HP 3000 SERIES II
COMPUTER SYSTEM
MANUAL OF STAND-ALONE DIAGNOSTICS

STAND-ALONE
HP 30012A EXTENDED INSTRUCTION
SET DIAGNOSTIC

Diagnostic D431

HEWLETT hp PACKARD

30012-90001
Printed in U.S.A.

5 76

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

TABLE OF CONTENTS

I. INTRODUCTION

The Stand-Alone HP30012A Extended-Instruction Firmware Diagnostic for the HP 3000 Series II constitutes both the Floating Point and the Decimal diagnostic programs. Section 1 is the Floating Point Diagnostic and Section 2 is the Decimal Diagnostic. The programs are designed to verify that each instruction, both Floating Point and Decimal, is executed correctly and that any residues derived are verified for correctness. A detailed description of the test for both the Floating Point and decimal is included in the subsequent sections of this manual.

The Extended-Precision Floating Point instructions tested are:

%020410	EADD	(EFP Add)
%020411	ESUB	(EFP Subtract)
%020412	EMPY	(EFP Multiply)
%020413	EDIV	(EFP Divide)
%020414	ENEG	(EFP Negate)
%020415	ECMP	(EFP Compare)

The Decimal instructions tested are:

%020601	DMPY	(Double Multiply)
%020602	CVAD	(Convert ASCII to Decimal)
%020603	CVDA	(Convert Decimal to ASCII)
%020604	CVBD	(convert Binary to Decimal)
%020605	CVDB	(Convert Decimal to Binary)
%020606	SLD	(Shift Left Decimal)
%020607	NSLD	(Normalizing Shift Left Decimal)
%020610	SRD	(Shift Right Decimal)
%020611	ADDD	(Add Decimal)
%020612	CMPD	(Compare Decimal)
%020613	SUBD	(Subtract Decimal)
%020614	MPYD	(Multiply Decimal)

II. MINI-OPERATING INSTRUCTIONS

A. Operating Instructions

1. Cold Load Diagnostic File #(associated with D431A) from Non-CPU Cold Load tape.

2. Depress "Carriage Return" at the Console.

3. Respond to the dialogue at the Console.

B. Switch-Register Options

BIT	FUNCTION
***	**********
0	SELECT EXTERNAL SWITCH REGISTER
1	SELECT SECTION
2-6	SPARE
7	OUTPUT TO LINE PRINTER (IF CONFIGURED IN SDUP)
8	HALT ON ERROR COUNT REACHED
9	SUPPRESS NON-ERROR MESSAGES
10	SUPPRESS ERROR MESSAGES
11	LOOP ON LAST STEP
12	HALT ON ERROR
13	HALT ON END OF STEP
14	HALT ON END SECTION
15	HALT AFTER COMPLETE PROGRAM CYCLE

C. Section Switch-Register

BIT	FUNCTION
***	**********
0	Re-Configure
1	Select Extended Floating Point Diagnostic
2	Select Decimal Diagnostic
3-15	Spares

D. Halt Assignments

HALT(%)	FUNCTION
*******	**********
0	Spare
1	Irrecoverable Unexpected Trap.
2	Irrecoverable Unexpected Trap in Trap STT 1, 16, 24 and 25 (not in Extended-Instruction Set).
3	Halt on Error Count Reached
4	Spare
5	Section Select Switch-Register Entry Halt
6	External Switch-Register Entry Halt
7	Restore External Switch-Register Entry Halt
10	Irrecoverable Error after an execution of Instruction Set.

With this halt, TOS contains the error code as follows:

ERROR CODE(TOS)	DEFINITION
1	DB Changed
2	Code Segment# in Status (8:8) Changed
3	S Bank Changed
4	Q Changed

HALT(%)	FUNCTION
*******	**********
11	Irrecoverable Halts for "DB Changed" in the following Trap STT#: *) Trap#25 (User Trap) *) Trap#24 (Stack Overflow) *) Trap#16 (Unimplemented Instr.) *) Trap# 1 (Bounds Violation) For above halt, the Trap STT# is displayed as an error code in TOS.
12	Recoverable Error Halt If Switch-Register Bit (12) was selected. TOS(Ø:8):=ERROR CODE;TOS(8:8);=STEP#.
13	Halt after step
14	Half after section
15	Halt after complete Program Cycle
16-17	Spares

III. REQUIREMENTS

A. Hardware

Minimum hardware required to run this diagnostic will be a HP3000/30 Series II Model 5, or Model 7.

B. Software

The Stand-Alone Diagnostic Utility Program (SDUP) is required to create the Stand-Alone Diagnostic Tape. This Cold Loadable tape is comprised of Cold Load Program, the Relocatable Loader, and one or more diagnostic Programs including the Stand-Alone Extended-Instruction Set Diagnostic program. All programs are coded in System Programming Language (SPL/3000). For detailed description of SDUP, see System Diagnostic Utility Manual (Part#03000-90125).

