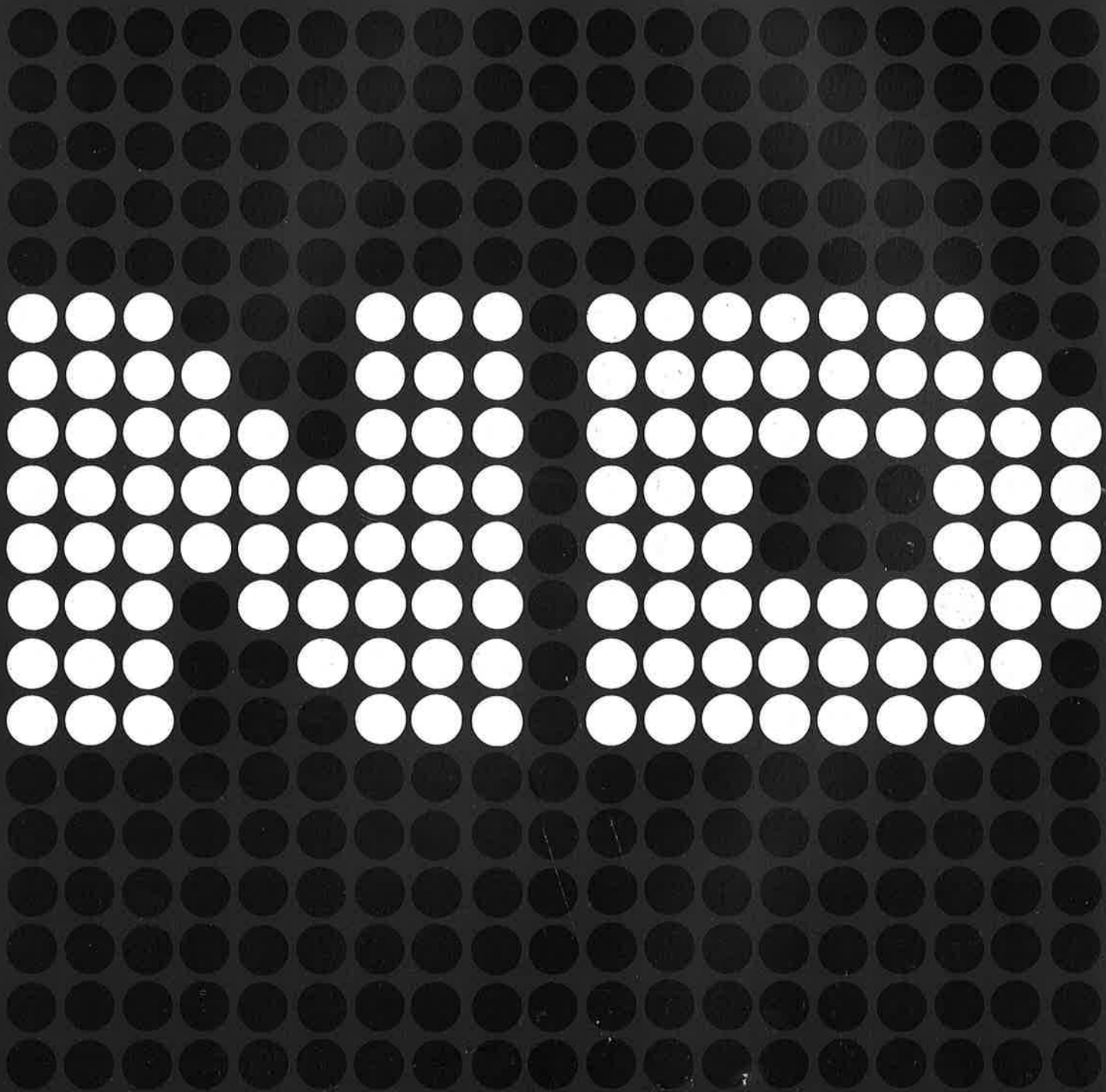
NORD-100
Input/Output System

NORSK DATA A.S



NORD-100

Input/Output System

NOTICE

The information in this document is subject to change without notice. Norsk Data A.S assumes no responsibility for any errors that may appear in this document. Norsk Data A.S assumes no responsibility for the use or reliability of its software on equipment that is not furnished or supported by Norsk Data A.S.

