005003 CLR R3
1576 010244 104003 HLT+3 ;PRINTOUT "DN NOT AVAILABLE"
1577 010246 000137 010200 JMP DNGO ;RESTART
1578
1579 010252 032714 000200 DNL1: BIT #DONE, @CSR ;IS DONE FLAG SET?
1580 010256 001775 BEQ DNL1 ;WAIT IF NO
1581 010260 042714 000200 BIC #DONE, @CSR ;RESET DONE
1582 010264 032714 140000 BIT #ACR+PWI, @CSR ;ANY ERRORS?
1583 010270 001003 BNE DNL1E ;BR IF YES
1584
1585
1586 010272 032714 000020 BIT #PND, @CSR ;IS PRESENT NEXT DIGIT SET
1587 010276 001005 BNE DNL1X ;BR IF YES
1588
1589 010300 011402 DNL1E: MOV @CSR, R2 ;SETUP CSR FOR PRINTOUT
1590 010302 005003 CLR R3
1591 010304 104004 HLT+4 ;PRINTOUT "DN ERROR"
1592 010306 000137 010200 JMP DNGO ;RESTART
1593
1594 010312 112364 DNL1X: MOVB (R3)+, 1(R4) ;LOAD NEXT DIGIT
1595 010316 052714 000002 BIS #DP, @CSR ;SET DIGIT PRESENT
1596 010322 105713 TSTB (R3) ;WAS THAT LAST CHAR?
1597 010324 001352 BNE DNL1 ;BR IF NO
1598
1599 010326 032714 000200 DNL2: BIT #DONE, @CSR ;WAIT FOR DONE FLAG
1600 010332 001775 BEQ DNL2
1601
1602 010334 032714 040000 BIT #ACR, @CSR ;WAS CALL ABANDONED?
1603 010340 001405 BEQ DNEX ;BR IF NO
1604
1605 010342 011402 MOV @CSR, R2 ;SETUP CSR FOR PRINTOUT
1606 010344 005003 CLR R3
1607 010346 104005 HLT+5 ;PRINTOUT "CALL ABANDONED"
1608 010350 000137 010200 JMP DNGO ;RESTART
1609
1610 010354 000207 DNEX: RTS PC ;RETURN TO ITEP
1611
1612 010356 034470 032467 031460 DIALNO: .ASCIZ "8975030" ;NUMBER TO DIAL
1613 010364 000060
1614 010366 000000 000000 000000 0,0,0,0,0,0
1615 010374 000000 000000 000000

```


1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637

; DM11-B INTERFACE SERVICE PARAMS

010402 046504 000102
010406 170500
010410 000310
010412 000200
010414 000000
010416 177777
010420 177777

DMB: .ASCIZ 'DMB' :ISR NAME
MBA: 170500 :BUS ADDRESS
IV: 310 :VECTOR ADDRESS
PRIO: 200 :PRIORITY
PARA1: 0 :PARAM #1
PARA2: 177777 :PARAM #2
PARA3: 177777 :PARAM #3

```

1638
1639
1640
1641 010422 000240
1642 010424 013704 010406
1643 010430 005037 011032
1644
1645
1646 010434 052714 006000
1647 010440 032714 000020
1648 010444 001375
1649 010446 013714 010414
1650 010452 052764 000003 000002
1651 010460 032737 020000 001562
1652 010466 001002
1653 010470 104400 007663
1654
1655 010474 032777 000005 000342 1$:
1656 010502 001444
1657
1658
1659 010504 052764 000004 000002 STARTX:
1660
1661 010512 032764 000040 000002 CTSW:
1662 010520 001016
1663 010522 023727 011032 000036
1664 010530 103770
1665 010532 016446 000002
1666 010536 032777 010000 000300
1667 010544 001001
1668 010546 104412
1669 010550 005037 011032 1$:
1670 010554 000756
1671
1672 010556 CTSOK:
1673
1674 010556 012737 177777 011024 REX:
1675 010564 005037 010616
1676 010570 012737 000014 010620
1677 010576 062737 000001 010616
1678 010604 001374
1679 010606 005337 010620
1680 010612 001371
1681 010614 000207 REX1:
1682
1683 010616 000000 TEMP1:
1684 010620 000000 TEMP2:
1685 000001
1686 000002
1687 000004
1688 000004
1689 000010
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000

```