IV. DETAILED OPERATING INSTRUCTIONS

A. Operating Instructions

The following are the instructions for loading, executing, and configuring the Stand-Alone HP30012A Extended-Instruction Set Diagnostic.

1. Cold Load by entering %3006 into the 30013-60013 Control Panel and simultaneously depress "LOAD" and "ENABLE" switches on the 30003-60013 control panel. It will pause.

2. Select an appropriate Diagnostic File# (associated with the Extended Instruction Set Diagnostic) and enter this number via the Switch Register. Depress "RUN". The tape will read the remaining records and will rewind at the end of last record read. (It should be noted that the Cold Load tapes supplied are identified by file names and their respective file position on the tape).

3. The HP30012A Extended-Instruction Set Diagnostic is now executable.

4. Depress "RETURN" Key at the Console to respond to Speed-Sense. Upon completing the previous operation, the program prints the diagnostic header and then requests necessary parameters to begin its execution cycle.

B. Options

Under Stand-Alone HP30012A Extended-Instruction Set Diagnostic Program, an operator can control the test sections or steps to be executed. The operator, via the Switch-Register option, can control halts after sections, steps, or upon program completion; control suppression os error and/or non-error messages; and control looping on a specific test step, or section. These control options may be selected when there is a request for a specific parameter entry. All configuration requests are made via the Console.

1. The options associated with each bit of the Switch Register entry request for the following message are the same as those described in Section II.B :

"Q01 ENTER SWREG. SELECT OPTIONS"

The usage description of bits (0 and 1) for this option is as follows:

BIT#0	BIT#1	FUNCTION
*****	*****	**********
0	0	Uses previously configured values.
0	1	Uses previously configured values.
1	0	The program uses whatever options currently selected on the Switch-Register of the control Panel or it will use whatever options entered by the request message "Q03 RESTORE SWREG.SELECT OPTIONS".
1	1	If this option is selected, it suggests possible reconfiguration (see bit#0 of Section IV.B.2).

2. The options associated with each bit for the following message are the same options described in Section II.e :

"Q02 ENTER SECTION SELECT OPTIONS"

If Bit#0 is not selected (0), the program requests no further parameters and continues execution using previously configured values.

If Bit#0 is selected (1), the program requests the restoration of External Switch-Register Options, requests maximum error and pass numbers from the following respective messages:

a. "Q04 ENTER MAXIMUM ERROR COUNT#="

B. "Q05 ENTER PASS NUMBER="

The maximum error count and pass number is 999 (decimal) for each respectively.

(Note that in all previous messages, the quotation marks are for clarity only)

C. Halt and Message Tables

1. Halt Assignments

When a program halts, an instruction is displayed in the Current Instruction Register (CIR) of the 30003-60013 Control Panel. The Halt instruction is displayed in the CIR register as:

CIR=(0 011 000 011 11X XXX)

where: X's is the Halt#(octal).

See Halt Assignment Table as described in Section II.D.

2. Message Formats

There are basically four types of message classifications: D,E, P, and Q classes.

a. D-class

Messages which describe program properties. Some operator intervention is necessary.

b. E-class

Messages related to errors within test steps. Some operator intervention is necessary.

c. P-class

Messages which describe the test completion of a Section or a step or an indicator for a certain tested properties.

d. Q-class

Inquiry messages by the program for the parameter entry. Operator intervention is required.

2.1 Message Descriptor

2.1.1 D-Types Messages

2.1.1.1 DØ1 HP30012A EXTENDED INSTRUCTION SET DIAGNOSTIC (D431X.YY.ZZ)

:This is the header information for this diagnostic program; where

X=Version Number
YY=Update Number
ZZ=Fix Number

2.1.1.2 DØ2 XXX PASSES COMPLETED

:This message indicates that an entry value from Section IV.B.2b. for a pass number has been completed.

2.1.2 Q-Types Messages

2.1.2.1 QØ1 SELECT SWREG OPTIONS

:This message implies a request for any of those options available in Section II.B.

2.1.2.2 QØ2 SELECT SECTION SWREG OPTIONS

:This message implies a request for any of those options available in Section II.C.

2.1.2.3 QØ3 RESTORE SWREG OPTIONS

:This message implies a request for any of those options availiable in Section II.B.

2.1.2.4 QØ4 ENTER MAXIMUM ERROR NUMBER=

:This message implies a request for maximum error number in decimal (maximum=999).

2.1.2.5 QØ5 ENTER PASS NUMBER=

:This message implies a request for maximum pass number (Module) in decimal (Maximum=999).

2.1.3 P-Type Messages

2.1.3.1 PØ1 STEP XXX COMPLETED

:This message implies that the test step (XXX) under execution was just completed. This message is printed only when Bit#13 of the options table in Section II.B. is on (1) and Bit #9 is not on (Ø).

