

IDENTIFICATION

Product Code:    MAINDEC-15-EOKA-D (D)  
Product Name:     ISZ TEST  
Date:             January 5, 1970  
Maintainer:       Diagnostic Group  
Author:            Edward P. Steinberger

(25)

1. ABSTRACT

The ISZ Test checks the operation of the ISZ instruction of the PDP-15. Various checks of the ISZ instruction are made, including ISZ of  $77777_8$  to  $0_8$  on all memory locations, and ISZ of random numbers stored in random memory locations from random memory locations. Errors are indicated to the operator via the teleprinter.

2. REQUIREMENTS

2.1 Equipment

Standard PDP-15 computer

2.2 Storage

The program uses all of 4K memory for the program or as a test area. When the program resides in upper 2K of memory, it occupies from locations 06440 to 07711. The random ISZ portion of the test tests all locations below 06440.

2.3 Preliminary Programs

Basic instruction tests

3. LOADING PROCEDURE

3.1 Method

- a. Put HRI tape of program in reader (high speed, if available).
- b. Set ADDRESS SWITCHES TO 00200; the BANK MODE switch on a 1.
- c. Depress and release READ-IN key.

4. STARTING PROCEDURE

4.1 Control Switch Settings

The following is a table of ACCUMULATOR SWITCH settings and their action on the program:

<u>AC Switch</u>	<u>Set As</u>	<u>Action</u>
0	1	Halt on error
	0	Don't halt on error
1	1	Don't print on errors
	0	Print errors
2	1	Ring bell on error
	0	Ring bell after N passes
3	1	Loop on current conditions
	0	Don't loop on current conditions
4	1	Loop on current test
	0	Don't loop on current test
5	1	Save initial error conditions of random ISZ
	0	Don't save initial error conditions of random ISZ
6	1	Vary location of ISZ instruction
	0	Don't vary location of ISZ instruction
7	1	Vary location of number incremented
	0	Don't vary location of number incremented
8	1	Vary number incremented
	0	Don't vary number incremented

(Switches 6, 7, 8 operate in conjunction with 5; 3 supercedes 4)

N is an arbitrary number (initially 20000<sub>8</sub> for random (ISZ's) which is controlled by the LAW-N instruction in location 07052 and may be changed at the operator's discretion.

#### 4.2 Starting Addresses

The starting address of the program is 00200. The restart addresses are 00200, 00244, 07000, 7052, and 7652 (see Section 5.3).

#### 4.3 Program and/or Operator Action

- a. Set ADDRESS SWITCHES to 00200
- b. Set ACCUMULATOR SWITCHES to desired positions (see Section 4.1). Normal setting is 510000.
- c. Depress I/O RESET
- d. Depress START

5. OPERATING PROCEDURE

5.1 Operational Switch Settings

See Section 4.1.

5.2 Subroutine Abstracts

None

5.3 Program and/or Operator Action

- a. To put the program in the 'scope mode, the ACCUMULATOR SWITCHES should be set to 270000, (don't halt, don't print, bell after N passes, loop on current number (location), loop on current test, save error conditions).
- b. To start program initially so that upper memory may be checked, start at location 00200.
- c. To start program initially so that lower memory may be checked without checking upper memory, start at location 00244.
- d. To restart program to check upper memory after program has moved, restart at 07652.
- e. To restart program to check lower memory after program has moved, restart at 07000.
- f. To restart program to check random ISZ's after program has moved, restart at 07052.

6. ERRORS

Unless AC switch 1 is a 1, all errors will be printed on the Teletype.

6.1 Error Halts and Description

<u>Location</u>	<u>Description</u>
00442	ISZ on upper memory did not skip
00504	Location in upper memory did not ISZ to 0
07466	ISZ on lower memory did not skip
07530	Location in lower memory did not ISZ to 0
07601	Random ISZ add failure

6.2 Error Recovery

#### 6.2.1 To Repeat Failure

If AC switch 0 is a 1, the computer will halt on an error. To recover and repeat the failure, reset AC switches 0 to 5 as necessary (see Section 4.1) and then depress CONTINUE key.

#### 6.2.2 Recovery with Random ISZ

The random ISZ portion of this test has special recovery features. AC switch 3, as with the other tests, may be used to put the program in the 'scope mode (loop on current conditions). If, however, it is desired to save the conditions of an error and vary the parameters which make up the current conditions, AC switches 5 to 8 may be used. If switch 5 is a 1, and an error occurs, the exact conditions which caused the error will be saved (location of ISZ instruction, location of number ISZ'd, number ISZ'd). By setting switches 6, 7, and/or 8 to a 1, any one or all of these conditions may be changed. Returning 6, 7, and/or 8 to 0 causes the original error condition for that switch to be used again. Thus it is possible to determine which condition(s) is causing the error. Switches 5-8 have no effect, if an error does not occur.

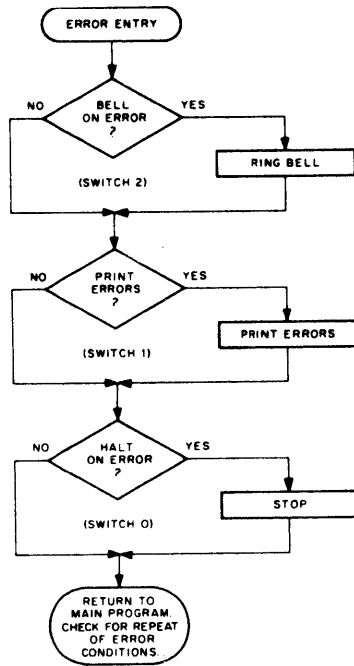
#### 6.2.3 Test Special Conditions with Random ISZ

To test special conditions in the random ISZ test, store the location of the ISZ instruction in 07112 (SAVE1) and 07115 (RAND1), store the location of the number ISZ'd in 07113 (SAVE2) and 07116 (RAND2) and store the number to be ISZ'd in 07114 (SAVE3) and 07117 (RAND3). Restart the program at location 07067 (PROCEED) with AC switches 3 and 5 to 1.

