NORD-10
Verification Programs

NORD-10
Verification Programs

REVISION RECORD

Revision	Notes
8/74	Original Printing

Publ. No.
ND-62.006.01
August 1974

NORSK DATA A.S.

Lorenveien 57, Postboks 163 Økern, Oslo 5, Norway

TABLE OF CONTENTS

+++

Chapters:		Page:
1	INTRODUCTION	1-1
1.1	Use of the Verification Programs	1-1
1.2	Error Messages	1-2
2	ONE-CHECK	2-1
2.1	Introduction	2-1
2.2	Usage	2-1
2.2.1	Load the Program	2-1
2.2.2	Stop 1	2-1
2.2.3	Stop 2	2-1
2.2.4	Stop 3	2-2
2.2.5	Continous Operation	2-2
2.2.6	Procedure to follow when the Program stops	2-2
2.3	Error Stops	2-2
2.3.1	Error List	2-3
3	TWO-CHECK	3-1
3.1	Introduction	3-1
3.2	Usage	3-1
3.2.1	Load the Program	3-1
3.3	Error Loops	3-1
3.3.1	Error List	3-2
4	THREE-CHECK	4-1
4.1	Introduction	4-1
4.2	Usage	4-1
4.2.1	Load the Program	4-1
4.3	Error Stops	4-2
4.3.1	Error List	4-2
5	FOUR-CHECK	5-1
5.1	Introduction	5-1
5.2	Usage	5-1
5.2.1	Load the Program	5-1
5.2.2	Interrupts to test	5-1
5.2.3	Printout Mode	5-2
5.2.4	Program Control	5-2
5.3	Error Printouts	5-2

Chapters:

		Page:
6	10-FLOATING	6-1
6.1	Introduction	6-1
6.2	Usage	6-1
6.2.1	Load the Program	6-1
6.3	Error Printout	6-1
6.4	Error Loop	6-1
7	NORD-10 MICRO-PROGRAMMED MEMORY TEST	7-1
7.1	Introduction	7-1
7.2	Usage	7-1
7.2.1	Setting of ALD	7-1
7.2.2	Setting of Memory Bounds	7-1
7.2.3	Starting the Memory Test	7-2
7.3	Error Indication	7-2
7.4	Method	
7.4.1	Test Patterns used	7-2
7.5	Further Test	7-3
7.5.1	Loop Description	7-3
7.5.2	Starting the Loop	7-3

Appendices: (To be obtained from ND on request)

Appendix A ONE-CHECK Program Listing

Appendix B TWO-CHECK Program Listing

Appendix C THREE-CHECK Program Listing

Appendix D FOUR-CHECK Program Listing

Appendix E 10-FLOATING Program Listing

Appendix F ALD Bit Map

1 INTRODUCTION

This manual describes available verification programs. Reliable computer self-diagnosis is usually very difficult to obtain. The purpose of these programs is twofold. First, they can be used to explore many CPU functions in a relatively short period of time. Second, they may, in many cases, be used to narrow down the area of search when the CPU is actually found to be failing.

The diagnosis suggested in the following will only apply if the CPU fails the way the program-designer was able to forecast. However, the program-listings are included for reference so that possible secondary error symptoms may be more easily analyzed.

1.1 Use of the Verification Programs

The programs will normally be made available as standard binary format tapes. How to read in a binary tape is explained in the NORD-10 Reference Manual. However, a short recollection is made here:

Make sure that bit 15 of the ALD register is not set. ALD is the Automatic Load Descriptor switch register found on the CPU Panel Driver Card (card position 17)*. If a console Teletype is available, then the setting of ALD may be checked by inspecting the internal register 12_8.

Put the binary tape in the appropriate device. Push Master Clear.

Type:

<device address$_8$> &

on the console Teletype and the tape is read in. The program will be started automatically if successfully loaded. If a load-error occurs, then either the light in the Master Clear button will turn on, or '?' will be typed on the console.