2.1.3.2 P02 END OF SECTION X

This message implies that the test section (X) under execution was just completed. This message is printed only when Bit #14=1 and Bit #9=0 on the Switch-Register.

2.1.4 E-Type Messages (Extended Floating Point Instruction Set)

E-Type messages may constitute anywhere from 3 to 4 lines on the console. An error definition associated with each respective error number is as follows: (all step and error numbers are in Decimal).

2.1.4.1 E1 AAAA ERR IN STEP BBB
OPERATION ERROR
TARG=%XXXXXX,%XXXXXX,%XXXXXX,%XXXXXX
RESU=%YYYYYY,%YYYYYY,%YYYYYY,%YYYYYY

This message implies that an erroneous operation had taken place. As a result, comparison with the expected result is in conflict.

where: AAAA=Floating Point Instruction in reference.
BBBB=Test step number in which the error had occurred.
X's=The result of actual Floating Point Instruction Operation.
Y's=The expected result of the operation.

2.1.4.2 E2 AAAA ERR IN STEP BBB
Z WD(S) STACK DELETE ERROR
STACK=%XXXXXX
SHOULD=%YYYYYY

This message implies that an erroneous number of words from the stack had been deleted during the Floating Point Instructions operation.

where: AAAA=Floating Point Instruction in reference
BBBB=Test step number in which the error had occurred.
Z=1,2,or3. Relative to the type of instruction in operation.

1 = ENEG
2 = ECMP
3 = EADD,ESUB,EMPY,andEDIV

X's=Actual stack pointer relative to DB.
Y's+Expected stack pointer relative to DB.

2.1.4.3 E3 AAAA ERR IN STEP BBB
ERROR OPND1 CHANGED
OPND1=%XXXXXX,%XXXXXX,%XXXXXX,%XXXXXX
SHOULD=%YYYYYY,%YYYYYY,%YYYYYY,%YYYYYY

This message implies that an OPND1(u) which contains the 4 word operand had changed during the operation.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred
X's=4 word operand (u)
Y's=Expected 4 word operand

2.1.4.4 E4 AAAA ERR IN STEP BBB
ERROR OPND2 CHANGED
OPND2=%XXXXXX,%XXXXXX,%XXXXXX,%XXXXXX
SHOULD=%YYYYYY,%YYYYYY,%YYYYYY,%YYYYYY

This message implies that an OPND2(v) which contains the 4 word operand had changed during the operation.

where: AAAA=Floating Point Instruction in reference.
BBB=Test step number in which the error had occurred.
X's=4 word operand (v)
Y's=Expected 4 word operand

2.1.4.5 E5 AAAA ERR IN STEP BBB
ZZZ ERROR
ZZZ=SSS
SHOULD=SSS

This message implies that the Condition Code [Status (6:2)] is different after the operation than expected.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred
ZZZ=CCA for: EADD
ESUB
EMPY
EDIV
ENEG
ZZZ=CCC for: ECMP
SSS=Either: UNC=unchanged(3)
CCE=equal(2)
CCL=less(1)
CCG=greater(Ø)

2.1.4.6 E6 AAAA ERR IN STEP BBB
UNEXPECTED TRAP ERROR
TRAP=XX

This message implies that the trap had occurred where one was not expected.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred.
XX=An erroneous trap STT number (decimal)

2.1.4.7 E7 AAAA ERR IN STEP BBB
EXPECTED OVERFLOW TRAP FAILED
TRAP=XX
SHOULD=25
EXPECTED OVERFLOW TRAP CODE ERROR
TRPCODE=%YY
SHOULD=%10

This message implies that the expected trap STT#25 (User Trap) for overflow did not occur. The message, also, implies that the expected Trap Code of %10 did not occur.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred
XX=Ø, No trap occurred
≠Ø, Wrong SST# XX to which it trapped
YY=An erroneous trap code

2.1.4.8 E8 AAAA ERR IN STEP BBB
EXPECTED UNDERFLOW TRAP FAILED
TRAP=XX
SHOULD=25
EXPECTED UNDERFLOW TRAP CODE ERROR
TRPCODE=%YY
SHOULD=%11

This message implies that the expected trap STT# 25 (user Trap) for UNDERFLOW did not occur. The message, also, implies that the expected trap code of %11 did not occur.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred.
XX=Ø, No trap occurred
≠Ø, Wrong SST# XX to which it trapped
YY=An erroneous trap code

2.1.4.9 E9 AAAA ERR IN STEP BBB
STATUS(OVFL) ERR FOR TRAP(ENABLED) EXPECTED
STATUS=%XXXXXX
SHOULD=%YYYYYY

This message implies that for an expected Trap STT# 25 (User Trap) for either OVFL, UNFL, or DZERO with trap enabled (STA(2:1)=1), the STA(4:1) was not zero (Ø) after the operation.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred
X's=Actual Status
Y's=Expected Status

