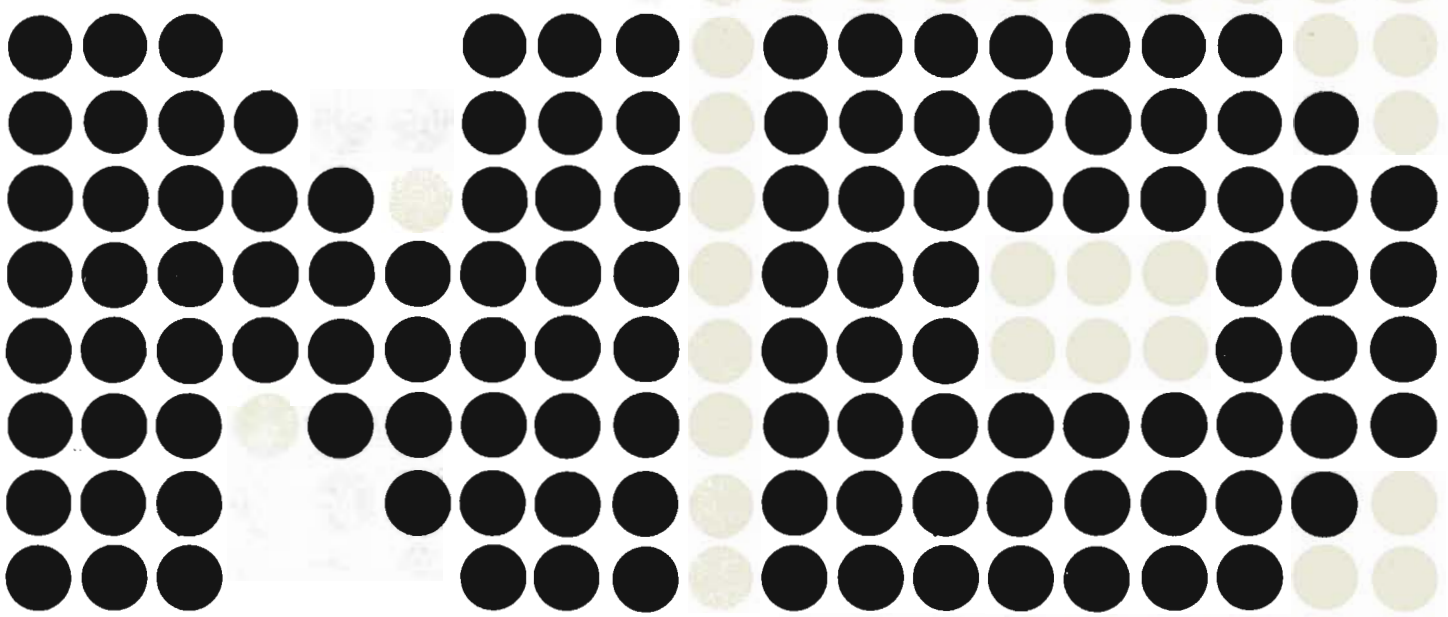
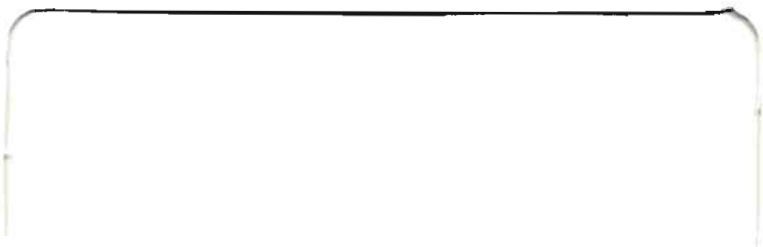
NOTEDS REFERENCE MANUAL

NORD-10

Test and Debugging System

A/S NORSK DATA-ELEKTRONIKK





NOTEDS REFERENCE MANUAL

NORD-10

Test and Debugging System

REVISION RECORD

Revision	Notes
1/74	Original Printing

ND-62.004.01
January 1974

A/S NORSK DATA-ELEKTRONIKK
Lørenveien 57, Oslo 5 - Tlf.: 21 73 71

PURPOSE

+++

This manual describes the NORD-10 Test and Debugging System (NOTEDS). NOTEDS is a computer-based tool used in NORD-10 production and maintenance.

Supplementary information will be found in the NORD-10 Microprocessor Manual and the NORD-10 Reference Manual.

--ooOoo--

TABLE OF CONTENTS

Chapters

Appendices:

1 INTRODUCTION

The NORD-10 Test and Debugging System (NOTEDS) is a novel, computer-based tool for use in NORD-10 (N-10) production and maintenance. The system provides extremely accurate and powerful means for pinpointing errors in the N-10 CPU. NOTEDS rests on the idea of having one computer test another. This makes it possible to perform very extensive and systematic tests in a short period of time and still maintain an absolute accuracy in the verification of test results.

When NOTEDS discovers an error in the CPU under test (CUT), it provides all relevant data for the test in question and locks in a fixed error loop. In this loop the test conditions are kept constant and a special program-triggered signal is given once per loop. This greatly facilitates good oscilloscope triggering. A lamp is flicked on or off each time an error is detected and thus provides a visual indication as to wether the error is permanent or intermittent.

Effective use of NOTEDS is only possible if the operator has at least some knowledge of N-10 hardware. NOTEDS does not provide any information as to the exact cause of the error, it only presents the error symptoms. However, the different tests are simple and easy to comprehend. In many cases it will be possible to locate the faulty card by examination of the error printouts only. The standard test sequence is incremental in design so that only limited new hardware is tested in each test.

The design and implementation of NOTEDS were carried out in parallel with the NORD-10 prototype development. Already at this early stage NOTEDS proved its usefullness and great flexibility.

2 HOW TO USE THIS MANUAL

This manual is the reference manual for the NORD-10 Test and Debugging System, NOTEDS. The information contained herein will be of interest to

1) NORD-10 Production Engineers.

2) NORD-10 Maintenance Engineers in multi-computer installations. Read Chapter 3 and 4.

3) Persons making extensions to the NORD-10 micro-program. Read chapter 5.

4) Persons maintaining NOTEDS hardware and software.

Truely advanced use of NOTEDS will require thorough knowledge of the NORD-10 CPU, the NORD-10 micro-processor specification and this manual.

3 HARDWARE

3.1 Introduction

NOTEDS hardware consists of four special cards.

The Test Driver plugs into the I/O-bus of the TSC* and accepts 8 device addresses. These addresses are fixed to $20 - 27_8$.

Three special cards are needed in the CUT**

The Test Connector plugs into a spare memory buffer position and is connected to the Test Driver via a cable. (Slot 18.)

The Test System Card plugs into the position normally occupied by the Panel Driver. (Slot 17.)

The ROM Simulator plugs into the ROM position. (Slot 10.)

The cable normally connecting the Panel Driver position and the panel itself must be disconnected when NOTEDS is in use.

A special test bus (TW) is established between these three cards when they are plugged in. This bus is 16 bit wide with 4 additional control lines. (TWCS, TWC0-3.)

Figure 3.1 shows the special NOTEDS hardware. The interaction points with the main CUT CPU is not shown.