The device address is the lowest address associated with the load device:

Tape reader : Device address 400_8
Console Teletype: Device address 300_8

On CPU's without console Teletype, the procedure is as follows:

Make sure that bits 12-15 of ALD are 0. Put the device address in ALD bits 0 - 10. Push Master Clear and LOAD.

If one experiences trouble in loading the programs, then there might be errors that effect either the load device or the CPU so that the operators communication does not work properly. In this case it could be worthwhile to interchange the two arithmetic cards (hardware personell only!).

---

* Refer to Appendix A for arrangement drawing.

1.2 Error Messages

Some of the verification programs rely on error stops to announce errors, while others give error printouts on the console Teletype.

If the CPU stops (STOP button is lighted), one should examine the IR register bits 0 - 7 to see the number of the actual error. One may then always press the CONTINUE button to resume program operation, or the RESTART button to restart the program.

2 ONE-CHECK

2.1 Introduction

This program checks all the NORD-1 instructions in NORD-10 except the IOT, MIS and floating FAD, FSB, FMU, FDV instructions. The instructions are tested in sequence such that an instruction is tested by means of the instructions already tested. To initiate this process the instructions "WAIT", "LDA" and "STZ" are tested separately at the beginning of the program. The program uses the content of two cells NK1 = 125252 and NK2 = 052525. These cells are referred to in the error list.

2.2 Usage

2.2.1 Load the Program

Read in the program as already described. The program will automatically start and then stop immediately in a WAIT 0 instruction. Check that:

IR register = 151000
P register = 001701 R2/

If ok, continue as described in Section 2.2.2.

If not ok, start the program in address 1700_8. The WAIT instruction does not work if the IR and P registers still are wrong.

2.2.2 Stop 1

The content of IR register should be 151001.
The content of A register should be 125252 = NK1. R5/

1) If IR ≠ 151001, then the WAIT instruction does not work.

2) If A ≠ 125252, the LDA does not work.

Now press CONTINUE.

2.2.3 Stop 2

The content of IR register should be 151002.
The content of A register should be 052525. R5/

1) If IR ≠ 151002, then the WAIT instruction does not work.

2) If A / 052525, then LDA or STA does not work.

Now press CONTINUE.

2.2.4 Stop 3

The content of IR register should be 151003.
The content of A register should be 0.

1) If IR ≠ 151003, then the WAIT instruction does not work.

2) If A ≠ 0, then STZ or LDA does not work.

Now press CONTING.

2.2.5 Continous Operation

The program will now run until the operator presses STOP or until an error is detected.

Note: If the program is restarted by means of the RESTART button, it will start at this point.

2.2.6 Procedure to follow when the Program stops

The octal value of the contents of bits 0 - 7 of the IR register gives the entrance to the error list. The error list contains some abbreviations which will be explained beneath.

1) All the instructions are represented by their MAC mnemonic.

2) DN means destination register A, T, D, X or B.

3) BN written in a BOP instruction means bit no. BN where BN is an octal number between 0 and 17. .

4) Further information about the error can be gained by examination of the symbolic version of the program. By means of this procedure it is possible to find for instance which bit or which register that failed in 2) and 3).

2.3 Error Stops

As mentioned this program tests one instruction at a time and uses only tested instructions in the tests. However, one instruction is tested in only a few addresses, operating on only a few different data patterns. When one instruction is tested, it is still possible that this instruction may fail for a certain combination of address and data patterns. Consequently the error list will tell the user only the most probable cause of failure.