2.1.4.10 E1Ø AAAA ERR IN STEP BBB
STATUS (OVFL) ERR FOR TRAP (DISABLED) EXPECTED
STATUS=%XXXXXX
SHOULD=%YYYYYY

This message implies that for an expected Trap STT# 25 (User Trap) for either OVFL, UNFL, or DZERO with trap disabled (STA(2:1)=Ø), the STA(4:1) was not one (1) after the operation.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error has occurred.
X's= Actual Status
Y's=Expected Status

2.1.4.11 E11 AAAA ERR IN STEP BBB
EXPECTED DIVIDE BY ZERO TRAP FAILED
TRAP=XX
SHOULD=25
EXPECTED (DIVIDE BY Ø) TRAP CODE ERROR
TRPCODE=%YY
SHOULD=%12

This message implies that the expected Trap STT# 25 (User Trap) for DIVIDE BY ZERO did not occur. The message, also, implies that the expected trap code of %12 did not occur.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred.
XX=Ø, No trap occurred
≠Ø, Wrong STT# XX to which it trapped
YY=An erroneous trap code

2.1.4.12 E12 AAAA ERR IN STEP BBB
(TOS) ERROR
STACK=%XXXXXX
SHOULD=%YYYYYY

This message implies that the TOS content after the Floating Point Instruction operations is different than expected.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred.
X's=TOS content after the instruction operation
Y's=Expected TOS content

2.1.4.13 E13 AAAA ERR STEP BBB
EXPECTED STACK OVFL TRAP ERROR
TRAP=XX
SHOULD=24

This message implies that the trap for STACK OVFL did not occur in Trap STT# 24, but rather in Trap STT# XX.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred.
XX=Ø, No trap occurred
=Ø, Wrong STT# XX to which it trapped

2.1.4.14 E14 AAAA ERR IN STEP ZZZ
EXPECTED BOUNDS VIOL. TRAP ERROR
TRAP=XX
SHOULD=Ø1

This message implies that the trap for BOUNDS VIOLATION did not occur in Trap STT# Ø1, but rather in Trap STT# XX.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred.
XX=Ø, No trap occurred.
≠Ø. Wrong STT# XX to which it trapped.

2.1.4.15 E15 AAAA ERR IN STEP BBB
X-VALUE CHANGED
ACTUAL=%XXXXXX
SHOULD=%YYYYYY

This message implies that the X-Value after the Floating Point Instruction Operation is different than expected.

where: AAAA=Floating Point Instruction in reference
BBB=Test step number in which the error had occurred
X's=X-Value after the instruction operation
Y's=X-Value before the instruction operation

2.1.5 DECIMAL INSTRUCTION SET ERROR MESSAGES

All steps and error numbers are in octal.

2.1.5.1 E30 DL GOT CHANGED IN STEP XXX

2.1.5.2 E31 Z GOT CHANGED IN STEP XXX

2.1.5.3 E32 WRONG S IN STEP XXX
S-Q=%AAAAAA
SHOULD=%BBBBBB

2.1.5.4 E33 WRONG RESULT IN STEP XXX
RESULT=AAAA AAAA AAAA AAAA AAAA AAAA AAAA AAAA
SHOULD=BBBB BBBB BBBB BBBB BBBB BBBB BBBB BBBB