```

:*****
: DM-11B INTERFACE SERVICE ROUTINE
:*****
DMBGO: NOP
MOV MBA, R4 ;SETUP BUS ADDR INDEX
CLR TIME ;RESET TIMER

BIS #CS+CM, @CSR ;CLEAR DM-11 B
BIT #BUSY, @CSR ;WAIT TIL FREE
BNE -4
MOV PARA1, @CSR ;SELECT LINE #
BIS #DTR+LE, 2(R4) ;SET DATA TERM RDY & LINE ENABLE
BIT #20000, FLAGS ;HAS DM11 MADE CONNECTION YET?
BNE 1$ ;BR IF YES
TYPE ,MSG22 ;TYPE "MAKE CONNECTION"

BIT #OWO+XLB, @SWR ;IS MODE = OWO OR XLB
BEQ REX1 ;BR IF NO

STARTX: BIS #RQTS, 2(R4) ;SET REQUEST TO SEND

CTSW: BIT #CTS, LSTAT(R4) ;IS CLEAR TO SEND SET?
BNE CTSOK ;BR IF YES
CMP TIME, #36 ;30 SECS ELAPSED?
BLO CTSW ;BR IF NO
MOV LSTAT(R4), -(SP) ;TYPE CONTENTS OF RCSR
BIT #SW12, @SWR ;INHIBIT TYPEOUTS?
BNE 1$ ;BR IF YES
XWAIT ;PRINTOUT 'WAITING FOR CTS'
CLR TIME ;RESET TIMER
BR CTSW ;WAIT SOME MORE

CTSW:

REX: MOV #-1, SETTLE ;SET UP DELAY FLAG
CLR TEMP1
MOV #14, TEMP2
ADD #1, TEMP1
BNE -6
DEC TEMP2
BNE -14
REX1: RTS PC ;RETURN TO CONTROL PROGRAM

TEMP1: 0
TEMP2: 0
OWO=1
OWI=2
TLB=4
XLB=4
ILB=10

RCSR EQUATES
.EQUIV R4, RCSR
.EQUIV R4, CSR

```



```

1694
1695      100000      RI=100000      ;RING INDICATOR
1696      000100      CF=100      ;CARRIER FLAG
1697      000040      CTS=40      ;CLEAR TO SEND
1698      010000      SRD=10000    ;SEC. RECEIVE DATA
1699      004000      CS=4000     ;CLEAR SCAN
1700      002000      CM=2000     ;CLEAR MUX
1701      001000      MM=1000     ;MAINT MODE
1702      000400      STEP=400    ;STEP
1703      000200      DONE=200    ;DONE
1704      000100      IE=100     ;INTERRUPT ENABLE
1705      000040      SE=40      ;SCAN ENABLE
1706      000020      BUSY=20     ;BUST
1707      000017      LINE=17     ;LINE NUMBER
1708
1709      ;LINE STATUS REGGRSTER EQUATES
1710      000002      LSTAT=2
1711
1712      000004      RQTS=4      ;REQUEST TO SEND
1713      000002      DTR=2      ;DATA TERMINAL READY
1714      000001      LE=1       ;LINE ENABLE
1715

```

1716 011000
 1717
 1718
 1719
 1720
 1721
 1722
 1723
 1724
 1725 011000
 1726 011000 000002
 1727 011004 000001
 1728 011006 000001
 1729 011010 000001
 1730 011012 000001
 1731 011014 000001
 1732 011016 000001
 1733
 1734 011020 000001
 1735 011022 000001
 1736 011024 000001
 1737 011026 000001
 1738 011030 000001
 1739 011032 000001
 1740 011034 000001
 1741 011036 000001
 1742 011040
 1743 011040 000001
 1744 011041
 1745 011041 000001
 1746
 1747 011042 000001
 1748 011044 177570
 1749 011046 177570
 1750 011050 000001
 1751 011052 000001
 1752 011054 000001
 1753 011056 000001
 1754 011060 000001
 1755 011062 000001
 1756 011064 000001
 1757 011066 000001
 1758 011070 000001
 1759 011072 000001
 1760 011074 000001
 1761
 1762 011076 000001
 1763 011100 000001
 1764 011102 000001
 1765 000001