PRINTING RECORD

Printing	Notes
12/80	Original Printing – Version - 01

NORD-100 INPUT/OUTPUT SYSTEM
Publication No. ND-06.016.01

NORSK DATA A.S
P.O. Box 4, Lindeberg gård
Oslo 10, Norway

Manuals can be updated in two ways, new versions and revisions. New versions consist of a complete new manual which replaces the old manual. New versions incorporate all revisions since the previous version. Revisions consist of one or more single pages to be merged into the manual by the user, each revised page being listed on the new printing record sent out with the revision. The old printing record should be replaced by the new one.

New versions and revisions are announced in the ND Bulletin and can be ordered as described below.

The reader's comments form at the back of this manual can be used both to report errors in the manual and to give an evaluation of the manual. Both detailed and general comments are welcome.

These forms, together with all types of inquiry and requests for documentation should be sent to the local ND office or (in Norway) to:

Documentation Department
Norsk Data A.S
P.O. Box 4, Lindeberg gård
Oslo 10

PREFACE

TO THE READER

This manual describes the NORD-100 Input/Output system architecture and principles from a hardware standpoint.

The description is general, not aimed at any particular device controller. Hence, this book is not a maintenance "trouble shooting" manual.

The main intention of this manual is to describe the parts of the I/O system that are common to all device controllers. That is, to give the needed background to understand the fundamental design concepts in the NORD-100 Input/Output system.

In the manuals covering specific device controllers, a knowledge of these concepts is assumed.

Since this is a hardware manual, it should be of interest to:

- all technical and maintenance personnel who wish to gain a good understanding of the connection of I/O interfaces to the NORD-100 computer system.

- system software personnel who program I/O interfaces. They should read Section I.

PREREQUISITE KNOWLEDGE

It is assumed that the reader of this manual is familiar with the NORD-100 CPU at the level described in the manual "NORD-100 Functional Description".

It is also assumed that the reader has some knowledge at the NORD-100 instruction set and assembly programming.

THE MANUAL

This manual is meant to be read from the beginning to the end since some sections assume knowledge of the previous sections.

Section I describes the NORD-100 I/O system in general terms as seen by a programmer. This part does not require particular knowledge of the involved hardware.

Sections II to V deal with the hardware implementation of the I/O system functions programmed in Part I.

Section II describes the functions of the system bus (the NORD-100 bus).

Section III describes using the NORD-100 bus in programmed information exchanged.

Section IV describes the separation of interrupting I/O interfaces.

Section V covers how information is exchanged directly between the NORD-100 memory system and I/O interfaces.

Section VI covers the NORD-100 bus extender.

Related manuals containing relevant information about the NORD-100 Input/Output system are:

- NORD-100 Functional Description
- Different specially dedicated device controller manuals, available on request

GENERAL

NORD-100 is a 16 bit general purpose computer suitable for most computer applications.

Requirements to a computer system changes from one application to another. Therefore, flexibility has been one of the key words in the design of the NORD-100 computer system.

The basic difference from one installation to another is found in the memory size, the software configuration and in the input/output system, i.e., the selection of peripheral equipment.

Therefore, a flexible and modular design of the above mentioned modules is specially important in order to configure a system after a customer's wishes.