---

* TSC - Test system computer

** CUT - Computer under test

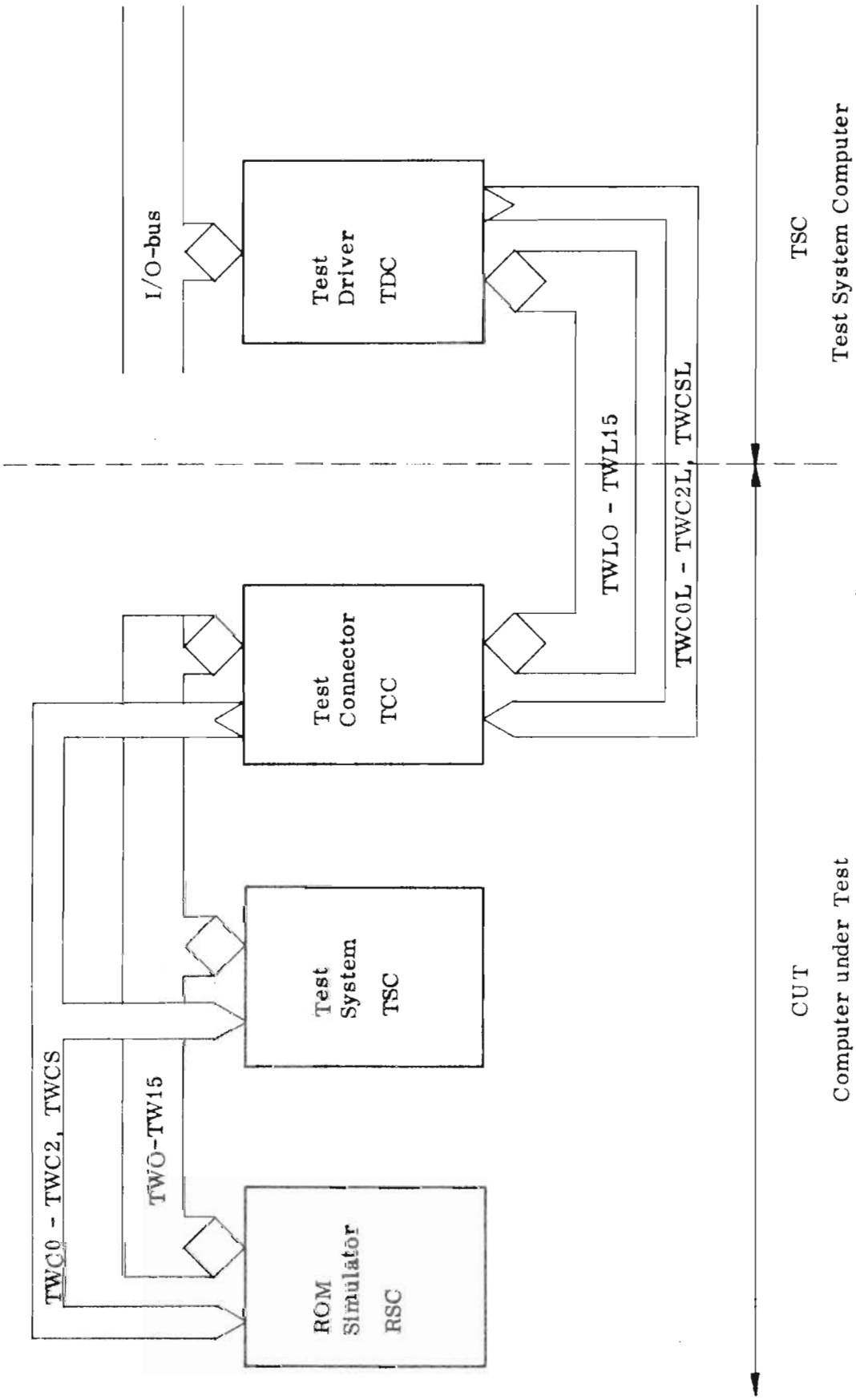


Figure 3.1
NOTEDS Hardware

3.2 Test Driver Card

The test driver card (TDC) resides in the TSC where it is connected to the I/O-bus. Its purpose is to interface the CUT to the TSC via a two-way bus (TWL).

The TDC accepts 8 device addresses:

3.2.1 Load most significant MIR

IOX HMIRL % 164023

The TSC A register is transferred to a buffer register on the RSC (refer to Section 3.5) to simulate bits 16 - 31 in the ROM.

3.2.2 Load least significant MIR

IOX HMIRR % 164021

The TSC A register is transferred to a buffer register on the RSC (refer to Section 3.5) to simulate bits 0 - 15 in the ROM.

3.2.3 Write Test Status

IOX HWTST % 164025

The TSC A register is transferred to the test system control register. The control word has the following format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		STOP	MCLIR	SCOPE	BUSDR	ROMDR	OPINT		FLICK		EWBUS				TMODE

Figure 3.2 WTST Format

3.2.3.1 TMODE

This bit defines the mode of operation of the ROM simulator. If TMODE = 1; the real ROM* will be enabled in the 1/2K address space defined on the RSC (normally 0 - 1/2K). Refer to Section 4.1.12.

---

* Real ROM is an option to NOTEDS

3.2.3.2 EWBUS

This bit enables the WBUS register onto the CUT IB-bus. (Refer to Section 3.2.3.10.)

3.2.3.3 FLICK

Each time FLICK = 1 two LED's on the RSC is flicked. This is used to signal an error in the test loop.

3.2.3.4 OPINT

OPINT = 1 simulates an operator's panel interrupt.

3.2.3.5 ROMDR

ROMDR = 1 gives a data-ready pulse signalling that simulated ROM data is present in MIRL and MIRR.

3.2.3.6 BUSDR

BUSDR = 1 gives a bus data-ready pulse to signal completion of the read or write operation performed. (Refer to Section 3.2.3.10.)

3.2.3.7 SCOPE

This bit will cause a special pulse each time it is 1. Its use is to facilitate stable oscilloscope triggering in the test loops.

3.2.3.8 MCLIR

MCLIR = 1 will issue a CUT Master Clear pulse.

3.2.3.9 STOP

This bit will enable the CUT CPU stop signal when equal '1'.

Note: This bit will also be set by hardware if the CUT CPU does a TRA from register 9. (Refer to Section 3.2.7.7.)

3.2.3.10 Timing

For ease of use there is a small delay on the BUSDR signal. Hence, the EWBUS and BUSDR may be applied in the same instruction without any risk of racing.

3.2.4 Write to Bus Buffer Register

IOX HWBUS % 164027

The TSC A register is transferred to a buffer register. This buffer register may be enabled onto the CUT IB-bus by setting the EWBUS bit in the test status word (see Section 3.2.3.2 and 3.2.3.10).

3.2.5 Read Micro Program Counter

IOX HRMPC % 164020

The μ-program address presented to the RSC is transferred to the TSC A-register.

3.2.6 Read Memory Address

IOX HRMAD % 164022

The CUT MR-bus is transferred to the TSC A register.

3.2.7 Read Test Status

IOX HRTST % 164024

The test system status word is transferred to the TSC A register.

The format of the status word is as follows:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRRO	TRAO	STOP					MOOR			IREQ	FREQ	WREQ	RREQ		ROOR

Figure 2.3 Test System Status Word

3.2.7.1 ROM out of range (ROOR)

ROOR = 1 signals that there is a request to the ROM that has not been acknowledged by a ROM data-ready.

3.2.7.2 Read Request (RREQ)

This bit signals a memory read request. It is enabled by the MOOR-bit (see Section 3.2.7.6).

3.2.7.3 Write Request (WREQ)

This bit signals a memory write request. It is enabled by the MOOR-bit.

3.2.7.4 Fetch Request (FREQ)

This bit signals that an instruction fetch is requested from memory. It is enabled by the MOOR-bit.

3.2.7.5 Indirect Request (IREQ)

This bit signals that an indirect reference is requested from memory. It is enabled by the MOOR-bit.

3.2.7.6 Memory out of range (MOOR)

This bit equals 1 if the latest memory request is not satisfied by a data ready pulse from memory (or the test system). A further specification as to the type of request may be found in bits 2-5. (See the Sections 3.2.7.2 to 3.2.7.5.)

3.2.7.7 STOP

This bit shows if the CPU stop-signal is on. It will be set by bit 13 in WTST (see Section 3.2.3.9), or by the CUT if it executes a TRA 9 μ-instruction.