1.3.1 Error List

Value of bits 0-7 of IR register	Probable cause of the stop
4	JAZ does not jump on A=0
5	JAF jumped on A=0
6	JAZ jumped on A≠0
7	MIN skipped on NK3≠0, A=NK3
10	MIN does not skip on NK3=0
11	SUB does not work
12	COPY CM1 SA DA failed
13	COPY CM2 SA DA failed
14	ADD failed
15	STT or LDT failed on NK1
16	STT or LDT failed on NK2
17	LDX or STX failed on NK1
20	LDX or STX failed on NK2
21	AND failed on NK2 - NK2 · NK2
22	AND failed on 0=NK1 · NK2
23	ORA failed on ORA(NK2, NK2)=NK2
24	ORA failed on ORA(NK1, NK2) + 1 = 0
25	JAP does not jump on A positive
26	JAF does not jump on A positive and ≠ 0
27	JAZ jumped on A≠0
30	JAN jumped on A positive
31	JXN jumped on X positive
32	JXZ jumped on X positive and X≠0
33	JPC does not jump on X positive
34	JNC jumped on X positive
35	JPC or JNC are not counting correctly
36	JAP jumped on A negative
37	JAZ jumped on A negative
40	JAN does not jump on A negative
41	JAF does not jump on A negative
42	JXZ jumped on X negative
43	JXN does not jump on X negative
44	JXZ does not jump on X=0

45	LDD, A is not loaded correctly
46	STD, A is not stored correctly
47	STD, LDD, D-reg. is either stored or loaded incorrectly during STD or LDD
50	LDF, A-reg. is not loaded correctly
51	LDF, STF, T-reg. is either stored or loaded incorrectly
53	STF, A-reg. is stored incorrectly during STF
53	LDF, STF, D is either stored or loaded incorrectly during LDF or STF
54 - 57	Not used
60	COPY DN or SKP IF DN EQL 0 failed
61	COPY SA DN or SKP IF DN GRE SA failed
62	SKP IF DN EQL SA failed
63	SKP IF DN LST SA failed
64	SKP IF DN UEQ SA failed
65	RADD SA DN, RSUB SA DN or SWAP SA DN failed
66	RINC DN or RDCR DN failed
67	RINC DN, RDCR DN or SWAP SA DN failed
70	COPY CM1 SX DN or RAND SX DN failed
71	COPY ST DN or RAND ST DN failed
72	SWAP SA DN failed
73	REXO SA DN failed
74	REXO DN failed
75	RORA DN failed on (N)=0
76	RORA DN failed on (N)≠0
77	Not used
100	JPL, P was not transferred to L
101	JPL does not jump
102	JPL, P was not correctly transferred to L
103	SAA-1 does not work
104	SAX-1 " " "
105	SAT-1 " " "
106	SAB-1 " " "
107	SAA 1 " " "

110	SAX 1 " " "
111	SAT 1 " " "
112	SAB 1 " " "
113	AAA " " "
114	AAX " " "
115	AAT " " "
116	AAB " " "
117	Not used

Testing of bit operations:

120	BSET ZRO SSK or BSKP ZRO SSK failed
121	BSET ONE SSK or BSKP ONE SSK failed
122	BSET BCM SSK failed
123	BSET BAC SSK failed
124	BSKP BCM SSK failed
125	BSKP BAC SSK failed
126	Not used
127	Not used
130	BSET ONE SSK or TRA STS failed
131	BSET ONE SSZ or TRA STS failed
132	BSET ONE SSQ or TRA STS failed
133	BSET ONE SSO or TRA STS failed
134	BSET ONE SSC or TRA STS failed
135	BSET ONE SSM or TRA STS failed
136	BSET ZRO SSK, SSZ, SSQ, SSO, SSC or SSM failed. The content of A shows which of them that failed, (indicated with a "one" in the corresponding bit in A).
137	BSET BCM SSK, SSZ, SSQ, SSO, SSC or SSM failed. Add 374_8 to A, and zeroes in bits 2-7 indicate which of the instructions that failed.
140	BSET BAC SSK, SSZ, SSQ, SSO, SSC or SSM failed. The "ones" in A-reg. bits 2-7, indicate which of the instructions that failed.
141	BSET ZRO DD BN failed
142	BSKP ZRO DD BN failed
143	BSKP ONE DD BN failed
144	BSET ZRO DD BN failed