```

.=11000
;*****
; THE INTERFACE SERVICE ROUTINE IS LOADED HERE
;*****
; THE FOLLOWING 18 WORDS ARE USED AS
; THE LINKAGE BETWEEN THE ISR AND THE
; CONTROL PROGRAM.
;
VISR:
ISR: .BLKW 2 ;.ASCIZ "DXX"
BA: .BLKW 1 ;175610 ;BUS ADDRESS
VA: .BLKW 1 ;300 ;VECTOR ADDRESS
PRIOR: .BLKW 1 ;340 ;PRIORITY
PARAM1: .BLKW 1 ;-1 ;PARAM #1
PARAM2: .BLKW 1 ;-1 ;PARAM #2
PARAM3: .BLKW 1 ;-1 ;PARAM #3

IRDA: .BLKW 1 ;.WORD 0 ;INITIAL READ DATA ADDRESS
IXDA: .BLKW 1 ;.WORD 0 ;INITIAL XMIT DATA ADDRESS
SETTLE: .BLKW 1 ;.WORD 0 ;LINE SETTLE DELAY FLAG
IRCC: .BLKW 1 ;.WORD 0 ;INITIAL RCV CHAR COUNT
B2016: .BLKW 1 ;.WORD 0 ;ADDR OF BIN TO OCT TYPE ROUTINE
TIME: .BLKW 1 ;.WORD 0 ;TIMER
MODEA: .BLKW 1 ;.WORD 0 ;ADDR OF ITEMP PARAMS
TX. TERM: .BLKW 1 ;.WORD START ;ISR ENTRY ADDRESS
RX. TERM: .BLKB 1 ;.BYTE 000 ;TRANSMITER TERMINATING CHAR.
          .BLKB 1 ;.BYTE 012 ;RECEIVER TERMINATING CHAR.

FLAG: .BLKW 1
SWR: 177570
DISPLAY: 177570
START: .BLKW 1 ;NOP
        .BLKW 1 ;NOP
        .BLKW 1 ;NOP
        .BLKW 1 ;NOP
CONT.: .BLKW 1 ;NOP
        .BLKW 1 ;NOP
        .BLKW 1 ;NOP
FINI: .BLKW 1 ;MOV #340,PS ;LOCK OUT INTERRUPTS.
       .BLKW 1 ;MOV #ENTER,2(SP) ;SET FOR RETURN IF SW14=0
       .BLKW 1 ;JSR PC,SAVEDS ;GO SAVE YOUR REGISTERS.
       .BLKW 1 ;RTS PC ;EXIT