#### 6.2.4 Test Particular Memory Location

To test a particular memory location in the ISZ to 0 tests, store the number of the location to be tested in 00252 (PNTR2) if program is in low memory and an upper memory location is to be tested (if program is not in low memory, move it there by restarting computer at 07652 (MOVEDN), or in 07046 (PNTR1) if program is in high memory and a lower memory location is to be tested (if program is not in high memory, move it there by restarting computer at 00244 (MOVEUP). Whether the program is in lower or upper memory can be determined by the program counter while the program is running (it may be easier to stop the computer before looking at the PC). If the PC contains a number above 04000, the program is in upper 2K of memory and conversely. Restart computer at 00206 (ISZH1+6) or 07005 (ISZLOW+5) (as appropriate) with AC switches 3 and 4 a 1.

6.3 Error Switch Hierarchy



6.4 Error Timeout Examples

6.4.1 Increment Memory from -1 to 0

6.4.1.1 No Skip

ISZ DID NOT SKIP, LOC 001234

The above example shows that the number in location 1234 (-1) when incremented, did not skip.

6.4.1.2 Bad Add

ISZ ADD  
NUMBER AT ORIGINAL BAD

001234      77777      77777

The above example shows that the number in location 1234 (-1) was not incremented.

#### 6.4.2 Random ISZ Test

ISZ ADD

NUMBER AT	ORIGINAL	BAD	ISZ AT
001234	765435	765434	005763

The above example shows that a number in location 1234 (the number was 765435) was ISZ'd improperly producing a result of 765434 (carry from bit 17 to bit 16 was lost). The ISZ instruction was in location 5763.

### 7. MISCELLANEOUS

#### 7.1 Execution Time

Not applicable

### 8. PROGRAM DESCRIPTION

There are three basic portions to the program, a portion which tests upper memory to be sure that all memory locations can be incremented to 0 (ISZHI), a portion which tests lower memory to be sure that all memory locations can be incremented to 0 (ISZLOW), and a portion which assures that random numbers stored in random memory locations can be ISZ'd from random memory location properly (RANISZ).

#### 8.1 ISZHI

- a. The first function performed is that of initializing the bell counter and setting up for header printing should an error occur.
- b. Then a pointer is set to 04000 to allow the program to access the first location to be tested.
- c. The number  $77777_8$  is stored in the memory location indicated in the pointer and it is then incremented to 0. Checks are made to assure that the computer skipped and the number went to 0.
- d. A check is then made to see if the memory location should be tested again (switch 3). If so, c is repeated immediately. If not, then the address is incremented and then c is repeated.
- e. Step d is repeated until location  $07777_8$  has been tested, at which time a check is made to determine if the sequence should be repeated (switch 4). If so, the program goes back to b. If not, the program is moved to upper memory and then control is transferred to the ISZLOW portion of the program.

## 8.2 ISZLOW

This portion is essentially the same as ISZHI except that location 0<sub>8</sub> to 3777<sub>8</sub> are tested and the program is not moved when ISZLOW is completed.

## 8.3 RANISZ

- a. First the initialization of the loop counter and the header is set up.
- b. Then three random numbers are generated for the location of the ISZ, location of the number, and the number respectively.
- c. The number is stored in the appropriate memory location and an ISZ instruction is formed for that location and stored in its proper place.
- d. The ISZ instruction is then executed and the result is checked with a synthesized ISZ for proper addition. If no errors occurred, a check is made to see if the test should be repeated (switch 3). If not then the program returns to step b. If it should be repeated, the program returns to step c.
- e. If an error occurred, the conditions which caused it are indicated to the operator (unless they have been suppressed by AC switch 1) and then a check is made to see if the conditions should be saved (switch 5). If not, the program proceeds on as if no error occurred.
- f. If the error conditions should be saved, the program then determines which of the three variables should be changed, and repeats the test for the new variables and/or the old ones. The operator can always return to original error conditions by setting AC switch 6, 7, and 8 to 0. By setting AC switch 5 to 0, the operator can continue on with testing independent of the error, if he so desires.



```

.TITLE ISZ-15
/ISZ TEST
.FULL
.LOC 200
/ROUTINE TO ISZ UPPER 2K MEMORY FROM -1 TO 0 (OCCUPY LOW MEMORY)
/
ISZHI   LAW
        DAC CNTR2           /SET LOOP COUNTER
        LAC CONST3        /SET UP TO PRINT HEADER
        DAC ERROR4+11
        LAC STRT
        DAC PNTR2         /SET POINTER TO 4000
        CLA:CLL:CMA       /SET AC TO -1
        DAC* PNTR2        /STORE IN MEMORY
        ISZ* PNTR2        /INDEX LOCATION TO 0
        JMS ERROR2        /ERROR, COMPUTER DID NOT SKIP
        LAC* PNTR2        /GET C(MEMORY)
        SZA               /IS IT 0?
        JMS ERROR4        /NO, ERROR
        LAS
        AND MASK12
        SZA               /LOOP ON CURRENT NUMBER?
        JMP ISZHI+6       /YES
        LAS
        RTL
        SPA               /RING BELL?
        JMP .+6           /NO
        LAC CNTR2
        TAD ONE2
        DAC CNTR2
        SNA
        JMS BELL2
        LAC PNTR2        /NO, GET C (PNTR2)
        SAD UPLIM2       /IS IT 7777
        JMP .+4           /YES
        TAD ONE2        /NO, INCREMENT FOR NEXT
        DAC PNTR2       /LOCATION
        JMP ISZHI+6
        LAS
        AND MASK22
        SZA:CLA          /LOOP ON THIS TEST?
        JMP ISZHI+4     /YES
MOVEUP  LAW -A          /NO, GET READY TO
        JMS MOVE2        /MOVE PROGRAM
        ISZHI           /ORIGIN ADDRESS -LOWEST ADDRESS -LOW
        B               /DESTINATION ADDRESS -LOWEST ADDRESS -HIGH
        JMP ISZLOW      /GO TO PROGRAM IN UPPER MEMORY
        CNTR2           0
        PNTR2           0
        ONE2            1
        MASK12          40000
        MASK22          20000
        STRT            4000
        UPLIM2          7777
.EJECT