145	BSET ONE DL BN failed
146	BSKP ONE DL BN failed
147	Not used
150	BSET ONE 110 DB or BSKP ONE 110 DB failed.
151	BSKP ZRO 110 DB failed
152	BSET BCM 110 DB failed on actual bit=0
153	BSET BCM 110 DB failed on actual bit=1
154	BSET BAC 130 DT or BSKP ZRO 130 DT failed.
155	BSKP BAC 130 DT failed
156	BSKP BCM 130 DT failed
157	BSET BAC 170 DL or BSKP ONE 170 DL failed.
160	BSKP BAC 170 DL failed
161	BSTA 60 DT or BSKP ONE 60 DT failed
162	BSTA 60 DT does not store zero in SSK
163	BSTC 70 DX or BSKP ONE 70 DX failed
164	BSTC 70 DX does not store "one" in SSK
165	BLDC 70 DX failed
166	BLDA 60 DT failed
167	BAND 60 DT failed
170	BANC 60 DT failed
171	BORA 50 DB or BSET ZRO 50 DB failed
172	BORA 60 DT failed
173	BAND 50 DB failed
174	BORC 50 DB failed
175 - 177	Not used
200	,X does not work
201	,B does not work
202	,X ,B does not work
203	I does not work (indirect addressing)
204	I ,X does not work
205	I ,B does not work
206	I ,B ,X does not work
207	Wrong result of MPY multiplicand in D-reg. multiplicator (memory) in A-reg. wrong result in B-reg. correct result in L-reg.

210	Overflow for MPY does not work. The correct value of the O flip-flop stands in the O bit in the T register
211 - 220	Not used

Test of Shift Instruction:

220	SHA failed
221	SAD failed
222	SAD SHR failed
223	SHT ROT failed
224	SHD ZIN failed
225	SHA 17 failed
226	SAD SHR 40 failed. Content of D-reg. is wrong.
227	SAD SHR 40 failed. Content of A-reg. is wrong.
230	SAD ROT failed
231	SAD ROT failed
232	The M flip-flop was not set to "one" during left shift.
233	SAD LIN SHR 1 does not work
234	The M flip-flop was not cleared during left shift.
235	The M flip-flop was not set to "one" during left shift
236	The M flip-flop was not cleared during right shift.
237	SAD ROT SHR does not work
240	C flip-flop was incorrectly set to "one" during ADD.
241	O flip-flop was incorrectly set to "one" during ADD.
242	Q flip-flop was incorrectly set to "one" during ADD.
243	C flip-flop was not set during SUB
244	O flip-flop was incorrectly set to "one" during SUB.
245	Q flip-flop was incorrectly set to "one" during SUB.
246	C flip-flop was incorrectly set to "one" during ADD.

247	O flip-flop was incorrectly cleared during ADD.
250	Q flip-flop was incorrectly cleared during ADD.
251	C flip-flop was incorrectly cleared during AAA.
252	Q flip-flop was incorrectly set to "one" during AAA.
253	O flip-flop was incorrectly cleared during AAA.
254	C flip-flop was incorrectly cleared during SUB.
255	O flip-flop was incorrectly cleared during SUB.
256	Q flip-flop was incorrectly cleared during SUB

3 TWO-CHECK

3.1 Introduction

TWO-CHECK verifies extended NORD-10 instructions (as related to NORD-1). The following is checked:

SKP	- all 8 skip conditions
MIX3	- multiply index by 3
RMPY	- register multiply
RDIV	- register divide
EXR	- execute register
IRR/IRW	- interregister read and write
LRB/SRB	- load and store register block
LBYT/SBYT	- load and store byte

Errors are reported through error stops (WAIT instructions). Refer to Section 3.3.

3.2 Usage

3.2.1 Load the Program

Read in the program as explained in the introduction to this manual. The program should start automatically. If not, start in address 20.