3.2.7.8 TRA 0

This bit is set if the CUT does a TRA 0 instruction.

3.2.7.9 TRR 0

This bit is set if the CUT does a TRR 0 instruction.

3.2.8 Read CUT IB-bus

```
IOX  HRBUS          %  164026
```

The CUT IB-bus is transferred to the TSC A register.

3.3 Test Connector Card

The test connector card (TCC) resides in the CUT where it is placed in the spare memory address slot (position 18). The purpose of the TCC is to interface the CUT to the TSC. It merely consists of the necessary tranceiver logic.

3.4 Test System Card (TSC)

The test system card resides in card slot 17 of the CUT. This is the slot normally occupied by the Panel Driver or its equivalent.

TSC holds the necessary logic to read and write the CUT IB-bus (RBUS and WBUS), including the WBUS buffer register. In addition functions normally belonging to the Panel Driver is located on this card.

These functions include the following:

- Master Clear (MCL).
- Bus Data Ready (MDRY).
- Operators Panel interrupt signal (OPINT).
- Stop signal (STOP).
- TRA0 and TRR0 status bits.

The STOP signal is set from the TSC (see Section 3.2.3.9) or from the CUT if a TRA9 μ-instruction is executed. The status of the STOP signal can be read from RTST (see Section 3.2.2).

3.5 ROM Simulator Card (RSC)

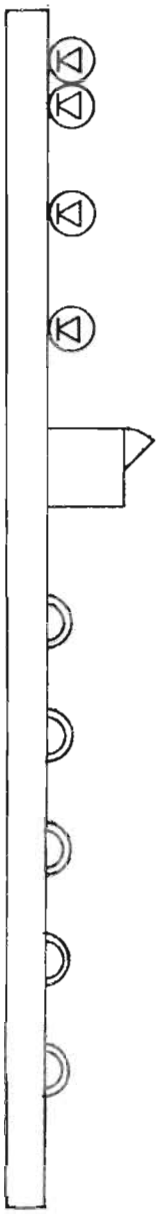
The ROM simulator card replaces the normal CUT ROM card (postion 10).

Two versions of the RSC may be delivered;

- ROM simulator alone.
- ROM simulator combined with 1/2 K real ROM.

The latter has the possibility of running in mixed mode during simulation. The main test system does not use the real ROM option and may be run with the simplest version.

In the following refer to Figure 3.3.



FLICK, refer to Section 3.5.1

SMEM, refer to Section 3.5.2

ROOR, refer to Section 3.5.3

DELAY, refer to Section 3.5.7

ERR, refer to Section 3.5.8

DSKPE, refer to Section 3.5.7

SKPE, refer to Section 3.5.4

ROMSKPE, refer to Section 3.5.6

ROMDRY, refer to Section 3.5.5

Figure 3.3 ROM Simulator Card, End View

3.5.1 FLICK

These two LED (Light Emitting Diode) will flick on and off each time NOTEDS discovers an error. This feature is used to display an indication of error frequency.

3.5.2 SMEM

This LED is on when a request to simulated main memory is on.

3.5.3 ROOR

This LED is on when a request to simulated ROM is on.

3.5.4 SKPE

On this test pin there is a positive going signal once per error loop. Refer to the appropriate error loop description. (Appendix A.)

3.5.5 ROMDRY

On this test pin there is a positive going signal each time ROMDRY is given from the test system.

3.5.6 ROMSKPE

This test pin presents the logical or between SKPE and ROMDRY. The SKPE pulse is wider than the ROMDRY pulse. Suggested use is to connect one scope channel to this point and the external trigger input to the SKPE signal. In this way one channel can be freed for other purposes. The ROMDRY and SKPF pulses can still be distinguished as they have different widths.

3.5.7 DSKPE

On this test pin a delayed SKPE pulse will occur.

The delay is given in ROMDRY pulses and is set by the DELAY switches. The delay may be varied from 0 - 15 ROMDRY pulses.

The DSKPE functions as follow:

Each time a SKPE pulse is given, a 4-bit counter is loaded with the content of the DELAY switches. This counter counts down on each of the following ROMDRY pulses. When the counter hits zero, DSKPE happens.

This feature may be useful when the SKPE pulse timing does not fit the problem at hand.

3.5.8 ERR

This test pin goes positive each time an error is detected in the test loop.

4 TEST SYSTEM USAGE

NOTEDS contains a number of separate tests. Each test is aimed at testing a particular part of the N-10 CPU or NOTEDS hardware. The tests that work on the CPU are named T1 ··· T<n>. Some special tests are named TX1 ··· TX<n>. These include system link tests, memory check etc.

The names of the tests that are to be included in a test run are listed in the Program Table. Normally one will work with a standard sequence (increasing test numbers) where the CPU is tested in a logical manner. However, it is possible to modify the program table so that it contains only selected tests (see Section 4.1.11).

The test run may be started at any test in the table. When the end of the program table is encountered, the test system will automatically loop back to the top of the table.

Running tests can be interrupted any time by simply typing a command to the test system monitor.

When NOTEDS discovers an error, an appropriate error printout will be presented on the display. The exact format of this printout differs from test to test, but generally it obeys the following:

i) Naming of the test that fails.

ii) The expected and actual result, bit by bit.

iii) Relevant parameters and data used in the test.

When the error printout is completed, the test system will repeat the test continously using the same test conditions. This error loop will be maintained even if the test should stop failing. A scope-trigger signal is presented once every loop. This signal is under program control and is given just ahead of the critical part in each test. Two LED's are flicked each time an error is detected. This provides an indication to whether the error is permanent or intermittent in nature. The error loops are documented in Appendix A.

4.1 Test System Monitor Commands

As already noted, the test system will accept commands at any time. When no tests are running, the test system monitor will announce itself with the herald:

)

The available commands are described in the following:

4.1.1 A - Call Assembler

The A command will cause an exit to the assembly system (or other external system), if available.

Initial linking can be performed by setting the assembler's start address in ASMBL. Refer to Section 4.1.12.

4.1.2 B - Back, restart current Test

The B command will restart the current test.

4.1.3 C - Continue current Test

The C command will continue the current test with the next data set. Refer to the appropriate test description.

4.1.4 D - Debug, Register examine/deposit

The D command will enter a special routine where most of the CUT registers may be examined and modified. This routine uses the herald ':'. Exit from the Debug routine will take place when the 'D' command is detected.

The general format is:

:<register name> <level>/ <old content> <optional new content>↵

Example:

:A13/- 177037 123456↵

The old content of the A register on level 13_8 (177037) will now be replaced by the new content 123456.

If the old content is immediately followed by a CR, then the register content is not disturbed.

Some registers can only be read. For these registers an automatic CR/LF will be given immediately after the old content. Likewise, some registers may only be written into. In this case there will be no display of old content following the /.

Examples:

:IIC/ 000010↵
:IIE/ 000↵

The special register name ALL means all registers on current * level. The following registers are included in the ALL command:

AC1, AC0, SH1, SH0, SS, SP, SCR, SC, H, R
T, A, D, X, B, P, L, STS, MPC,

The AC0 (least significant sum register) can only be correctly displayed if there is no real memory connected to the CUT. This fact is signalled by MMASK (refer to Section 4.1.12). An appropriate warning is given if AC0 is disturbed.

Note: The MPC, AC and H registers will be correctly displayed, but are disturbed during the ALL command.

Following is a list of legal register names.
Note that if the level code is left out, level 0 is assumed.