```

```

/SUBROUTINE TO MOVE THE PROGRAM (OCCUPY LOW MEMORY)
MOVE2      0
00260      000000
00261      040313      DAC TALLY2
00262      220260      LAC* MOVE2
00263      040314      DAC POINT3
00264      200260      LAC MOVE2
00265      340253      TAD ONE2
00266      040260      DAC MOVE2
00267      220260      LAC* MOVE2
00270      040315      DAC POINT4
00271      220314      LOOP2  LAC* POINT3
00272      260315      DAC* POINT4
00273      200314      LAC POINT3
00274      340253      TAD ONE2
00275      040314      DAC POINT3
00276      200315      LAC POINT4
00277      340253      TAD ONE2
00300      040315      DAC POINT4
00301      200313      LAC TALLY2
00302      340253      TAD ONE2
00303      040313      DAC TALLY2
00304      740200      SZA
00305      600271      JMP LOOP2
00306      200260      LAC MOVE2
00307      340253      TAD ONE2
00310      040260      DAC MOVE2
00311      754000      CLA:CLL
00312      620260      JMP* MOVE2
00313      000000      TALLY2  0
00314      000000      POINT3   0
00315      000000      POINT4   0
/
/Routine TO RING BELL -LOW MEMORY
BELL2      0
00316      000000
00317      760207      LAW 207
00320      100531      JMS TYPE2
00321      400200      XCT ISZHI
00322      040251      DAC CNTR2
00323      620316      JMP* BELL2
/
/Routine TO ISSUF CR-LF LOW MEMORY
CRLF2      0
00324      000000
00325      760215      LAW 215
00326      100531      JMS TYPE2
00327      760212      LAW 212
00330      100531      JMS TYPE2
00331      620324      JMP* CRLF2
.EJECT

```

```

/TYPE OUT THE CONTENTS OF THE AC IN OCTAL (LOW)
TYPOL0      0
DAC TEMP2
LAW -6
DAC TALLY2
LAC TEMP2
RAL
RAL
RTL
DAC TEMP2
AND SEVEN2
XOR ASKII2
JMS TYPE2
LAC TALLY2
TAD ONE2
DAC TALLY2
SNA
JMP* TYPOL0
LAC TEMP2
JMP .-14
TEMP2      0
SEVEN2     7
ASKII2     260
/
/
/ERROR MESSAGES (LOW MEMORY)
/
MESS12     .+1
311323    /I,S
332240    /Z,SP
304311    /D,I
304240    /D,SP
316317    /N,0
324240    /T,SP
323313    /S,K
311320    /I,P
254240    /.,SP
314317    /L,0
303240    /C,SP
377000    /R0
.EJECT

```

00375	00376	MESS22	.+1	
00376	215212		215212	/CR,LF
00377	311323		311323	/I,S
00400	332240		332240	/Z,SP
00401	301304		301304	/A,D
00402	304215		304215	/D,CR
00403	212212		212212	/LF,LF
00404	316325		316325	/N,U
00405	315302		315302	/M,B
00406	305322		305322	/E,R
00407	240301		240301	/SP,A
00410	324240		324240	/T,SP
00411	317322		317322	/O,R
00412	311307		311307	/I,G
00413	311316		311316	/I,N
00414	301314		301314	/A,L
00415	240240		240240	/SP,SP
00416	302301		302301	/B,A
00417	304240		304240	/D,SP
00420	377000		377000	/RO,

		/		
		/ERROR REPORTING SUBROUTINE 2 (LOW MEMORY)		
00421	000000	ERROR2	0	
00422	750004		LAS	
00423	742010		RTL	
00424	741100		SPA	/BELL ON ERROR?
00425	100316		JMS BELL2	
00426	750004		LAS	
00427	740010		RAL	
00430	741100		SPA	/PRINT ERRORS?
00431	600440		JMP .+7	/NO
00432	100324		JMS CRLF2	/YES, CRLF
00433	200360		LAC MESS12	
00434	100510		JMS TMESS2	/TYPE OUT "ISZ DID NOT SKIP, LOC"
00435	200252		LAC PNTR2	
00436	100332		JMS TYPOLO	/TYPE OUT NUMBER OF LOCATION
00437	100324		JMS CRLF2	/CR-LF
00440	750004		LAS	
00441	741100		SPA	/HALT ON ERROR?
00442	740040		XX	/YES
00443	620421		JMP* ERROR2	/EXIT
			.EJECT	

```

/ERROR MESSAGE REPORTING SUBROUTINE 4 (LOW MEMORY)
ERROR4      0
            LAS
            RTL
            SPA                /BELL ON ERROR?
            JMS RELL2          /YES
            LAS
            RAL
            SPA                /PRINT ERRORS?
            JMP .+26           /NO
            LAC MESS22
            JMS TMESS2         /TYPE OUT HEADER
            JMS CRLF2
            LAC CONST2
            DAC .-4
            LAC PNTR2
            JMS TYPOLO        /TYPE OUT LOCATION OF NUMBER
            LAW 240
            JMS TYPE2         /TYPE 5 SPACES
            JMS TYPE2
            JMS TYPE2
            JMS TYPE2
            JMS TYPE2
            CLA: CMA
            JMS TYPOLO        /TYPE ORIGINAL NUMBER
            LAW 240
            JMS TYPE2
            JMS TYPE2
            LAC* PNTR2
            JMS TYPOLO        /TYPE BAD RESULT
            JMS CRLF2        /CR-LF
            LAS
            SPA                /HALT ON ERROR?
            XX                /YES
            JMP* ERROR4       /EXIT
CONST2      JMP ERROR4+16
CONST3      LAC MESS22
            .EJECT