Normal stop is WAIT 377. If this instruction is removed, the program will loop until stopped by an error or the operator.

3.3 Error Loops

As already noted, the indicated cause of error is only the most probable one.

3.3.1 Error List

Value of IR Bits 0-7	Probable Cause of Strap
0	SKP IF DA EQL SD soes not skip.
1	SKP IF DA GFQ* SD does not skip.
2	SKP IF DA GRF SD docs not skip.
3	SKP IF DA MGRE SD does not skip.
4	SKP IF DA UEQ SD does not skip.
5	SKP IF DA LSS* SD does not skip.
6	SKP IF DA LST SD does not skip.
7	SKP IF DA MLST SD does not skip.
10	SKP IF DA EQL SD skips.
11	SKP IF DA GEQ* SD skips.
12	SKP IF DA GRE SD skips.
13	SKP IF DA MGRE SD skips.
14	SKP IF DA UFQ SD skips.
15	SKP IF DA LSS* SD skips.
16	SKP IF DA LST SD skips.
17	SKP IF DA MLST SD skips.
21	MIX3, wrong result. T register shows correct result.
22	RMPY SD DA fails - AD actual result - BL correct result - T 1. factor (A register) - X 2. factor (D register)

---

* GEQ NORD-1 equivalent GRE (wrong operation if expression overflows)

LSS NORD-1 equivalent LST (wrong operation if expression overflows)

23 RDIV ST, wrong result

- AD actual result
- BL correct result
- T divisor
- X memory address of dividend (double word)

24 RDIV ST, wrong error indicator setting

- registers as for WAIT 23
- SSZ actual error indicator setting
- SSK correct error indicator setting

25 EXR SX fails

- X contains JMP * + 2
- EXR does not jump

26 EXR ST fails

- T contains LDA ,X
- A actual result
- X correct result

27 EXR SA fails
A contained a STZ ,X
- A actual, should be 0

30 EXR, incorrect error indicator setting.
One or more of the above has probably set the error indicator.

31 EXR SA, incorrect error indicator setting.

- A contains EXR SX
- X contains EXR ST
- T contains COPY 00 DD

The Z-indicator is not set when trying to execute an EXR.

32 EXR SA, refer to error stop 31.
The execute of an execute is performed.

33 TWO-CHECK is not running on level 0.
Push Master Clear and restart the program.

34 IRR or IRW failed

- A wrong result
- failing register in D0-2 (register number)
- failing level in D3-6 (program level)

The register code
(Register number + 10_8 * level + bit 7)
is written into all registers except on level 0.
The same registers are then read back and checked.

35 LRB does not work.

- A wrong result
- failing register in D0-2 (register number)
- failing level in D3-6 (program level)

The register blocks on all levels (except 0) is loaded with the register code (register number + 10_8 * level)
The registers are then checked with the IRR instruction.

Note: The actual instruction is EXR SA with A = LRB <program level * 10>

36 SRB does not work.
Error indication as for error stop 35.

Note: The actual instruction is EXR SA with A = SRB <program level * 10>

37 LBYT fails

- correct byte in B0-7
- wrong byte in A
- address in X and T

40 SBYT fails
Error information as for error stop 37

377 Correct stop.
Press CONTINUE to repeat test.
This instruction can be removed if continuous testing is wanted.

4 THREE-CHECK

4.1 Introduction

THREE-CHECK verifies programmed interrupts. The PIE and PID registers are set bit by bit. This moves the CPU from program level to program level with increasing priority. When level 15 is reached, the programs start executing WAIT's. This will move the program towards level 0.

On each level it is checked that present and previous level is as expected together with the settings of PIE and PID registers.

The program runs about .5 second on each level.

Errors are reported through error stops. Refer to Section 4.3.

4.2 Usage

4.2.1 Load the Program

Read in the program as explained in the introduction to this manual. Normally one will display active levels on the operators panel. This will show how the program moves from level to level.