Legal register names:

Z, Z0 - Z17	Zero register
D, D0 - D17	D register
P, P0 - P17	P register
B, B0 - B17	B register
L, L0 - L17	L register
A, A0 - A17	A register
T, T0 - T17	T register
X, X0 - X17	X register
S, S0 - S17	STS register (as A operand)
SCR, SCR0 - SCR17	Scratch register
SP, SP0 - SP17	Scratch register (saved P)
SS, SS0 - SS17	Scratch register
R	Address bus
H	H register (hard to get at in normal cases)
MPC	Micro-program counter (Note: Disturbed by Master Clear.)
SC	Shift counter
SH1, SH0	Most/least significant shift register
BM0 - BM17	Bit mask, bit 0 - bit 17_8
XR	R register (read as A operand)

* For all practical purposes this will be level 0.

Read	Write	Internal Register
STS	STS	Status register
PGS		Paging status register
	PCR	Paging control register
PVL		Previous level
	MISC	Miscellaneous register (PON, ION etc.)
IIC		Internal interrupt code register
	IIE	Internal interrupt enable register
PID	PID	Priority interrupt detect register
PIE	PIE	Priority interrupt enable register
PES		Parity error status register
	CAR	Computed address register
MPA		MPC (μ-program counter)
	IR	Instruction register
PEA		Parity error address register

4.1.5 L - List tests in Program Table

The L command lists the content of the program table. The print-out may be stopped at any point by typing 'space' on the display keyboard.

4.1.6 M - Master Clear

The M command issues a hardware CUT* Master Clear pulse.

4.1.7 N - Start next Test in Table

The N command aborts the current test (if running) and starts the next test found in the program table.

4.1.8 P - Start previous Test in Table

The P command aborts the current test (if running) and starts the previous test found in the program table.

4.1.9 R - Restart at beginning of Test Table

The R command will restart the test run at the beginning of the test table.

---

* CUT - Computer under test.

4.1.10 S - Step Function in current Test

The S command will step the current test to the next logical function (if applicable). Refer to the actual test description for details.

4.1.11 T - Form new Test Table

This command is used to modify the standard sequence of tests. It operates on the test table in a sequential 'examine' - 'optional deposit' mode.

Example:

```
)T
>OLD    (NEW)
>T1     ↵
>T2     T3↵
>T3     T99↵
>②
)
```

T1 is maintained as 1. test, but T2 is replaced by T3 and T99 and is inserted instead of T3. T99 is no test, it simply signals end of test run and restarts at the top of the test table.

② signals end of modification and returns to the monitor. Note that all entries not specifically changed will remain untouched.

4.1.12 U - Update Test System Parameters

The U command makes available an examine/deposit function on the test system parameters. Symbolic names are used to specify the parameters.

General format:

<parameter name> / <old value> <new value>↵

The < new value > is optional. Exit from the parameter update routine is by a '② ' command.

The following is a list of available parameters and their functions.

AMODE - If this parameter is ≠ 0, the tests will be terminated when an error is detected and no Master Clear pulse is issued automatically. This mode can be used to study a 'frozen' CPU if hang-up problems occur.

Normal value: 0

MASKP - Main comparison mask. A '0' in this mask will suppress errors in the corresponding (data) bit.

Normal value: 177777

RMASK - Bit mask that flags the registers (0 - 17) to test. The registers should be legal A-operands and destinations, i.e.: D, B, L, A, T, X, SCR, SP, SS.

Normal value: 150372

LMASK - Bit mask that flags the levels (0 - 17) to test in register addressing and register data tests.

Normal value: 177777

FLMSK - Bit mask that flags the logical ALU functions (0 - 17) to test.

Normal value: 177777

FAMSK - Bit mask that flags the arithmetic ALU functions (0 - 17) to test.

Normal value: 177777

SMASK - Bit mask that flags the shift functions to test (0 - 17, MIR12-15 in LOOP instruction).

Normal value: 177777

CMASK - Bit mask that shows the conditions to test (0 - 17, MIR12 - MIR15).

Normal value: 177777
(Note: 0 - 7 are unconditional conditions.)

CYMSK - Bit mask to flag memory cycles to test (0 - 7).

Normal value: 000377

BMASK - Bit mask to flag bits to test in the bit mask (0 - 17).

Normal value: 177777

IMASK - Interrupt level mask. Flags interrupt levels to test (0 - 17).

Normal value: 177777

IMIR
IWTST
IWBUS
IMPC
IMAD
IRTST
IRBUS
- Bit masks that shows which bits should be complemented to compensate for hardware invertion of the data. It will normally not be necessary to change these masks. They are included for historical reasons. The names refer to logical test system links: MIR (MIRR and MIRL), WTST, WBUS, MPC, MAD, RTST and RBUS.

TMODE - Test mode

= 0 The simulator (refer to Chapter 5) will use simulated ROM

= 1 The simulator will use real ROM (1/2K)*

Normal value: 0

MMODE - Memory Mode

Bit mask to show how much real memory is connected to the CUT. Each bit signals a 8K block, 0 - 128K. Care should be taken that this mask describes the actual configuration at all times. Refer also to Section 5.1.7.

ASMBL - Start address of assembler if available. If not available, a 0 could be deposited.

Normal value: Address of routine FAKE

ROM - Start address of simulated ROM in TSC main memory. Care should be taken that addresses 2*(ROM) < 2*(ROM) +2*(ROMZ) do not conflict with test system programs.

ROMZ - Size of simulated ROM in 32-bit words.

Normal value: 2000 (1K ROM)

IOT - IOT break address in micro-program.

Normal value: 177777 (no break) or 170

MON - Micro program break address for main memory break point.

Normal value: 2217 (entry point for 143700).

---

* Real ROM is an option to NOTEDS hardware.

WAIT - Micro-program break address for trapping WAIT instructions.

Normal value: 244 (WAIT entry point).

MBPC - Main memory break point code.

Normal value: 143700
(Note: (MON) should always contain the entry point address of the MBPC.)

4.1.3 X - Call Simulator

The X command will switch to the simulator monitor, if available. This monitor announces itself with the herald:

$

Return from the simulator monitor is by X or @ command.

4.1.4 ! - Start specified Test

The ! command will ask for a test name. When this name has been typed in, the test will be started. If no errors are detected, the test system will continue with the next test in the program table.

Example:

```
) !
TEST NAME: T20
NOW RUNNING: T20  TG  CONTROL
```

4.2 Available Tests

The following is a summary of available tests. For more details, refer to Appendix A.

4.2.1 Standard Test Sequence (T <n>)

T0 - TSC ← CUT IB ← TSC

Transfers data to and from the CUT IB bus. Used to test the IB bus. The test does not involve the CUT CPU.

(Note: This test may be absent in some versions of NOTEDS. In this case TX5 will serve the same purpose.)

T1 - TRANSFER TO/FROM H-REGISTER

Transfers data to and from the CUT H register (data input register). This is a very basic test.

T2 - A AND D REGISTER TRANSFER

Transfers data to and from the A and D registers in a very basic way. This test is used to establish contact with CUT central registers.

T3 - STANDARD ROUTINES

Basically the same test as T2, but uses more general software routines.

T4 - LOGICAL ALU FUNCTIONS

Test the logical ALU functions between A and D registers.

T5 - REGISTER ADDRESSING

Tests the addressing of all level dependant registers.

T6 - REGISTER DATA

Tests the registers more fully with regard to data combinations.

T7 - REGISTER DISTURBANCE

Very extensive register test. Performs a thorough distrubance test on all general registers.

T8 - TRANSFER TO/FROM STS REGISTER

Tests the setting and reading of the STS register (as A-operand).

T9 - ARITHMETIC ALU FUNCTIONS

Full test of arithmetic ALU functions including carry input and output.

T10 - CONDITIONAL ARITHMETIC

Tests the conditional arithmetic.

T11 - ABSOLUTE JUMP

Tests the micro-processor absolute jump.

T12 - CONDITIONAL JUMP

Tests the micro-processor conditioned jump.