TABLE OF CONTENTS

+ + +

Section:		*Page:*
I	NORD-100 INPUT/OUTPUT SYSTEM OVERVIEW	
I.1	INTRODUCTION	I–1–1
I.1.1	Definition of Terms	I–1–1
I.2	NORD-100 ARCHITECTURE	I–2–1
I.2.1	NORD-100 Bus Structure — The NORD-100 Bus	I–2–1
I.2.1.1	Physical Arrangement of the NORD-100 Bus	I–2–2
I.2.2	Functions of the NORD-100 Bus	I–2–3
I.2.2.1	General	I–2–3
I.2.2.2	NORD-100 Bus and the I/O System	I–2–3
I.2.2.2.1	General	I–2–3
I.2.2.2.2	I/O System Information Exchange and NORD-100 Bus Usage	I–2–4
I.2.3	Organization of NORD-100 Modules	I–2–5
I.2.4	NORD-100 Configuration Examples	I–2–7
I.3	PROGRAMMING OF I/O DEVICE CONTROLLERS (INTERFACES) — THE INPUT/OUTPUT INSTRUCTIONS IOX/IOXT	I–3–1
I.3.1	General	I–3–1
I.3.2	Introduction to IOX/IOXT	I–3–1
I.3.3	Format and Functions of the IOX and IOXT Instructions	I–3–2
I.3.3.1	Definition of IOX/IOXT Transfer Direction	I–3–3
I.3.4	Calculation of the IOX/IOXT Device Register Address	I–3–6
I.3.4.1	The Device Register Address Range	I–3–6
I.3.4.2	Specification of an I/O Device Register Address for Norsk Data Designed Interfaces	I–3–9
I.3.4.3	Device Selection — the Hardware Device Number	I–3–10
I.3.4.4	Device Register Selection on an I/O Interface	I–3–11
I.3.5	Format of the Control and Status Registers for Norsk Data Designed PIO and DMA Interfaces	I–3–17
I.3.5.1	General	I–3–17
I.3.5.2	Format and Functions of the Status Register	I–3–17
I.3.5.3	Format and Function of the Control Register	I–3–20

Section:		*Page:*
I.3.6	Programming of a PIO Interface	I–3–23
I.3.6.1	General	I–3–23
I.3.6.2	Programmed Input from a PIO Interface	I–3–23
I.3.6.3	Programmed Output to a PIO Interface	I–3–26
I.3.7	Programming of DMA Controllers	I–3–28
I.3.7.1	DMA Controller Operation	I–3–28
I.4	THE I/O SYSTEM AND THE INTERRUPT SYSTEM	I–4–1
I.4.1	General	I–4–1
I.4.2	NORD-100 Interrupt System General Description	I–4–2
I.4.3	NORD-100 Interrupt System Level Assignment under SINTRAN III	I–4–6
I.4.4	NORD-100 Input/Output Device Controllers Level Usage	I–4–7
I.4.5	Identification of an Interrupting I/O Device Controller	I–4–9
I.4.5.1	General	I–4–9
I.4.5.2	The Ident Code	I–4–9
I.4.5.3	The Ident Instruction	I–4–10
I.4.5.4	The Ident Search Mechanism	I–4–11
I.4.5.5	Input/Output Interrupt Programming	I–4–12
I.4.5.5.1	Initialization of the Interrupt System	I–4–12
I.4.5.5.2	I/O Interface Interrupt Generation	I–4–12
I.4.5.5.3	Handling of I/O Interface Interrupts	I–4–16
II	THE NORD-100 BUS	
II.1	GENERAL	II–1–1
II.2	THE NORD-100 BUS – BUS REQUESTORS	II–2–1
II.3	A NORD-100 BUS CYCLE – GENERAL DESCRIPTION	II–3–1
II.4	FUNCTIONS OF THE BUS CONTROL LOGIC (BCU)	II–4–1
II.4.1	Allocation of the NORD-100 Bus	II–4–1
II.4.1.1	The Allocation Requests	II–4–1
II.4.1.1.1	NORD-100 CPU Allocation Request	II–4–2
II.4.1.1.2	DMA Allocation Request	II–4–3
II.4.1.1.3	Memory Refresh Allocation Request	II–4–3
II.4.1.2	The Bus Control Unit (BCU) Allocation Priority Rules	II–4–4
II.4.2	The Bus Control Unit (BCU) and Termination of Bus Cycles	II–4–6

Section:		*Page:*
II.5	DATA TRANSFER ON THE NORD-100 BUS — GENERAL DESCRIPTION	II—5—1
II.5.1	Organization of a NORD-100 Bus Cycle	II—5—1
II.5.1.1	The Address Cycle	II—5—2
II.5.1.2	The Data Cycle	II—5—2
III	PROGRAMMED INFORMATION EXCHANGE BETWEEN I/O INTERFACES AND NORD-100 CPU — EXECUTION OF THE IOX/IOXT INSTRUCTIONS	
III.1	INTRODUCTION	III—1—1
III.2	IOX/IOXT INSTRUCTION EXECUTION	III—2—1
III.2.1	General	III—2—1
III.2.2	IOX/IOXT Instruction Entry Point Generation	III—2—1
III.2.3	IOX/IOXT Microprogram Operation	III—2—4
III.2.4	IOX/IOXT Execution and the NORD-100 Bus	III—2—7
III.2.5	IOX/IOXT Execution and the I/O Interfaces	III—2—11
III.2.6	PIO Interface Module Organization	III—2—12
III.2.7	Hardware Implementation of the Device Identification Logic	III—2—15
IV	IDENTIFICATION OF INTERRUPTING I/O INTERFACES — EXECUTION OF THE IDENT PLxx INSTRUCTION	
IV.1	INTRODUCTION	IV—1—1
IV.2	EXECUTION OF IDENT PLxx	IV—2—1
IV.2.1	Ident Microprogram Operation	IV—2—1
IV.2.2	Ident PLxx Execution and the NORD-100 Bus	IV—2—4
IV.2.3	Ident PLxx Execution and the I/O Interfaces	IV—2—8
IV.2.4	Hardware Implementation of the Ident Control Logic on PIO and DMA Interfaces	IV—2—9