4.3 Error Stops

When an error occurs, the interrupt system is turned off and the CPU halts. Normally the failing level can be seen on the operators panel or in location XREG. The address of XREG is found on the tape.

Error stops 0-5 occur with the interrupt system turned off.

Note that register names always refer to the failing level.

4.3.1 Error List

Value of IR0-7	Probable cause
0	The test program is not started on level 0. Push Master Clear before starting the program.
1	TRR IIE changes the A register. A should equal 0.
2	TRR PIE changes the A register. A should equal 0.
3	TRR PID changes the A register. A should equal 0.
4	PIE is not set to 0.
5	PID is not set to 0, possibly an external condition (jammed interrupt line).
20	Wrong level read from STS when moving up. - A offending level number. - X correct level.
21	Wrong previous level code moving up. - A offending code. - T correct code.
22	Wrong PIE when moving up. - A offending PIE. - D correct PIE.
23	Wrong PID when moving up. - A offending PID. - D correct PID.
24	Wrong level read from STS when moving down. - A offending level number. - X correct level

25	Wrong previous level code when moving down. - A offending code. - T correct code.
26	Wrong PIE when moving down. - A offending PIE. - L correct PIE.
27	WAIT is ignored on a level different from level 0. - A offending level.
30	Correct stop. The test may be run again by pushing CONTINUE. If this WAIT instruction is removed, then the test is repeated continuously.
31	Wrong level on way up. Wrong level no. in X, correct in A on current level.
32	Wrong level on way down. Wrong level no. in X, correct in A on current level
33	Wrong PIM (internal register 9 or 011B). Read PIM in A, expected in B.
40	PID set. Expected to be zero
41	MON do not set PID14
42	Difficult to reset PID14 after MON-instruction
377	This is the level-changing WAIT and it should never stop the CPU.

5 FOUR-CHECK

5.1 Introduction

FOUR-CHECK is an internal interrupt verification program. All possible internal interrupts in the CPU are triggered from program level 0 and it is checked that they report to level 14 in the specified way.

All internal interrupts will normally be enabled. The following instructions are used to trigger the interrupts.

IIC Code		
1	MON 123	% MONITOR CALL
4	143700	% UNIMPLEMENTED INSTRUCTION
5	BSET ONE SSZ	% ERROR INDICATOR
5	DNZ -20	% DNZ OVERFLOW (ERROR INDICATOR)
5	FDV NULL-A,B	% FDV BY 0.0 (ERROR INDICATOR)
5	EXR SX	% EXECUTE EXECUTE (ERROR INDICATOR)
7	IDX 3777	% IDX ERROR
9	LDA I (177377	% MEMORY-OUT-OF-RANGE

If memory parity is detected, this will be reported together with the contents of registers PES and PEA.

All other interrupts will be reported as errors.

5.2 Usage

5.2.1 Load the Program

Read in the program as explained in the introduction to this manual. Note that this program uses the console Teletype for printouts.

5.2.2 Interrupts to test

Only the enabled interrupts will be tested. IIFM is an internal interrupt mask that is transferred to the IIE register during program initialization.

The address of IIEM is noted on the tape and may be changed according to need.

5.2.3 Printout Mode

If OPR bit 15 is 1, only a minimum of text is printed.

If OPR bit 15 is 0, a full report will be printed.

5.2.4 Program Control

The program may be controlled by typing commands on the console Teletype. The following commands are available:

S - stop the program

G - continue the program (after S)

I - initialize the program (after S)

5.3 Error Printouts

Errors will only be reported when there is a change in error status. That is, if a working interrupt starts failing or a failing interrupt starts working. All interrupts are initially believed to be working.

All printouts are self-explanatory.

If OPR bit 14 is 1, the program will automatically simulate an 'S'-command after each error printout.