T13 - COMPUTED ADDRESS JUMP

Tests the micro-processor computed address jump. This also tests the setting of the 12 lower bits of the instruction register.

T14 - BIT MASK

Tests the bit mask B operand.

T15 - SC - SETTING

Tests the setting and reading of the shift counter.

T16 - TRANSFER TO/FROM SHIFT REGISTER

Tests the data transfer to and from the shift register.

T17 -

T18 - DELTA - H

Tests the sign-extended half part of the H register.

T19 - LOOP SHIFT

Tests the LOOP instruction shift functions, including the link flip-flop input and output.

T20 - TG - SETTING

Tests the setting of the TG flip-flop (floating point rounding control).

T21 -

T22 -

T23 - FETCH CYCLE

Test of fetch cycle request setting.

T24 - MICRO-ADDRESS GENERATOR

Tests all micro-processor entry points (excluding interrupt entry points).

T25 - MEMORY REQUESTS AND ADDRESS ARITHMETIC

Test all memory cycle request settings and the address arithmetic.

T26 - FETCH, CONDITIONALLY ADD 1 TO P

Test of special feature, P as destination register together with memory reference relative to the P register.

T27 - MPC COUNTING

Tests the MPC for proper counting.

T28 - P COUNTING

Tests the P register for proper counting.

T-29 - R COUNTING

Tests the R register for proper counting.

T30 - END-INPUT FOR DIVISION

Tests the end-input to the shift register during division loops.

T31 -

T32 - SPECIAL ENTRY POINTS

Test the entry point for STOP, MASTER CLEAR and PANEL INTERRUPT.

T33 - 1. INTERRUPT TEST (MPC)

Tests the entry point for external interrupt.

T24 - 2. INTERRUPT TEST (PIL)

Tests correct program level after interrupt and level shift.

T35 - 3. INTERRUPT TEST (PIL)

Tests correct program level during two succeeding level shifts.

T36 - 4. INTERRUPT TEST (PVL)

Tests the previous level (code) after level shift.

T37 - INTERRUPT REGISTER SELECTION

Tests that the correct register set is selected for all program levels.

T99 - END-OF-TEST

Fake test to signal end of test run. Program will loop back to top of the program table.

4.2.2 Special Tests (TX <n>)

The following is a short description of some special tests. These tests are not included in the normal test run, but may be called upon to test special hardware or in special test situations.

TX1 - WTST ← OPR

Transfers TSC OPR to the WTST (write test status) register. The (proposed) software inversion of data is included meaning that a '1' in QPR should result in a '1' in WTST. No error printout.

TX2 - MIR ← OPR

Transfers TSC OPR to CUT MIRR and MIRL. Software inversion is included. No error printout.

TX3 - WBUS ← OPR

Transfers TSC OPR to CUT IB-bus. Software inversion is included. No error printout.

TX4 - REPEAT SINGLE MICRO-INSTRUCTION

This test asks the operator for a micro-instruction (see Section 5.1.14 for details regarding the format). The operator also decides whether a Master Clear should follow each cycle. The micro-instruction is then repeated continously with Scope-pulse and (optional) Master Clear. No error printout.

TX5 - TSC ← CUT IB ← TSC

Tests the basic test system data transfer by setting and reading the CUT IB-bus. This test only involves the IB-bus and not the CUT CPU.
Error printout.

TX6 - SINGLE REGISTER WRITE/READ

Writes and reads a specified pattern to and from a specified register. This register must be a legal A operand and destination (i.e. D, B, L, A, T, X, SS, SCR, SP).

Format of register/pattern specification (only current level):

<register name> / <pattern>

Example:

A/123456

Error printout.

TX7 - SIMPLE OR-TEST

This test is aimed at testing the or-logic and associated data paths.

The operator specifies a micro-instruction which normally contains the required or-selection. During the test the TSC OPR is transferred to CUT IR (CAR) before the micro-instruction is executed.

No error printout.

TX8 - PROGRAMMED MICRO-SEQUENCE

The operator may program a small (maximum 10) sequence of micro-instructions (see Section 5.1.14 for related format). This sequence is run repeatedly with a Scope and Master Clear pulse for each pass.

If a (unsatisfied) memory request occurs (MOOR=1), a data-ready pulse will be given.

No error printouts.

TX9 - SIMPLE IOX/IOT TEST

The operator specifies a IOX or IOT instruction and a (optional, special) data pattern. The IOX/IOT is transferred to the CUT IR (CAR) and the micro sequence for these instructions are executed. If the CUT A register is changed, an error printout will result.

TX10 - MEMORY TEST

Tests the main memory. The operator must specify:

1. data pattern : (DP(0))
Data increment : (DI)
Address limits : (LL < UL)

The data pattern is stored in the specified memory and then read back and tested. The actual data pattern is:

$DP(i) = DP(i-1) + DI,\ i \in (LL, UL)$

In this manner it is possible to test with a fixed pattern or with a 'address in address' type pattern.

4.3 Setting up NOTEDS

Setting up NOTEDS is very simple. The four special cards mentioned earlier should be plugged in as follows.

In the TSC:

The Test Driver may be plugged into any vacant I/O slot. Remember, however, to disconnect any device cable connected to the backwiring in that slot.

In the TSC:

The ROM Simulator plugs into slot 10.

The Test System plugs into slot 17. Remember to disconnect the cable connecting the operators panel.

The Test Driver plugs into slot 18.

Disable Memory and I/O timeout by shortening the two test pins on the CUT Interrupt Control.

If the tests should be run, disconnect any real memory in the CUT, otherwise T23, T24 and T25 will not work properly.

5 SIMULATOR SUBSYSTEM

As an extention to the test system, a simulation system may be added.

The simulator provides means for observing CUT* behaviour when performing longer instruction sequences than included in the normal tests.

The simulator will be particularly useful when debugging customer specified instructions. Provisions are made for mixed mode operation, that is; the simulator may substitute the whole or part of the main and/or read-only memory (ROM).

The test system command link includes signals to show different memory requests. The simulator reads and decodes these requests and provides the appropriate data and data entry pulses to the CUT.

The simulator also contains a number of service routines included for ease of use.

5.1 Simulator Monitor Commands

The available commands when in the simulator, are described in the following. Generally these commands may be typed at any time, if a simulation happens to be running, the execution will be interrupted and the command acted upon.

5.1.1 A - Call Assembler

This command is similiar to the test system A command. See Section 4.1.1.

5.1.2 C - Compare

The C command will perform a comparison between CUT and TSC** memory between the limits specified in the last " < command (see Section 5.1.15). If a difference is detected, an error printout result:

< address > / < TSC content > < CUT content >

Note: The comparison may be interrupted by typing a command to the monitor.

5.1.3 D - Debug, Register examine/deposit

This command is similiar to the test system D command, see to Section 4.1.4.

---

* CUT - Computer under test.
** TSC - Test system computer.

5.1.4 L - Load from Paper Tape

The L command will load a standard Binary Format tape (as dumped by the MAC command)BPUN) from the TSC tape reader into CUT memory. The tape will be loaded into real or simulated CUT memory according to the last S command. The binary program is not started after successfull loading.

Note: There is no checking of simulated core limits. Care should be taken not to overlay the test system.

5.1.5 M - CUT Master Clear

The M command issues a CUT Master Clear pulse.

5.1.6 P - Print Simulation Log

The P command will dump the last (up to 48) references to simulated main or ROM memory on the test system console. This simulation record should be read columnwise, the last reference being the last in the rightmost column.

Each simulator reference is described by three numbers.