```

```

/MESRAGE TYPEOUT SUBROUTINE (LOW)
TMESS2      0
            DAC POINT3
00510      000000
00511      040314
00512      220314
            LAC* POINT3
00513      740020
            RAR
00514      742020
            RTR
00515      742020
            RTR
00516      742020
            RTR
00517      742020
            RTR
00520      100531
            JMS TYPE2
00521      540537
            SAD RUBOT2
00522      620510
            JMP* TMESS2
00523      220314
            LAC* POINT3
00524      100531
            JMS TYPE2
00525      440314
            ISZ POINT3
00526      540537
            SAD RUBOT2
00527      620510
            JMP* TMESS2
00530      600512
            JMP TMESS2+2
00531      000000
TYPE2      0
00532      500537
            AND RUBOT2
00533      700406
            TLS
00534      700401
            TSF
00535      600534
            JMP .-1
00536      620531
            JMP* TYPE2
RUBOT2     377
            .EJECT
```

```

07000
07000 760000
07001 047045
07002 207533
07003 047501
07004 147046
07005 754001
07006 067046
07007 467046
07010 107445
07011 227046
07012 740200
07013 107470
07014 750004
07015 507050
07016 740200
07017 607005
07020 750004
07021 742010
07022 741100
07023 607031
07024 207045
07025 347047
07026 047045
07027 741200
07030 107276
07031 207046
07032 547044
07033 607037
07034 347047
07035 047046
07036 607005
07037 750004
07040 507051
07041 750200
07042 607004
07043 607052
07044 003777
07045 000000
07046 000000
07047 000001
07050 040000
07051 020000

```

```

      .LOC 7000
/ROUTINE TO ISZ LOWER 2K MEMORY FROM 777777 TO 0 (OCCUPY HIGH MEMORY)
ISZLOW LAW
      DAC CNTR1
      LAC CONST5
      DAC ERROR3+11
      DZM PNTR1           /ZERO POINTER
      CLA:CLL:CMA        /SET AC TO -1
      DAC* PNTR1         /STORE -1 IN MEMORY
      ISZ* PNTR1         /INDEX LOCATION TO 0
      JMS ERROR1         /GO TO ERROR SUBROUTINE
      LAC* PNTR1         /GET CONTENTS OF MEMORY
      SZA                /IS IT 0?
      JMS ERROR3        /NO, ERROR
      LAS
      AND MASK11
      SZA                /LOOP ON CURRENT NUMBER?
      JMP ISZLOW+5       /YES
      LAS                /GET C (ACS)
      RTL                /MOVE 2 LEFT
      SPA                /RING BELL?
      JMP .+6           /NO
      LAC CNTR1
      TAD ONE1
      DAC CNTR1
      SNA                /IS CNTR 0?
      JMS BELL1         /YES, RING BELL
      LAC PNTR1         /NO, GET C (PNTR)
      SAD UPLIM1       /IS IT 7777
      JMP .+4           /YES
      TAD ONE1
      DAC PNTR1        /NO, INCREMENT FOR NEXT LOCATION
      JMP ISZLOW+5
      LAS
      AND MASK21
      SZA:CLA           /LOOP ON THIS TEST
      JMP ISZLOW+4     /YES
      JMP RANISZ       /NO
UPLIM1 3777
CNTR1  0
PNTR1  0
ONE1   1
MASK11 40000
MASK21 20000
      .EJECT

```

```

/RANDOM ISZ TEST (OCCUPIES HIGH MEMORY)
07052 760000 RANISZ LAW
07053 047045 DAC CNTR1 /SET UP TO COUNT LOOPS
07054 207533 LAC CONST5
07055 047545 DAC ERROR5+11
07056 107312 JMS GET1 /GET LOCATION OF ISZ
07057 047112 DAC SAVE1
07060 107323 JMS GET2 /GET LOCATION TO BE ISZ'D
07061 047113 DAC SAVE2
07062 107334 JMS COMPAR /COMPARE RAND1 AND RAND2
07063 607060 JMP .-3 /TO RE SURE THEY ARE DIFFERENT
07064 107224 JMS GEN3 /GET NUMRER TO BE ISZ'D
07065 047117 DAC RAND3
07066 047114 DAC SAVE3
07067 107123 PROCED JMS ISZTST /PERFORM AND CHECK THE ISZ
07070 107534 JMS ERRORS /ERROR, RETURN TO THIS INSTRUCTION
07071 750004 LAS /NO ERROR, RETURN HERE
07072 507050 AND MASK11
07073 740200 SZA /LOOP ON CURRENT NUMBERS
07074 607067 JMP PROCED /YES
07075 750004 LAS
07076 742010 RTL
07077 741100 SPA /RING BELL?
07100 607111 JMP .+11 /NO
07101 207045 LAC CNTR1 /YES
07102 347047 TAD ONE1
07103 047045 DAC CNTR1
07104 740200 SZA
07105 607111 JMP .+4
07106 107276 JMS BELL1
07107 407052 XCT RANISZ
07110 047045 DAC CNTR1
07111 607056 JMP RANISZ+4
07112 000000 SAVE1 0
07113 000000 SAVE2 0
07114 000000 SAVE3 0
07115 000000 RAND1 0
07116 000000 RAND2 0
07117 000000 RAND3 0
07120 771340 UPLIM3 -B /MINUS LOWER LIMIT OF UPPER PROGRAM
07121 440000 ISZCON ISZ
07122 017777 CONST1 17777
.EJECT