2.1.5.5 E34 WRONG INDEX IN STEP XXX
INDEX=%AAAAAA
SHOULD=%BBBBBB

2.1.5.6 E35 WRONG STATUS IN STEP XXX
STATUS=A AAA AAA AAA AAA AAA
SHOULD=B BBB BBB BBB BBB BBB

2.1.5.7 E36 WRONG TOS IN STEP XXX

2.1.5.8 E37 UNEXPECTED TRAP IN STEP XXX
TRAP SST=AA

2.1.5.9 E40 WRONG TRAP SST IN STEP XXX
TRAP SST=AA
SHOULD=BB

2.1.5.10 E41 WRONG TRAP CODE IN STEP XXX
TRAP CODE=AA
SHOULD=BB

2.1.5.11 E42 FAILED TO TRAP IN STEP XXX
TRAP SST=BB

V. Detailed Test Description

A. Section 1 (Extended Floating Point Instructions)

1. The following test steps comprise the following information:

```
STEP X : AAAA
     U
     V
    -V
     W
     CC
    OVERFLOW

Where:
    X = step number
 AAAA = Floating Point Instruction Type
    U = Operand -1
    V = Operand -2
   -V = Negate (operand -V)
    W = Operation result (Expected result)
   CC = Conditon Code (STA(6:2)) expected
        Where:
           CC = CCL (less than)
              = CCG (greater than)
              = CCE (equal to)
 OVERFLOW = Overflow Bit (STA(4:1))
          = ON or OFF (Expected)

STEP 1 : EADD
     U =%040001,%000002,%000003,%000004
     V =%140000,%002000,%003000,%004000
     W =%037176,%000375,%000574,%001000
    CC = CCG: OVERFLOW =OFF

STEP 2 : EADD
     U =%140000,%002000,%003000,%004000
     V =%040001,%000002,%000003,%000004
     W =%037176,%000375,%000574,%001000
    CC = CCG: OVERFLOW =OFF

STEP 3 : EADD
     U =%040000,%000002,%100000,%004000
     V =%140000,%000002,%000003,%000004
     W =%035077,%176407,%176000,%000000
    CC = CCG: OVERFLOW =OFF

STEP 4 : EADD
     U =%140000,%000002,%000003,%000004
     V =%040000,%000002,%100000,%004000
     W =%035077,%176407,%176000,%000000
    CC = CCG: OVERFLOW =OFF
```

V. Continued

```
STEP 5 : EADD
     U =%040000,%000002,%000003,%100000
     V =%140000,%000002,%000003,%000004
     W =%033077,%176000,%000000,%000000
    CC = CCG: OVERFLOW =OFF

STEP 6 : EADD
     U =%140000,%000002,%000003,%000004
     V =%040000,%000002,%000003;%100000
     W =%033077,%176000,%000000,%000000
    CC = CCG: OVERFLOW =OFF

STEP 7 : EADD
     U =%040001,%000002,%000003,%000004
     V =%000000,%000000,%000000,%000000
     W =%040001,%000002,%000003,%000004
    CC = CCG: OVERFLOW =OFF

STEP 8 : EADD
     U =%040301,%000002,%000003,%000004
     V =%040200,%100001,%100001,%100001
     W =%040341,%040002,%140003,%140005
    CC = CCG: OVERFLOW =OFF

STEP 9 : EADD
     U =%040301,%000002,%000003,%000004
     V =%040100,%100001,%100001,%100001
     W =%040321,%020002,%060003,%060004
    CC = CCG: OVERFLOW =OFF

STEP 10: EADD
     U =%040301,%000002,%000003,%000004
     V =%040000,%100001,%100001,%100001
     W =%040311,%010002,%030003,%030004
    CC = CCG: OVERFLOW =OFF

STEP 11: EADD
     U =%046701,%000002,%000003,%000004
     V =%040077,%002000,%003000,%004000
     W =%046701,%000002,%000003,%000005
    CC = CCG: OVERFLOW =OFF

STEP 12: EADD
     U =%047001,%000002,%000003,%000004
     V =%040077,%002000,%003000,%004000
     W =%047001,%000002,%000003,%000004
    CC = CCG: OVERFLOW =OFF
```

V. Continued

```
STEP 13: EADD
     U =%047101,%000002,%000003,%000004
     V =%040077,%002000,%003000,%004000
     W =%047101,%000002,%000003,%000004
    CC = CCG: OVERFLOW =OFF

STEP 14: EADD
     U =%040001,%000002,%000003,%000004
     V =%140001,%000002,%000003,%000004
     W =%000000,%000000,%000000,%000000
    CC = CCE: OVERFLOW =OFF

STEP 15: EADD
     U =%040077,%177777,%177777,%177777
     V =%040077,%177777,%177777,%177777
     W =%040177,%177777,%177777,%177777
    CC = CCG: OVERFLOW =OFF

STEP 16: EADD
     U =%040257,%177777,%177777,%177777
     V =%040000,%000000,%000000,%000003
     W =%040300,%000000,%000000,%000000
    CC = CCG: OVERFLOW =OFF

STEP 17: EADD
     U =%040077,%177777,%177777,%177777
     V =%140000,%000000,%000000,%000000
     W =%037777,%177777,%177777,%177776
    CC = CCG: OVERFLOW =OFF

STEP 18: EADD
     U =%040100,%000000,%000000,%000000
     V =%140077,%177777,%177777,%177777
     W =%031200,%000000,%000000,%000000
    CC = CCG: OVERFLOW =OFF

STEP 19: EADD
     U =%040200,%000000,%000000,%000000
     V =%140077,%177777,%177777,%177777
     W =%040100,%000000,%000000,%000001
    CC = CCG: OVERFLOW =OFF

STEP 20: EADD
     U =%077757,%177777,%177777,%177777
     V =%077500,%000000,%000000,%000001
     W =%077777,%177777,%177777,%177777
    CC = CCG: OVERFLOW =OFF
```