ENTER: .BLKW 1 ;JSR PC,RESTOS ;GO AND RESTORE REGISTERS
        .BLKW 1 ;MOV #-1,DELAY ;INDICATE DELAY FOR TX.
        .BLKW 1 ;JMP CONT. ;CONTINUE IN PROGRAM
        .END

```


GETPRM	003642	914#						
GETP3	003666	913	918	922#				
GETSTR=	104404	880	929	1027	1432#			
GETVA	003616	906#						
GOTIC	006040	1257	1264#					
HLTX	005262	1208	1211#					
IBUF	001604	803#	930	931	961	1029	1038	1040
IF	= 000100	1704#						
IFB	= 000010	1689#						
IOYVEC	= 000020	628#						
IRCC	011026	1737#						
IRCA	011020	962	1029*	1116	1734#			
ISX	011000	1094	1726#					
IV	010410	1623#						
IXDA	011022	963	1018*	1033*	1110	1735#		
KBDIN	= 104416	1100	1211	1442#				
LF	= 000001	1650	1714#					
LF	= 001537	780#	1269	1317				
LINE	= 000017	1707#						
LSTAT	= 000002	1661	1665	1710#				
LI	001374	729	731	734#	751	759	964	
LI	001426	743#	750					
LI	001452	741	753#					
MANBA	003404	840	845	848	852#			
MANIN	003316	834#						
MANINX	003460	862	866	869#				
MBA	010406	1622#	1642					
MFill	010155	970	1640#					
MM	= 001000	1701#						
MODEA	011034	1740#						
MSECS	006334	1046*	1335*	1337*	1341#			
MSC00	007152	745	755	1540#				
MSC001	007154	876	1540#					
MSC002	007177	897	1540#					
MSC003	007222	903	1540#					
MSC004	007241	906	1540#					
MSC005	007263	909	1540#					
MSC006	007277	914	1540#					
MSC007	007314	919	1540#					
MSC008	007331	924	1540#					
MSC009	007350	977	980	1540#				
MSC010	007372	1026	1540#					
MSC011	007411	1103	1540#					
MSC012	007430	1167	1540#					
MSC013	007441	1196	1540#					
MSC014	007452	1202	1540#					
MSC015	007462	1452	1540#					
MSC016	007506	1452	1540#					
MSC017	007506	1452	1540#					
MSC018	007506	1452	1540#					
MSC019	007506	1452	1540#					
MSC020	007570	1066	1540#					
MSC021	007616	1493	1540#					
MSC022	007637	1015	1540#					
MSC023	007653	1540#	1653					
MSC024	007767	1178	1540#					
MSC025	010003	1125	1540#					

MSG25	010021	1460	1540#						
MSG26	010060	1109	1540#						
MSG27	010106	1115	1540#						
MSG	010141	1503	1540#						
NOISR	003570	892	896#						
NOLC	004422	1049	1056#						
NOB	004070	959	966#						
NORTC	004524	1052	1063	1071#					
NULL	001524	685	771#						
OCT	006016	1259#	1262						
OUT	007040	1498	1512#						
OWI	000002	1107	1686#						
OWO	000001	1112	1655	1685#					
PARAM1	011012	1730#							
PARAM2	011014	1731#							
PARAM3	011016	1732#							
PARA1	010414	1625#	1649						
PARA2	010416	1626#							
PARA3	010420	1627#							
PND	000020	1557#	1586						
P10	010412	1624#							
PRIOR	011010	1729#							
PRTY0	000000	585#	1065	1075	1122	1494			
PRTY1	000040	586#							
PRTY2	000100	587#							
PRTY3	000140	588#	1152						
PRTY4	000200	589#							
PRTY5	000240	590#							
PRTY6	000300	591#							
PRTY7	000340	592#	819	1514					
PWI	100000	1553#	1571	1582					
PWRVEC	000024	629#							
QUES	001526	720	777#	878	927	1140	1314		
RECT	004202	979	1010#						
RECTRT	004210	1010	1011#						
RESEVEC	000010	624#							
REX	010556	1674#							
REX1	010514	1656	1681#						
RT	100000	1695#							
ROTS	000004	1659	1712#						
RSTART	004102	967	970#	1016					
RWAIT	104410	1436#							
RX.TER	011041	1744#							
SWT	000040	1705#							
SWTTSWT	104420	978	1446#						
SWTTLE	011024	1674#	1736#						
SWTCD	010000	1698#							
SWTACK	001070	528#	818	1074	1124	1492			
SWTART	011050	1750#							
SWTARTX	010504	1659#							
TEP	000400	1702#							
TEP	104414	819	1065	1075	1122	1152	1440#	1494	
USWR	104420	820	1444#						
UTIME	004384	1046#							
UXCC	004272	1020	1022	1029#					
WFO	010101	1500	1540#						

K04

MAINDEC-11-DZITA-D INTERPROCESSOR TEST PROGRAM MACY11 27(1006) 01-DEC-76 11:01 PAGE 51
DZITAD.P11 01-DEC-76 11:00 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0049

BOX	634#	696	772	814	825	831	872	988	1023	1079	1095	1135	1148	1229	1243
	1332	1448	1540	1561	1616	1638	1717								
HELLO	1#														
HLT	541#	1576	1591	1607											

. ABS. 011104 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DSKZ:DZITAD,DZITAD/SOL/CRF/EN:AMA=DZITAD.P11
RUN-TIME: 9 14 1 SECONDS
RUN-TIME RATIO: 107/25=4.1
CORE USED: 11K (21 PAGES)