```



```

/ISZ TEST SETUP AND EXECUTION SUBROUTINE
ISZTST 0
07123 000000 LAC RAND3
07124 207117 DAC* RAND2 /STORE NUMBER TO BE ISZ'D
07125 067116 LAC ISZCON /FORM ISZ
07126 207121 TAD RAND2 /INSTRUCTION, AND
07127 347116 DAC* RAND1 /STORE IT, THEN
07130 067115 XCT* RAND1 /EXECUTE IT
07131 427115 NOP /FILLER
07132 740000 LAC RAND3 /SYNTHESIZE THE
07133 207117 TAD ONE1 /ISZ AND CHECK
07134 347047 SAD* RAND2 /TO SEE THAT THE
07135 567116 SKP /ANSWERS AGREE
07136 741000 JMP* ISZTST /THEY DON'T, EXIT
07137 627123 LAC ISZTST /INCREMENT ISZTST
07140 207123 TAD ONE1 /THE HARD WAY
07141 347047 DAC ISZTST
07142 047123 JMP* ISZTST
07143 627123

```

```

/SUBROUTINE TO MOVE THE PROGRAM (OCCUPY HIGH MEMORY)
MOVE1 0
07144 000000 DAC TALLY1
07145 047177 LAC* MOVE1
07146 227144 DAC POINT1
07147 047200 LAC MOVE1
07150 207144 TAD ONE1
07151 347047 DAC MOVE1
07152 047144 LAC* MOVE1
07153 227144 DAC POINT2
07154 047201 LOOP1 LAC* POINT1
07155 227200 DAC* POINT2
07156 067201 LAC POINT1
07157 207200 TAD ONE1
07160 347047 DAC POINT1
07161 047200 LAC POINT2
07162 207201 TAD ONE1
07163 347047 DAC POINT2
07164 047201 LAC TALLY1
07165 207177 TAD ONE1
07166 347047 DAC TALLY1
07167 047177 SZA
07170 740200 JMP LOOP1
07171 607155 LAC MOVE1
07172 207144 TAD ONE1
07173 347047 DAC MOVE1
07174 047144 CLA:CLL
07175 754000 JMP* MOVE1
07176 627144

/
TALLY1 0
POINT1 0
POINT2 0
.EJECT

```

```

/RANDOM NUMBER GENERATORS HIGH MEMORY
07202 000000 GEN1 0
07203 207211 LAC R1
07204 744010 RAL:CLL
07205 741400 SZL
07206 347212 TAD R1+1
07207 047211 DAC R1
07210 627202 JMP* GEN1
07211 000037 R1 000037
07212 000003 3
07213 000000 GEN2 0
07214 207222 LAC R2
07215 744010 RAL:CLL
07216 741400 SZL
07217 347223 TAD R2+1
07220 047222 DAC R2
07221 627213 JMP* GEN2
07222 000001 R2 000001
07223 000003 3
07224 000000 GEN3 0
07225 207252 LAC R3
07226 744010 RAL:CLL
07227 741400 SZL
07230 347253 TAD R3+1
07231 047252 DAC R3
07232 047254 DAC R3+2
07233 567255 SAD* R3+3
07234 741000 SKP
07235 627224 JMP* GEN3
07236 207255 LAC R3+3
07237 347047 TAD ONE1
07240 047255 DAC R3+3
07241 227255 LAC* R3+3
07242 047252 DAC R3
07243 741200 SNA
07244 607247 JMP ,+3
07245 207254 LAC R3+2
07246 627224 JMP* GEN3
07247 207710 LAC ADDR5
07250 047255 DAC R3+3
07251 607245 JMP , -4
.EJECT

```

07252	000000	R3	000000
07253	000003		3
07254	000000		0
07255	007256		R3+4
07256	000000		000000
07257	777775		777775
07260	056427		056427
07261	000175		000175
07262	000171		000171
07263	000137		000137
07264	000065		000065
07265	000037		000037
07266	000031		000031
07267	000023		000023
07270	000021		000021
07271	000015		000015
07272	000013		000013
07273	000005		000005
07274	000001		000001
07275	000000		000000

```

/ROUTINE TO RING BELL HIGH MEMORY
/

```

07276	000000	BELL1	0
07277	760207		LAW 207
07300	107702		JMS TYPE1
07301	407000		XCT ISZLOW
07302	047045		DAC CNTR1
07303	627276		JMP* BELL1

```

/
/ROUTINE TO ISSUE CR-LF HIGH MEMORY
/

```

07304	000000	CRLF1	0
07305	760215		LAW 215
07306	107702		JMS TYPE1
07307	760212		LAW 212
07310	107702		JMS TYPE1
07311	627304		JMP* CRLF1

.EJECT

```

/GET RANDOM NUMBER SUBROUTINES
GET1      0
          JMS GEN1           /GET RANDOM
          AND CONST1        /MASK
          DAC RAND1         /STORE
          TAD UPLIM3        /COMPARE TO SEE
          SMA                /IF IT IS IN THE PROGRAM
          JMP .-5           /IT IS, GENERATE ANOTHER
          LAC RAND1         /NO, IT ISN'T, EXIT
          JMP* GET1         /WITH NUMBER IN AC

GET2      0
          JMS GEN2
          AND CONST1
          DAC RAND2
          TAD UPLIM3
          SMA
          JMP .-5
          LAC RAND2
          JMP* GET2

/COMPARE RAND1 AND RAND2 SUBROUTINE
COMPAR    0
          LAC RAND1
          CMA
          TAD ONE1
          TAD RAND2
          SNA
          JMP* COMPAR
          LAC COMPAR
          TAD ONE1
          DAC COMPAR
          JMP* COMPAR
          .EJECT

```

```

07312  000000
07313  107202
07314  507122
07315  047115
07316  347120
07317  740100
07320  607313
07321  207115
07322  627312
07323  000000
07324  107213
07325  507122
07326  047116
07327  347120
07330  740100
07331  607324
07332  207116
07333  627323