V. Continued

```
STEP 21: EADD
     U =%077757,%177777,%177777,%177777
     V =%077500,%000000,%000000,%000002
     W =%000000,%000000,%000000,%000000
    CC =CCG: OVERFLOW =ON

STEP 22: EADD
     U =%077777,%177777,%177777,%177777
     V =%077777,%177777,%177777,%177777
     W =%000077,%177777,%177777,%177777
    CC = CCG: OVERFLOW =ON

STEP 23: EADD
     U =%100100,%000000,%000000,%000001
     V =%100077,%177777,%177777,%177777
     W =%100000,%000000,%000000,%000000
    CC = CCL: OVERFLOW =ON

STEP 24: EADD
     U =%000100,%000000,%000000,%000000
     V =%100077,%177777,%177777,%177777
     W =%071200,%000000,%000000,%000000
    CC = CCG: OVERFLOW =OFF

STEP 25: EADD
     U =%000100,%000000,%000000,%000001
     V =%100000,%000000,%000000,%000001
     W =%000000,%000000,%000000,%-00001
    CC = CCG: OVERFLOW =OFF

STEP 26: ESUB
     U =%040000,%100000,%003000,%004000
     V =%040000,%000002,%000003,%000004
     W =%037077,%177005,%176407,%176000
    CC = CCG: OVERFLOW =OFF

STEP 27: ESUB
     U =%140000,%000002,%003000,%000004
     V =%140000,%000000,%003000,%004000
     W =%037077,%177005,%176407,%176000
    CC = CCG: OVERFLOW =OFF

STEP 28: ESUB
     U =%000001,%000002,%000003,%000004
     V =%100000,%100000,%000000,%000000
     W =%000100,%140001,%000001,%100002
    CC = CCG: OVERFLOW =OFF

STEP 29: ESUB
     U =%100001,%000002,%000003,%000004
     V =%000000,%000000,%100000,%000000
     W =%100100,%100001,%040001,%100002
    CC = CCL: OVERFLOW =OFF
```

V. Continued

```
STEP 30: ESUB
     U =%040001,%000002,%000003,%000004
     V =%040000,%177777,%177777,%100000
     W =%035300,%000160,%000200,%000000
    CC = CCG: OVERFLOW =OFF

STEP 31: ESUB
     U =%040001,%000002,%000003,%000004
     V =%040000,%002000,%177777,%100000
     W =%037176,%000200,%000700,%001000
    CC = CCG: OVERFLOW =OFF

STEP 32: ESUB
     U =%040001,%000002,%000003,%000004
     V =%040000,%177777,%003000,%100000
     W =%035337,%040120,%000200,%000000
    CC = CCG: OVERFLOW =OFF

STEP 33: EMPY
     U =%040001,%000002,%000003,%000004
     V =%000000,%000000,%000000,%000000
     W =%000000,%000000,%000000,%000000
    CC = CCE: OVERFLOW =OFF

STEP 34: EMPY
     U =%000000,%000000,%000000,%000000
     V =%040001,%000002,%000003,%000004
     W =%000000,%000000,%000000,%000000
    CC = CCE: OVERFLOW =OFF

STEP 35: EMPY
     U =%040077,%177777,%177777,%177777
     V =%040077,%177777,%177777,%177777
     W =%040177,%177777,%177777,%177776
    CC = CCG: OVERFLOW =OFF

STEP 36: EMPY
     U =%140001,%000002,%000003,%000004
     V =%140010,%002000,%003000,%004000
     W =%040011,%022022,%043073,%064205
    CC = CCG: OVERFLOW =OFF

STEP 37: EMPY
     U =%077777,%177777,%177777,%177777
     V =%177777,%177777,%177777,%177777
     W =%137777,%177777,%177777,%177776
    CC = CCL: OVERFLOW =ON
```

V. Continued

```
STEP 38: EMPY
     U =%100000,%000000,%000000,%000001
     V =%000000,%000000,%000000,%000001
     W =%140000,%000000,%000000,%000002
    CC = CCL: OVERFLOW =ON

STEP 39: ENEG
     V =%000001,%000002,%000003,%000004
    -V =%100001,%000002,%000003,%000004
    CC = CCL: OVERFLOW =OFF

STEP 40: ENEG
     V =%100000,%000002,%000003,%000004
    -V =%000000,%000002,%000003,%000004
    CC = CCG: OVERFLOW =OFF

STEP 41: ENEG
     V =%000000,%000002,%000003,%000004
    -V =%100000,%000002,%000003,%000004
    CC = CCL: OVERFLOW =OFF

STEP 42: ENEG
     V =%000000,%000000,%000003,%000004
    -V =%100000,%000000,%000003,%000004
    CC = CCL: OVERFLOW =OFF

STEP 43: ENEG
     V =%000000,%000000,%000000,%000004
    -V =%100000,%000000,%000000,%000004
    CC = CCL: OVERFLOW =OFF

STEP 44: ENEG
     V =%000000,%000000,%000000,%000000
    -V =%000000,%000000,%000000,%000000
    CC = CCE: OVERFLOW =OFF

STEP 45: ECMP
     U =%000000,%000002,%000003,%000004
     V =%140000,%002000,%003000,%004000
    CC = CCG: OVERFLOW =OFF

STEP 46: ECMP
     U =%040001,%000002,%000003,%000004
     V =%040000,%002000,%003000,%004000
    CC = CCG: OVERFLOW =OFF

STEP 47: ECMP
     U =%140001,%000002,%000003,%000004
     V =%140000,%002000,%003000,%004000
    CC = CCL: OVERFLOW =OFF
```