Section:		*Page:*
V	DIRECT MEMORY ACCESS (DMA) INFORMATION EXCHANGE VIA THE NORD-100 BUS	
V.1	GENERAL	V—1—1
V.2	THE DMA REQUEST GENERATION	V—2—1
V.3	DMA REQUESTS AND THE BUS CONTROL UNIT — THE BUS ALLOCATION ACKNOWLEDGE MECHANISM	V—3—1
V.4	A DMA TRANSFER AND THE NORD-100 BUS	V—4—1
V.5	DMA TRANSFERS AND THE DMA CONTROLLERS	V—5—1
VI	NORD-100 BUS EXTENDER (BEX)	
VI.1	GENERAL	VI—1—1
VI.2	FUNCTIONAL DESCRIPTION	VI—2—1
VI.2.1	Definition of Terms	VI—2—1
VI.2.2	Organization of Modules in a Bus Extended System	VI—2—2
VI.3	CONTROL OF THE BUS EXTENDER (BEX) MODULES	VI—3—1
VI.3.1	Introduction	VI—3—1
VI.3.2	The Memory Address Routing Mechanism	VI—3—2
VI.3.3	Hardware Switch Setting	VI—3—4
VI.3.4	Bus Extender (BEX) Programming Specifications	VI—3—8
VI.4	CONFIGURATION EXAMPLES	VI—4—1
VI.5	BEX INTERCONNECTION — PHYSICAL CABLE ARRANGEMENT	VI—5—1

Appendix:		*Page:*
A	STANDARD NORD-100 DEVICE REGISTER ADDRESSES AND IDENT CODES	A—1
B	INTERNAL REGISTERS	B—1
B.1	Programming Specifications for Terminal No. 1	B—3
B.2	NORD-100 4 or 8 Asynchronous Serial Interface Programming Specifications	B—5
B.2.1	NORD-100 4 or 8 Asynchronous Current Loop Programming Specifications	B—5
B.2.2	NORD-100 Asynchronous V24 Programming Specifications	B—12
B.3	Specification of Line Printer Interface for CDC 9380 for NORD-10/100	B—18
B.4	NORD-100 Disk Programming Specifications	B—20
C	SWITCH SETTINGS FOR THE DIFFERENT NORD-100 MODULES	C—1
C.1	Switches on the CPU Module	C—1
C.1.1	ALD — Automatic Load Descriptor	C—2
C.1.2	Console: Speed setting for console terminal	C—2
C.2	Switches on Floppy and 4 Terminals Module (3010)	C—3
C.2.1	1 Floppy Disk System	C—3
C.2.2	2 Terminal Group	C—4
C.2.3	3 Initial Baud Rate for Terminals	C—4
C.3	Switches on Memory Modules (3005)	C—6
C.4	Switches on the 10MB Disk Module (3004)	C—7
C.5	Switch Setting on the Pertec Magnetic Tape Module (3006)	C—8
C.6	Switch Setting on NORD-100 Bus Adapter (3008)	C—8
C.7	Switch Setting on Local I/O Bus	C—9
D	NORD-100 PLUG PANEL FOR EXTERNAL DEVICE CONNECTION	D—1
E	ORGANIZATION OF NORD-100 MODULES	E—1
E.1	Location of Integrated Circuits (Card Coordinates)	E—1
E.2	Arrangement Drawings Examples	E—3
E.3	Notation of Signals to and from NORD-100 Modules	E—8
E.4	Cable List for External Device Connection	E—9