```

```

07334  000000
07335  207115
07336  740001
07337  347047
07340  347116
07341  741200
07342  627334
07343  207334
07344  347047
07345  047334
07346  627334

```

```

/TYPE OUT THE CONTENTS OF THE AC IN OCTAL (HIGH)
07347 000000
07350 047372
07351 777772
07352 047177
07353 207372
07354 744010
07355 740010
07356 742010
07357 047372
07360 507373
07361 247374
07362 107702
07363 207177
07364 347047
07365 047177
07366 741200
07367 627347
07370 207372
07371 607355
07372 000000
07373 000007
07374 000260

TYP0HI 0
DAC TEMP1
LAW -6
DAC TALLY1
LAC TEMP1
RAL!CLL
RAL
RTL
DAC TEMP1
AND SEVEN1
XOR ASKII1
JMS TYPE1
LAC TALLY1
TAD ONE1
DAC TALLY1
SNA
JMP* TYP0HI
LAC TEMP1
JMP .-14

TEMP1 0
SEVEN1 7
ASKII1 260
.EJECT

```

## /ERROR MESSAGES (HIGH MEMORY)

07375	007376	MESS11	.+1	
07376	311323		311323	/I,S
07377	332240		332240	/Z,SP
07400	304311		304311	/D,I
07401	304240		304240	/D,SP
07402	316317		316317	/N,O
07403	324240		324240	/T,SP
07404	323313		323313	/S,K
07405	311320		311320	/I,P
07406	254240		254240	/..SP
07407	314317		314317	/L,O
07410	303240		303240	/C,SP
07411	377000		377000	/RO
07412	007413	MESS21	.+1	
07413	215212		215212	/CR,LF
07414	311323		311323	/I,S
07415	332240		332240	/Z,SP
07416	301304		301304	/A,D
07417	304215		304215	/D,CR
07420	212212		212212	/LF,LF
07421	316325		316325	/N,U
07422	315302		315302	/M,B
07423	305322		305322	/E,R
07424	240301		240301	/SP,A
07425	324240		324240	/T,SP
07426	317322		317322	/O,R
07427	311307		311307	/I,G
07430	311316		311316	/I,N
07431	301314		301314	/A,L
07432	240240		240240	/SP,SP
07433	302301		302301	/B,A
07434	304240		304240	/D,SP
07435	377000		377000	/RO
07436	007437	MESS31	.+1	
07437	240240		240240	/SP,SP
07440	311323		311323	/I,S
07441	332240		332240	/Z,SP
07442	301324		301324	/A,T
07443	215212		215212	/CR,LF
07444	377000		377000	/RO
			.EJECT	

```

/ERROR REPORTING SUBROUTINE 1 (HIGH MEMORY)
ERROR1  0
07445  000000  LAS
07446  750004  RTI
07447  742010  SPA
07450  741100  /BELL ON ERROR?
07451  107276  JMS BELL1  /YES
07452  750004  LAS
07453  740010  RAL
07454  741100  SPA
07455  607464  JMP .+7  /PRINT ERRORS
07456  107304  JMS CRLF1 /NO
07457  207375  LAC MESS11 /YES, CR-LF
07460  107657  JMS TMESS1 /TYPE OUT "ISZ DID NOT SKIP, LOC"
07461  207046  LAC PNTR1
07462  107347  JMS TYPOHI /TYPE OUT NUMBER OF LOCATION
07463  107304  JMS CRLF1 /CR-LF
07464  750004  LAS
07465  741100  SPA
07466  740040  XX
07467  627445  JMP* ERROR1 /HALT ON ERROR
/YES
/EXIT

```

```

/ERROR MESSAGE REPORTING SUBROUTINE 3 (HIGH MEMORY)
ERROR3  0
07470  000000  LAS
07471  750004  RTL
07472  742010  SPA
07473  741100  /BELL ON ERROR
07474  107276  JMS BELL1  /YES
07475  750004  LAS
07476  740010  RAL
07477  741100  SPA
07500  607526  JMP .+26  /PRINT ERRORS?
07501  207412  LAC MESS21 /NO
07502  107657  JMS TMESS1 /TYPE OUT HEADER
07503  107304  JMS CRLF1
07504  207532  LAC CONST4
07505  047501  DAC .-4
07506  207046  LAC PNTR1
07507  107347  JMS TYPOHI /TYPE OUT LOCATION OF NUMBER
07510  760240  LAW 240
07511  107702  JMS TYPE1 /5 SPACES
07512  107702  JMS TYPE1
07513  107702  JMS TYPE1
07514  107702  JMS TYPE1
07515  107702  JMS TYPE1
.EJECT

```

07516	750001	CLA:OMA	
07517	107347	JMS TYPOHI	/TYPE OUT ORIGINAL NUMBER
07520	760240	LAW 240	
07521	107702	JMS TYPE1	
07522	107702	JMS TYPE1	
07523	227046	LAC* PNTR1	
07524	107347	JMS TYPOHI	/TYPE BAD RESULT
07525	107304	JMS CRLF1	/CR-LF
07526	750004	LAS	
07527	741100	SPA	/HALT ON ERROR?
07530	740040	XX	/YES
07531	627470	JMP* ERROR3	/EXIT
07532	607506	JMP ERROR3+16	
07533	207412	LAC MESS21	
		.EJECT	