V. Continued

```
STEP 48: ECMP
     U =%000000,%000002,%000003,%000004
     V =%000000,%100000,%003000,%004000
    CC = CCL: OVERFLOW =OFF

STEP 49: ECMP
     U =%177777,%000002,%000003,%000004
     V =%177777,%100000,%003000,%004000
    CC = CCG: OVERFLOW =OFF

STEP 50: ECMP
     U =%000001,%000000,%000000,%000004
     V =%000001,%000000,%100000,%004000
    CC = CCL: OVERFLOW =OFF

STEP 51: ECMP
     U =%100000,%100000,%000000,%000004
     V =%100000,%100000,%100000,%004000
    CC = CCG: OVERFLOW =OFF

STEP 52: ECMP
     U =%000001,%000002,%000003,%000004
     V =%000001,%000002,%000003,%004000
    CC = CCL: OVERFLOW =OFF

STEP 53: ECMP
     U =%177777,%000002,%000003,%000004
     V =%177777,%000002,%000003,%004000
    CC = CCG: OVERFLOW =OFF

STEP 54: ECMP
     U =%000001,%000002,%000003,%100000
     V =%000001,%000002,%000003,%100000
    CC = CCE: OVERFLOW =OFF

STEP 55: EDIV
     U =%040000,%000002,%000003,%000004
     V =%000000,%000000,%000000,%000000
     W =%040000,%000002,%000003,%000004
    CC = CCG: OVERFLOW =ON

STEP 56: EDIV
     U =%000000,%000000,%000000,%000000
     V =%140000,%000002,%000003,%000004
     W =%000000,%000000,%000000,%000000
    CC = CCE: OVERFLOW =OFF
```

V. Continued

```
STEP 57: EDIV
     U =%040052,%17376,%062413,%127645
     V =%040000,%000000,%000000,%000000
     W =%040052,%173476,%062413,%127645
    CC = CCG: OVERFLOW =OFF

STEP 58: EDIV
     U =%141000,%000000,%000000,%000000
     V =%140077,%177600,%000000,%000000
     W =%040700,%000100,%000100,%000100
    CC = CCG: OVERFLOW =OFF

STEP 59 : EDIV
     U =%040077,%177777,%177777,%177777
     V =%137000,%000000,%000000,%000000
     W =%141077,%177777,%177777,%177777
    CC = CCL; OVERFLOW =OFF

STEP 60: EDIV
     U =%140077,%177777,%177777,%177777
     V =%040077,%177600,%000000,%000000
     W =%140000,%000100,%000100,%000100
    CC = CCL: OVERFLOW =OFF

STEP 61: EDIV
     U =%040077,%177600,%000000,%000000
     V =%040000,%000177,%177777,%177777
     W =%040077,%177200,%001377,%175002
    CC = CCG: OVERFLOW =OFF

STEP 62: EDIV
     U =%040000,%000000,%000000,%000000
     V =%040077,%177777,%177777,%177777
     W =%037700,%000000,%000000,%000001
    CC = CCG: OVERFLOW =OFF

STEP 63: EDIV
     U =%040000,%000377,%137400,%000000
     V =%040077,%177777,%177777,%177777
     W =%037700,%000377,%137400,%000001
    CC = CCG: OVERFLOW =OFF

STEP 64: EDIV
     U =%040000,%000000,000000,%077777
     V =%040000,%00000,%000000,%100000
     W =%037777,%177777,%17777,%177776
    CC = CCG: OVERFLOW =OFF

STEP 65: EDIV
     U =%040042,%016212,%127664,%122623
     V =%040020,%005564,%137141,%104405
     W =%040016,%070777,%177710,%107404
    CC = CCG: OVERFLOW =OFF
```