Format of reference:

< 3 digit request word > < 6 digit data > < 6 digit memory address >

Request word

The request word shows the reference type, and it has the following format:

8	7	6	5	4	3	2	1	0
MOOR	SOC*	SIC*	IREQ	FREQ	WREQ	RREQ		ROOR

Figur 5.1 Request Word

MOOR - Memory out of range. If this bit is set, the reference is a reference to simulated main memory. Further specification is given by IREQ, FREQ, WREQ and RREQ.

SOC* - Memory output command.

SIC* - Memory input command.

---

* Not included in the standard test system.

IREQ - Indirect reference.

This bit is enabled by MOOR and signals a request for an indirect address.

FREQ - Fetch request.

This bit is enabled by MOOR and signals a request for an instruction.

WREQ - Write request.

This bit is enabled by MOOR and signals a CPU data write request.

RREQ - Read request.

This bit is enabled by MOOR and signals a CPU data read request.

ROOR - ROM out of range.

This bit signals a reference to simulated ROM.

Note: The MOOR bit has higher priority than the ROOR when request type is decoded.

It is important to note that bits 2 - 7 are only valid if bit 8, MOOR, is set.

The following is a list of normal request words:

021 - ROM request

011

005

441, 440 - Indirect request

421, 420 - Fetch request

411, 410 - Write request

401, 404 - Read request

Data Word

This word has two different meanings, depending on the request type.

If the MOOR bit is set, the data word contains the actual data transferred to/from simulated memory.

If the MOOR bit is not set (the reference being a ROM reference), then the data word contains the micro-program address referenced.

Memory Address

This is the content of the memory address bus. In some cases the simulator will read the address bus when it is not enabled. The memory address 177777 will result. As this is of no interrest, a memory reference to address 177777 (true or false) will not be displayed as the actual number, but as 6 spaces.

The main memory references will not be "visible" if real memory is referenced.

The ROM references will not be "visible" if real ROM is referenced.

The logger table is only maintained when program execution is started by the ! command.

5.1.7 S - Set Memory Mode

The 'S' command will logically split CUT memory into real and simulated parts.

Format:

Ø S

MMODE/XXXXXX YYYYYY

XXXXXX - Old memory mode word

YYYYYY - (Optional) new memory word. Each bit = 1 flags a 8K block as belonging to real CUT memory.

Bit 0 = 1 0 - 8K
Bit 1 = 1 8 - 16K
Bit 15 = 1 120 - 128K

Care should be taken that the memory mode word always corresponds to the physical memory connected to the CUT. If this is not observed, the simulator may not function properly. The memory mode word also affects the examine and deposit function together with the L command and the main memory break point setting.

5.1.8 T - Transfer to CUT Memory

The 'T' command will copy the content of TSC core (as defined by the < command) into real or simulated CUT core (as defined by the S command).

5.1.9 U - Update System Parameters

This command is identical to the test system U command, see Section 4.1.12.

5.1.10 X - Exit from Simulator

The X command will exit from the Simulator back to the test system monitor (see also Section 5.1.18).

5.1.11 '.'- Main Memory Break Point

The '.' command has two uses. If it is given without a preceding address, it means: 'Reset break point'. If it is preceded by an (octal) address, a break point code will substitute the actual instruction in the specified address. (Any old break point will be reset first.) When the ROM simulator runs, it will check for references to the micro-program break address, ((MON)). The simulator stops if a match is found and an appropriate message is displayed together with all the registers on the current level.

Normally, the break point code is 143700 with a corresponding micro-program break address = 2217 (2217 is the entry point for the 143700 instruction.

The break point code and the corresponding break address may be modified within the parameter update routine (see Section 4.1.12).

Note: If the main memory break point is located in simulated memory, it must be within the limits defined by the '<' command (see Section 5.1.15).

5.1.12 ',' ROM Break Point

The ',' command has the same use as the '.' command (see Section 5.1.11), but operates in the simulated Read-Only-Memory.

When the break point address is referenced, the simulator will stop and display all registers on current level together with an appropriate message.

Note: If an 'all zero' micro instruction is hit, then the message 'MICRO STOP' is displayed. The simulator stops and displays all registers on current level.

This fact can be used to trap references to illegal addresses.

5.1.13 '/' - Main Memory examine/deposit

The / command is used to examine and modify the content of CUT real or simulated main memory.

Format:

$ < address >/ (#) < old content> < new content >
(#) < old content of address + 1> < new content >
(#) < old content of address + 2 > ␣
$

The < new content > is optional. CR will display the content of the next address. The examine/deposit function may be terminated by typing a space (or any monitor command).

The < old content > will be preceded by an # if the address belongs to real CUT memory (as defined by MMASK).

Note that no deposit is allowed in simulated main memory, outside the limits set by the '<' command (see Section 5.1.15).

If * is typed when in examine/deposit mode, the address of the current location is printed out.

5.1.14 '\' ROM examine/deposit

The '\' is used to examine and modify the content of the simulated ROM:

Format:

XXXX\

The content of ROM address XXXX is displayed in an appropriate format.

MIR TYPE: (A/I/J/L):Q

Appropriate heading
Z . . . . . Z

XXXX	-	ROM address
Q	-	= A - arithmetic micro instruction I - interblock J - jump L - loop CR - no deposit
ZZ .. ZZ	-	Bits 29 - 0 in the new micro instruction Note: Z = 0, 1 or 'control Z'

Edit features:

If Z = ASCII 32*, then the previously typed bit is deleted.

If Z ≠ 0, 1 or ASCII 32, then the micro instruction typed so far is dropped and the question 'MIR TYPE' is asked again.

For details regarding the micro instruction format, refer to the micro processor manual or Appendix B.

5.1.15 '<' Set Memory Limits

The '<' command is used to define CUT simulated core limits.

Format:

<lower limit> <upper limit>

Note: Both limits inclusive.

When running, the CUT is not permitted to reference simulated memory outside the defined address space. The simulation is stopped and an appropriate message is displayed if this should occur.

Note that the limits defined by '<' also is referred to by the /, C, and T commands.

5.1.16 ! - Start CUT Execution

The '!' command is used to start the CUT at a specified address.

Format:

<start address>!

---

* ASCII 32 = control Z or the 'move cursor back' command on the display.

The following sequence of operations is included:

- Clear logger table (see Section 5.1.6).
- Do a CUT master clear
- Set CUT P register
- Set test mode (see Section 4.1.12)
- Do a CUT Fetch cycle

A default start address is assumed if the ! command is given without an address. This default address (DFAD) is set as follows:

- When hitting a main memory break point:
  DFAD:= break point address
- When hitting a WAIT instruction:
  DFAD:= instruction address + 1

Thus a single ! can be used to resume operation after a break point or WAIT stop (the break point must first be reset).

During the execution the simulator will check for limits and break conditions. All references to simulated ROM or main memory will be logged (refer to Section 5.1.6).

5.1.17 ↑ - Start CUT Execution

The '↑' command is very similiar to the '!'command. However, there is no logging or memory limit checking. This increases simulation speed.

Note: This mode of operation should be used with care. A CUT program running wild may destroy the test system programs.

5.1.18 R - Load ROM Program

The R command will load the simulated ROM from the TSC tape reader.

Format:

```
ØR
LOAD FROM ROM ADDRESS: <first address>
```

<first address> is the first ROM address to be loaded from the tape in question.

Tape format:

- Everything up to the first "/" will be skipped.
- Each micro-instruction is read as two octal numbers, each terminated with an non-octal character (LF is completely ignored).

- Loading will terminate with the occurence of '$\mathcal{D}$' on the tape.

The tape format is compatible with the one dumped by the assembler commands)PRINT or)PUNCH.

5.1.19 $\mathcal{D}$ - Exit from Simulator

The $\mathcal{D}$ command will exit from the simulator and back to the test system monitor.

5.2 ROM Program

Included in the simulator is a modified version of the NORD-10 μ-program. The following changes have been made:

1) The STOP entry point (=0) will jump to μ-address 7776.