CONST4  
CONST5



```

/ERROR REPORTING SUBROUTINE 5 (HIGH MEMORY)
ERROR5  0
07534  100000  LAS
07535  750004  RTL
07536  742010  SPA /BELL ON ERROR
07537  741100  JMS RELL1 /YES
07540  107276  LAS
07541  750004  RAL
07542  740010  SPA /PRINT ERRORS?
07543  741100  JMP .+33 /NO
07544  607577  LAC MESS21
07545  207412  JMS TMESS1 /TYPE HEADER
07546  107657  LAC MESS31
07547  207436  JMS TMESS1 /TYPE "ISZ AT"
07550  107657  LAC CONST6
07551  207651  DAC .-5
07552  347545  LAC RAND2
07553  207116  JMS TYPOHI /TYPE LOCATION OF NUMBER
07554  107347  LAW 240
07555  760240  JMS TYPE1 /5 SPACES
07556  107702  JMS TYPE1
07557  107702  JMS TYPE1
07560  107702  JMS TYPE1
07561  107702  JMS TYPE1
07562  107702  LAC RAND3
07563  207117  JMS TYPOHI /TYPE ORIGINAL NUMBER
07564  107347  LAW 240
07565  760240  JMS TYPE1
07566  107702  JMS TYPE1
07567  107702  LAC* RAND2
07570  227116  JMS TYPOHI /TYPE BAD NUMBER
07571  107347  LAW 240
07572  760240  JMS TYPE1
07573  107702  LAC RAND1
07574  207115  JMS TYPOHI /TYPE LOCATION OF ISZ
07575  107347  JMS CRLF1
07576  107304  LAS
07577  750004  SPA /HALT ON ERROR?
07600  741100  XX /YES
07601  740040  RACK LAS
07602  750004  AND MASK31
07603  507645  SNA /SAVE ERROR CONDITIONS?
07604  741200  JMP* ERROR5 /NO
07605  627534  .EJECT

```

07606	750004	ISZLOC	LAS	
07607	507646		AND MASK41	
07610	741200		SNA	/VARY LOCATION OF ISZ?
07611	607616		JMP .+5	/NO
07612	107312		JMS GET1	/YES, GET ANOTHER ADDRESS
07613	107334		JMS COMPAR	/IS RAND1=RAND2
07614	607612		JMP .-2	/YES, TRY AGAIN
07615	607620		JMP OPLOC	/ALL OK, GO ON
07616	207112		LAC SAVE1	/TRANSFER C(SAVE1)
07617	047115		DAC RAND1	/TO RAND1
07620	750004	OPLOC	LAS	
07621	507647		AND MASK51	
07622	741200		SNA	/VARY LOCATION OF OPERAND?
07623	607630		JMP .+5	/NO
07624	107323		JMS GET2	/YES, GET OP ADDRESS
07625	107334		JMS COMPAR	/IS RAND1=RAND2?
07626	607624		JMP .-2	/YES, TRY AGAIN
07627	607632		JMP OPNUM	/ALL OK, GO ON
07630	207113		LAC SAVE2	/TRANSFER C(SAVE2)
07631	047116		DAC RAND2	/TO RAND2
07632	750004	OPNUM	LAS	
07633	507650		AND MASK61	
07634	741200		SNA	/VARY OPERAND?
07635	607640		JMP .+3	/NO
07636	107224		JMS GEN3	/YES
07637	607641		JMP .+2	
07640	207114		LAC SAVE3	
07641	047117		DAC RAND3	
07642	107123	TRYDIF	JMS ISZTST	/PERFORM AND CHECK THE ISZ
07643	607535		JMP ERROR5+1	/ERROR RETURNS HERE
07644	607602		JMP BACK	/NO ERROR RETURNS HERE
07645	010000	MASK31	10000	
07646	004000	MASK41	4000	
07647	002000	MASK51	2000	
07650	001000	MASK61	1000	
07651	607553	CONST6	JMP ERROR5+17	
			.EJECT	

```

/ROUTINE TO MOVE PROGRAM TO LOWER MEMORY AND START ISZHI
MOVEDN   LAC  -A           /GET READY TO
          JMS MOVE1       /MOVE A WORDS
          R                /FROM HIGH TO
          ISZHI           /LOW MEMORY
          JMP ISZHI       /THEN TRANSFER CONTROL TO ISZHI

/MESSAGE TYPFOUT SUBROUTINE (HIGH)
TMESS1   0
          DAC POINT1
          LAC* POINT1
          RAR
          RTR
          RTR
          RTR
          RTR
          JMS TYPE1
          SAD RUBOT1
          JMP* TMESS1
          LAC* POINT1
          JMS TYPE1
          SAD RUBOT1
          JMP* TMESS1
          LAC POINT1
          TAD ONE1
          DAC POINT1
          JMP TMESS1+2
TYPE1    0
          AND RUBOT1
          TLS
          TSF
          JMP  -1
          JMP* TYPE1
ADDRS   R3+4
RUBOT1  377
A=RUBOT2-ISZHI+1+RUBOT1-ISZLOW+1
B=ISZLOW-RUBOT2+ISZHI-1
        .END
        NO ERROR LINES

```

07652	776526
07653	107144
07654	006440
07655	000200
07656	600200
07657	000000
07650	047200
07661	227200
07662	740020
07663	742020
07664	742020
07665	742020
07666	742020
07667	107702
07670	547711
07671	627657
07672	227200
07673	107702
07674	547711
07675	627657
07676	207200
07677	347047
07700	047200
07701	607661
07702	000000
07703	507711
07704	700406
07705	700401
07706	607705
07707	627702
07710	007256
07711	000377
	001252
	006440
	000000

A	001252
ADDRS	07710
ASKI11	07374
ASKI12	00357
R	006440
RACK	07602
RELL1	07276
RELL2	00316
CLOF	700004
CLON	700044
CLSF	700001
CNTR1	07045
CNTR2	00251
COMPAR	07334
CONST1	07122
CONST2	00506
CONST3	00507
CONST4	07532
CONST5	07533
CONST6	07651
CRLF1	07304
CRLF2	00324
ERROR1	07445
ERROR2	00421
ERROR3	07470
ERROR4	00444
ERROR5	07534
GEN1	07202
GEN2	07213
GEN3	07224
GET1	07312
GET2	07323
ISZCON	07121
ISZHI	00200
ISZLOC	07606
ISZLOW	07000
ISZTST	07123
KRR	700312
KSF	700301
LOOP1	07155
LOOP2	00271
MASK11	07050
MASK12	00254
MASK21	07051
MASK22	00255
MASK31	07645
MASK41	07646
MASK51	07647
MASK61	07650
MESS11	07375
MESS12	00360
MESS21	07412
MESS22	00375
MESS31	07436
MOVEDN	07652