6 10-FLOATING

6.1 Introduction

10-FLOATING verifies these instructions:

DNZ, NLZ, FMU, FDV, FAD, FSB

The program contains a table of test data, and compares the results of instruction execution with the correct answers.

If an error is detected, an error message is printed.

6.2 Usage

6.2.1 Load the Program

Read in the program as explained in the introduction to this manual. The program should start automatically. If not, start it in address 20.

If no error occurs, the program loops indefinitely.

6.3 Error Printout

<N> <n>

A: <floating accumulator>
H: <floating memory data>
C: <correct result>
R: <actual result>

N indicates the failing instruction. It may be DZ, NZ, MU, DV, AD, and SB. n is data set number.

Pointers to the different data sets may be found in addr 0470 < 0475. Each data set occupies 9 locations.

6.4 Error Loop

After an error message, the program stops with WAIT 246.

a) Press CONTINUE. The program continues with new data.

b) Enter 0246 into the P register and press CONTINUE. The program loops and repeats the failing data set without error printouts. The result is copied to L, X, and B.

7 NORD-10 MICRO-PROGRAMMED MEMORY TEST

7.1 Introduction

The NORD-10 micro-program contains a special program that tests the main memory. This is a very useful feature as it may quickly be decided wether a given error is to be blamed on the CPU or the memory.

7.2 Usage

The use of this program is dependant on the ALD register. This is a 16-bit switch register located on the Panel Driver (or its alternative) in CPU card position 17. The setting of this register may be determined by inspecting internal register 12_8. (Type I12/ on the console Teletype.)

7.2.1 Setting of ALD

Note the initial setting of ALD (for later resetting).

Set up

101662	(version 1.0)
101657	(version 2.0)

in ALD (refer to Appendix F for arrangement drawing).

WARNING: This number refers to version 1.0 and 2.0 of the micro-program. Make sure what is the actual version on your machine.

This setting has the following significance:

Bit 15 = 1	Take bit 0 - 11 as micro-program address.
Bit 0 - 11	Start address of test program in the ROM memory.

Setting of ALD is only necessary for version 1.0.

7.2.2 Setting of Memory Bounds

The area of memory that is to be tested must be set up in the following registers (on level 0):

B	- lower memory bound
X	- upper memory bound

The tests include the specified bounds.

7.2.3 Starting the Memory Test

Push Master Clear and Load. The light in the Load-button will be on during the test.

Note: The normal operators communication will not be available when the test runs. This means that the display of register or memory contents is impossible (on the operators panel).

Position DATA or ADR, however, still shows the actual memory accesses.

For version 2.0 of the micro-program, writing 101657$ on the TTY is sufficient to start the test.

7.3 Error Indication

If the memory test fails, a '?' will be typed on the console TTY and/or the light in Master Clear will turn on. The following registers (on level 0) describe the error:

B	-	lower memory
X	-	upper memory bound
T	-	failing bits (=1) T D 'xor' L)
D	-	error pattern
L	-	test pattern
P	-	failing address

7.4 Method

The test writes the actual test pattern into all specified memory locations. The pattern is then read back. If no error occurred, the test goes on to the next test pattern.

7.4.1 Test Patterns used

The following test patterns are used:

000001
000002
000004
:
100000
177776
177775
177773
:
077777

Address stored in address (run 16 times)
Complement of address stored in address (run 16 times)

7.5 Further Test

Also located in the micro-program is a single location write/read loop. This is useful when tracing the data paths with an oscilloscope.

7.5.1 Loop Description

This program performs the following loop:

- Store L register in address pointed to by B register.
- Load D register with content of address pointed to by B register.
- Form 'exclusive or' of D and L registers and put result in T register.
- Loop back.

7.5.2 Starting the Loop

Load L and B registers with desired test pattern and address

Push Master Clear
Set ALD to 101771 and push Load (version 1.0)
Write 101767 $ on TTY (version 2.0).

The loop can only be broken by pushing Master Clear.

Note that it is not possible to examine the central register when this loop runs.