V. Continued

```
STEP 66: EDIV
     U =%040000,%000200,%000000,%000001
     V =%040000,%000000,%000000,%000001
     W =%040000,%000200,%000000,%000000
    CC = CCG: OVERFLOW =OFF

STEP 67: EDIV
     U =%040000,%000300,%000200,%000001
     V =%040000,%000100,%000000,%000001
     W =%040000,%000200,%000000,%000000
    CC = CCG: OVERFLOW =OFF

STEP 68: EDIV
     U =%040000,%000177,%177200,%000000
     V =%040000,%000177,%177777,%177600
     W =%037777,%177777,%176400,%003400
    CC = CCG: OVERFLOW =OFF

STEP 69: EDIV
     U =%040000,%000000,%000400,%000000
     V =%040000,%000000,%000200,%000000
     W =%040000,%000000,%000200,%000000
    CC = CCG: OVERFLOW =OFF

STEP 70: EDIV
     U =%040000,%000100,%000400,%000200
     V =%040000,%000100,%000200,%000000
     W =%040000,%000000,%000200,%000000
    CC = CCG: OVERFLOW =OFF

STEP 71: EDIV
     U =%040000,%000000,%000200,%000200
     V =%040000,%000000,%000000,%000200
     W =%040000,%000000,%000200,%000000
    CC = CCG: OVERFLOW =OFF

STEP 72: EDIV
     U =%040000,%000100,%000200,%000400
     V =%040000,%000100,%000000,%000200
     W =%040000,%000000,%000200,%000000
    CC = CCG: OVERFLOW =OFF

STEP 73: EDIV
     U =%040000,%000177,%177777,%177600
     V =%040000,%000177,%177600,%000000
     W =%040000,%000000,%000177,%177200
    CC = CCG: OVERFLOW =OFF
```

V. Continued

STEP 74: EDIV
U =%040000,%000000,%000200,%000200
V =%040000,%000000,%000200,%000000
W =%040000,%000000,%000000,%000200
CC = CCG: OVERFLOW =OFF

STEP 75: EDIV
U =%000000,%000000,%000000,%000001
V =%077777,%177777,%177777,%177777
W =%040000,%000000,%000000,%000002
CC = CCG: OVERFLOW =ON

STEP 76: EDIV
U =%077777,%177777,%177777,%177777
V =%000000,%000000,%000000,%000001
W =%037777,%177777,%177777,%177775
CC = CCG: OVERFLOW =ON

2. Special Test Cases

STEP 77: Tests 'ECMP' instruction with 4 word U-operand and its source address on TOS. Uses same case data as Step 54.

STEP 78: Tests 'EDIV' instruction with SR=3 uses same case data as Step 74.

STEP 79: Tests 'EMPY' instruction with SR=3 uses same case data as step 33

STEP 80: Tests 'EADD' instruction with SR=3 uses same case data as step 19.

STEP 81: Tests 'ESUB' instruction with SR=3 uses same case data as Step 29.

STEP 82: Tests 'ENEG' instruction with 4-word V-operand and its source address on TOS.
Uses same case data as Step 44.

STEP 83: Tests Stack Overflow with 'EADD' instruction with Z<S. It should trap to STT #24 and leave the stack unchanged during the operation.
TOS is loaded with pre-defined pattern for verification.

STEP 84: Tests W>(S-4) with 'EADD' instruction for a upper bounds test. It should trap to STT #1 and leave the stack unchanged during the operation.
Uses same case data as Step 19.

V. Continued

STEP 85: Tests U>(S-4) with 'EADD' instruction for a upper bounds test. It should trap to STT #1 without any error and leave the stack unchanged during the operation. Uses same case data as Step 1.

STEP 86: Tests V>(S-4) with 'EADD' instruction for a upper bound test. It should trap to STT #1 without any error and leave the stack unchanged during the operation. Uses same case data as Step 2.

STEP 87: Tests W<DL with 'EADD' instruction for a lower bound test. It should trap to STT #1 without any error and leave the stack unchanged during the operation. Uses same case data as Step 4.

STEP 88: Tests U<DL with 'EADD' instruction for a lower bound test. It should trap to STT #1 without any error and leave the stack unchanged during the operation. Uses same case data as Step 5.

STEP 89: Tests V<DL with 'EADD' instruction for a lower bound test. It should trap to STT #1 without any error and leave the stack unchanged during the operation. Uses same case data as Step 6.

STEP 90: Tests V>(S-4) with 'EDIV' instruction for a upper bound test. It should trap to STT #1 without any error and leave the stack unchanged during the operation. Uses same case data as Step 56.

STEP 91: Tests U>(S-4) with 'ECMP' instruction for a upper bound test. It should trap to STT #1 without any error and pop 2 words off the stack during the operation. Uses same case data as Step 54.

STEP 92: Tests V>(S-4) with 'ECMP' instruction for a upper bound test. It should trap to STT #1 without any error and leave the stack unchanged during the operation. Uses same case data as step 53.

V. Continued

STEP 93: Tests U<DL with 'ECMP' instruction for a lower bound test. If should trap to STT #1 without any error and pop 2 words off the stack during the operation. Uses same case data as Step 52.

STEP 94: Tests V<DL with 'ECMP' instruction for a lower bound test. It should trap to STT #1 without any error and leave the stack unchanged during the operation. Uses same case data as Step 51.

STEP 95: Tests overflow bit (ON) (STA(4)=1) with user trap disable (STA(2)=Ø) for an expected underflow trap without trapping. It should not trap to STT #25 (user trap) with trap code equal %11 but expect STA(4)=1(overflow bit on). Uses same case data as Step 23.