2) The MASTER CLEAR entry point (= 1) will jump to μ-address 7777.

3) The PANEL INTERRUPT entry point will jump to μ-address 7775.

4) The WAIT instruction will not work with the interrupt system turned on.

5) MOPC will not support the PANEL STATUS register.

6) MOPC will simulate the ALD-register by treating the 16-bit word preceding $ or & as if this number was read from ALD (thus bit 12 - 15 carry a special meaning). Note, however, that $ or & overrides bit 12.

APPENDIX A

ERROR LOOP DESCRIPTIONS

This appendix contains information related to the error loops that the program performs when an error has been detected. As already noted, an effort has been made in each loop to maintain 'static' conditions, i.e. test data is kept constant. Normally, these conditions are presented in the error printouts. The following contains examples of error printouts and an exact description of the micro-instruction sequence that is included in each error loop.

Notation

Each error loop is divided into micro-step, each micro-step is the action performed during one micro-instruction.

Two columns, labelled MC and SC are used to specify the times at which the Master Clear and Scope pulse are triggered. These two columns are given a higher resolution than the micro-step to enable some degree of timing information.

Example:

Micro step	MC	SC
	a	a
i	b	b

An asterisk (*) in fields 'a' signals that the associated pulse is triggered in between the RDRY (ROM Data Ready) pulse for micro-steps 'i-1' and 'i'.

An asterisk in fields 'b' signals that the associated pulse is triggered syncronous to the RDRY for step 'i'.

The micro-step description is given in two parts. The 1. line describes the action and the 2. line gives the exact micro instruction.

In describing the action the following notation is used:

Registers are described by a two-symbol group enclosed in brackets:

[<level>,<register>]

<level> is a number or name designating a program level (octal notation). An asterisk (*) is used to denote 'current level'. Current level is the level contained in the PIL registers (STS bits 8-11).
<register> is a symbol naming the actual register.

Example:

[12, A]	A register on level 12_8.
[QQ, X]	X register on level QQ. The value of QQ can be found in the error printout.

The brackets and level designation are dropped if the register is level-independent.

Example:

H

The symbol '←' is the transfer operator. The left side is always destination. In some cases there is no destination. This means that destination code = 0 is used.

Examples:

P ← H	H register is transferred to P.
[15, X] ← [*, A]	A register on current level is transferred to the X register on level 15_8.
←[*, A] - [*, D]	D register is subtracted from the A register (on current level). The result is put nowhere.

IB is sometimes used as source or destination. This means that data is transferred from or to the NOTEDS link registers via the IB-bus.

As noted, the action description is accompanied by an exact micro-instruction description. Normally, this description is using MIC-MAC mnemonics (refer to the micro-processor specification).

In some cases it is necessary to enter variable expressions into the MIC MAC description. Frequently the letters Q and Y are used for this purpose. By relating the comment with the error printout, the actual variable can be decided.

Example:

[*, D] ← [*, D] 'op' [*, A]	Y-arithmetic select.
LOGY XXXX D,D B,D A,A	XXXX- ALU function.

The error printout will tell which arithmetic that failed (Y), the ALU function (XXXX) and the operation ('op').

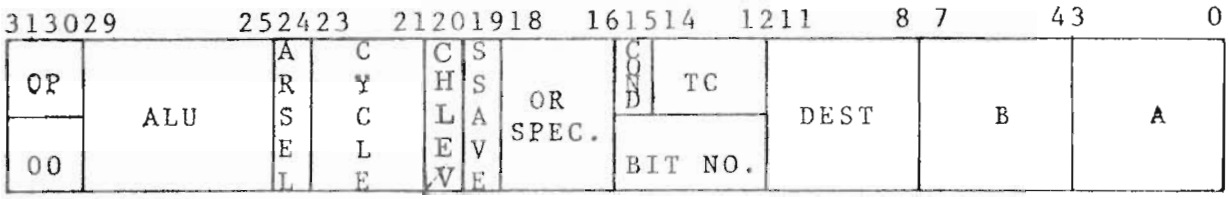
Use of the Error Loop Charts

Normally, two channels are used to display the SKPE and MIRKL (or ROMDR). These signals form a frame of reference. Note that while the charts present the error loop in a functionally cronological order, the oscilloscope picture will (if syncronized with the SKPE-pulse) display the 'most interresting' part first. It may happen that the normal SKPE-pulse is not suitable for the problem at hand. In these cases the possibility to use an delayed Scope-pulse (DSKPE) may be useful. Refer to Section 3.5.7 for details.

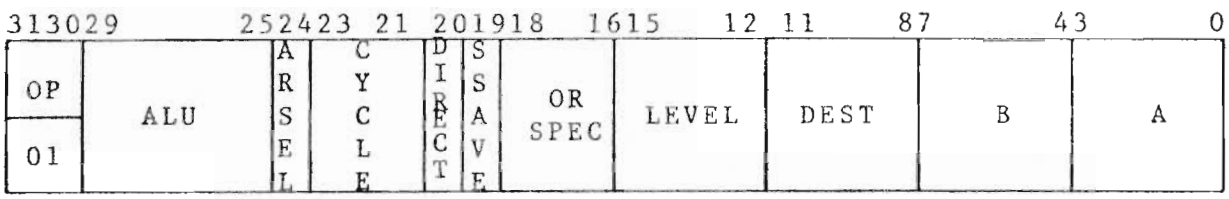
APPENDIX B

μ-INSTRUCTION FORMAT

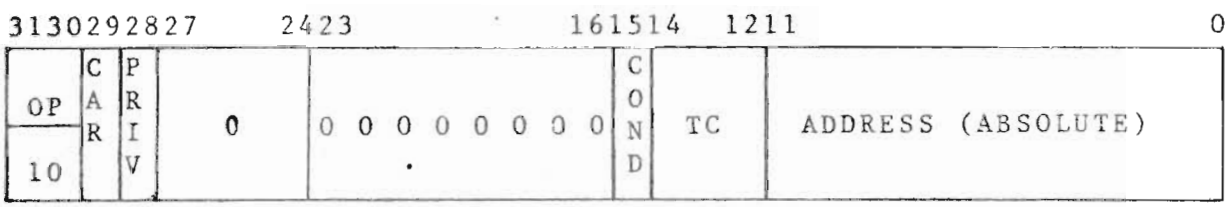
ARITM:



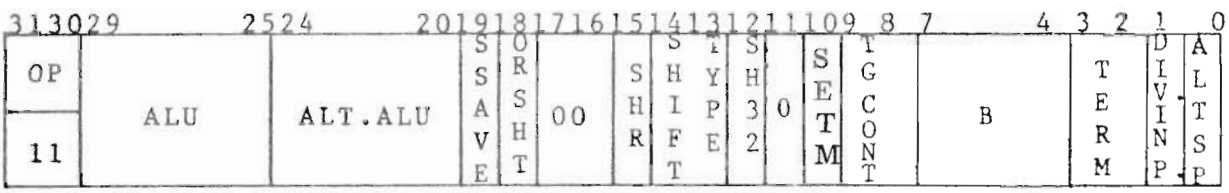
INTERBLOCK:



JUMP:

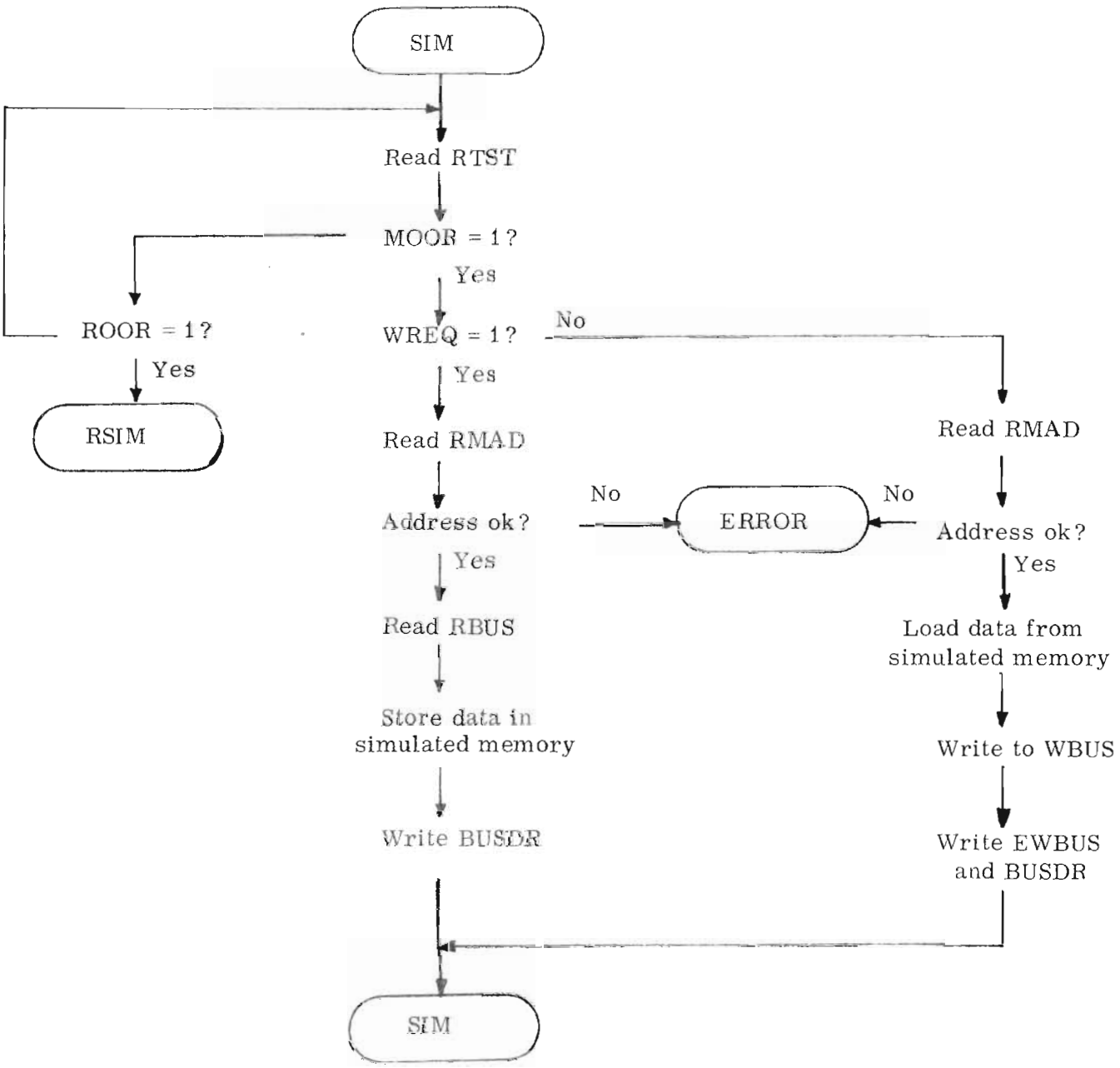


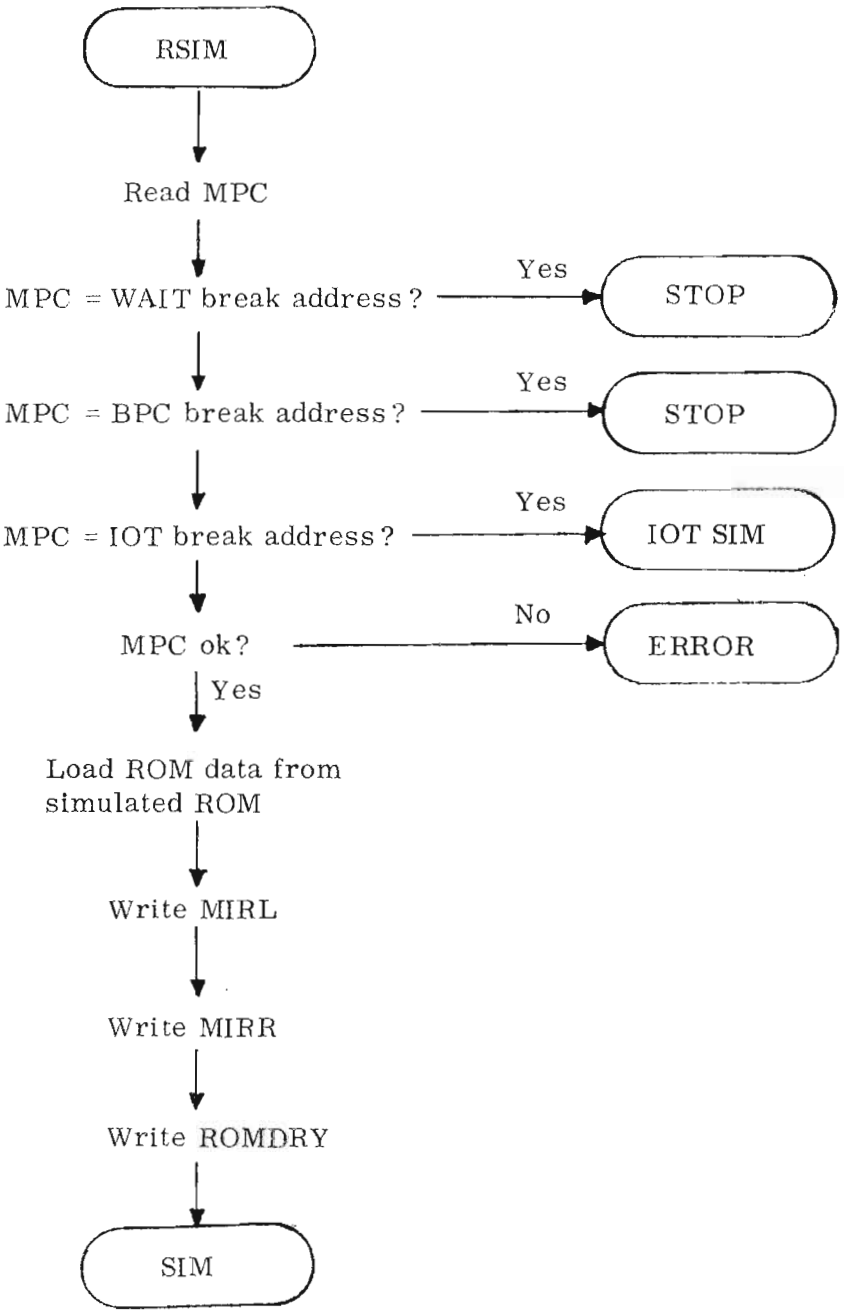
LOOP:



APPENDIX C

FUNCTIONAL FLOWCHART OF SIMULATOR





ERROR - Error printout.

STOP - Simulation stopped, printout.

IOT SIM - (IOT simulated, NORD-1 versions only.)

A/S NORSK DATA-ELEKTRONIKK
Lørenveien 57, Oslo 5 - Tlf. 21 73 71

COMMENT AND EVALUATION SHEET

ND-62.004.01 NOTEDS Reference Manual

In order for this manual to develop to the point where it best suits your needs, we must have your comments, corrections, suggestions for additions, etc. Please write down your comments on this pre-addressed form and post it. Please be specific wherever possible.

FROM: _____

_____

_____

– we want bits of the future

A/S NORSK DATA-ELEKTRONIKK ØKERNVEIEN 145 OSLO 5 NORWAY PHONE: 21 73 71 TELEX: 18284