MOVEUP	00244
MOVE1	07144
MOVE2	00260
ONE1	07047
ONE2	00253
OPLOC	07620
OPNUM	07632
PCF	700202
PNTR1	07046
PNTR2	00252
POINT1	07200
POINT2	07201
POINT3	00314
POINT4	00315
PROCD	07067
PSA	700204
PSB	700244
PSF	700201
RAND1	07115
RAND2	07116
RAND3	07117
RANISZ	07052
RCF	700102
RRR	700112
RSA	700104
RSB	700144
RSF	700101
RUBOT1	07711
RUBOT2	00537
R1	07211
R2	07222
R3	07252
SAVE1	07112
SAVE2	07113
SAVE3	07114
SEVEN1	07373
SEVEN2	00356
STRT	00256
TALLY1	07177
TALLY2	00313
TCF	700402
TEMP1	07372
TEMP2	00355
TLS	700406
TMESS1	07657
TMESS2	00510
TRYDIF	07642
TSF	700401
TYPE1	07702
TYPE2	00531
TYP0H1	07347
TYP0L0	00332
UPLIM1	07044
UPLIM2	00257
UPLIM3	07120

ISZHI	00200
MOVEUP	00244
CNTR2	00251
PNTR2	00252
ONE2	00253
MASK12	00254
MASK22	00255
STRT	00256
UPLIM2	00257
MOVE2	00260
LOOP2	00271
TALLY2	00313
POINT3	00314
POINT4	00315
BELL2	00316
CRLF2	00324
TYPOLO	00332
TEMP2	00355
SEVEN2	00356
ASKII2	00357
MESS12	00360
MESS22	00375
ERROR2	00421
ERROR4	00444
CONST2	00506
CONST3	00507
TMESS2	00510
TYPE2	00531
RUBOT2	00537
A	001252
B	006440
ISZLOW	07000
UPLIM1	07044
CNTR1	07045
PNTR1	07046
ONE1	07047
MASK11	07050
MASK21	07051
RANISZ	07052
PROCED	07067
SAVE1	07112
SAVE2	07113
SAVE3	07114
RAND1	07115
RAND2	07116
RAND3	07117
UPLIM3	07120
ISZCON	07121
CONST1	07122
ISZTST	07123
MOVE1	07144
LOOP1	07155
TALLY1	07177
POINT1	07200
POINT2	07201

GEN1	07202
R1	07211
GEN2	07213
R2	07222
GEN3	07224
R3	07252
BELL1	07276
CRLF1	07304
GET1	07312
GET2	07323
COMPAR	07334
TYP0HI	07347
TEMP1	07372
SEVEN1	07373
ASKII1	07374
MESS11	07375
MESS21	07412
MESS31	07436
ERROR1	07445
ERROR3	07470
CONST4	07532
CONST5	07533
ERROR5	07534
RACK	07602
ISZLOC	07606
OPLOC	07620
OPNUM	07632
TRYDIF	07642
MASK31	07645
MASK41	07646
MASK51	07647
MASK61	07650
CONST6	07651
MOVEDN	07652
TMESS1	07657
TYPE1	07702
ADDRS	07710
RUR0T1	07711
CLSF.	700001
CLOF	700004
CLON	700044
RSF1	700101
RCF	700102
RSA	700104
RRB	700112
RSB	700144
PSF	700201
PCF	700202
PSA	700204
PSB	700244
KSF	700301
KRR	700312
TSF	700401
TCF	700402
TLS	700406

MAINDEC EVALUATION REQUEST

After sufficient familiarization with the operation and documentation of this MAINDEC, please indicate your assessment of the following areas and return this form to Digital Equipment Corporation.

IDENTIFICATION: MAINDEC NO. \_\_\_\_\_ Program Title \_\_\_\_\_

USAGE: Used by: Field Service  Production  Other \_\_\_\_\_

Frequency of Usage: Daily  Weekly  Monthly

SUGGESTIONS FOR IMPROVEMENT

1. Are the program loading and operating instructions: clear? , incomplete? , difficult to follow?
2. Do the error reports and program documentation provide sufficient diagnostic information. in all cases? , in most cases? , in very few cases? . Suggestions for improvement:

---

---

---

3. Is the program effective in isolating malfunctions: in all cases? , in most cases? , in very few cases? . Would additional Scope loops or Switch Register control be helpful? \_\_\_\_\_  
Suggestions for improvement:

---

---

---

4. Does the program ever fail to detect malfunctions exposed by other software? \_\_\_\_\_  
Were Margins used? \_\_\_\_\_ Please describe malfunction in detail:

---

---

---

5. Does the program ever report non-existent malfunctions? \_\_\_\_\_  
Please indicate erroneous report and any pertinent operating conditions:

---

---

---

6. Does this MAINDEC ever expose malfunctions in the Central Processor or other peripheral units not detected by the appropriate MAINDEC? \_\_\_\_\_  
Please describe malfunction and MAINDEC(S) used:

---

---

---

7. Does the document provide a general understanding of the functional programming requirements of the system? Good , Fair , None . Would a general description of programming requirements increase the effectiveness of this MAINDEC? \_\_\_\_\_

Remarks:

---

---

---



..... Fold Here .....

..... Do Not Tear - Fold Here and Staple .....

FIRST CLASS  
PERMIT NO. 33  
MAYNARD, MASS.

BUSINESS REPLY MAIL  
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

**digital**

Digital Equipment Corporation  
Diagnostic Programming Group  
146 Main Street, Building 12  
Maynard, Massachusetts 01754