Appendix:		*Page:*
F	NORD-100 BUS BACKPLANE SIGNALS	F—1
F.1	Representation of Signals in Timing Diagrams	F—7
G	SCHEMATICS	G—1
G.1	NORD-100 CPU	G—2
G.2	4 Terminals and Floppy Disk	G—7
G.3	8 Terminals	G—12
G.4	10MB Disk Controller	G—17
G.5	Dynamic Ram	G—19

I *NORD-100 INPUT/OUTPUT SYSTEM OVERVIEW*

I.1 INTRODUCTION

The purpose of the input/output system (I/O system) is to perform the physical communication between connected peripheral equipment and the NORD-100 computer system.

The user normally does not interact directly with the I/O system, only indirectly via the operating system.

Thus, all I/O operations are normally invisable to the user.

However, privileged users may access the I/O system directly and users with special real-time requirements (running direct tasks) may bypass the I/O system for direct access to specific devices.

I.1.1 *DEFINITION OF TERMS*

As shown in Figure I.1, the I/O system is part hardware and part software.

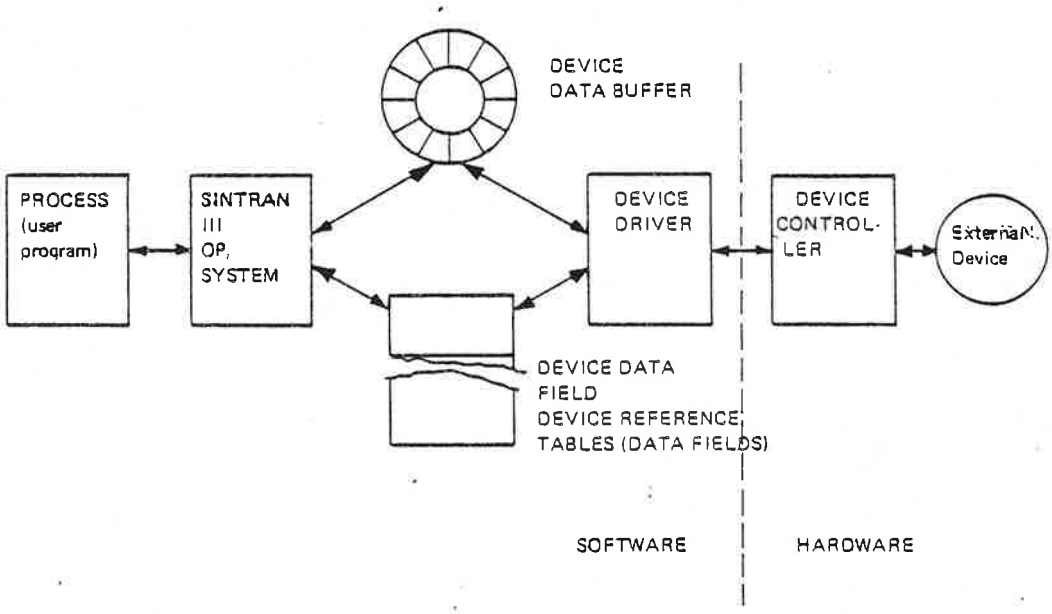


Figure I.1: Fundamental Elements of the I/O System

Figure I.1 shows some of the important elements of the I/O system. The picture is not complete, but it illustrates the chain of linkages that are basic to the I/O system.

A *device controller*, as shown in Figure I.1, is the hardware I/O interface. The functions of the device controllers is to:

- synchronize the speed between a peripheral and the NORD-100
- formating of data to and from the peripheral

Depending on the particular controllers, the device controller may drive only one peripheral (such as a terminal) or may be capable of driving several peripherals (such as disk units).

For each *device controller* there is an entry in a *device data field* table. A *device data field* defines a device and is used by a *device driver*, the program that accesses and controls a peripheral.

For each *device* there is a *device data buffer* where information between the operating system and the I/O system is exchanged.

This manual is intended to give a functional description of the hardware involved in the I/O system.

I.2 NORD-100 ARCHITECTURE

I.2.1 *NORD-100 BUS STRUCTURE — THE NORD-100 BUS*

In the NORD-100 computer system all hardware modules are connected together via a common bus, the NORD-100 bus (see Figure I.2.1).

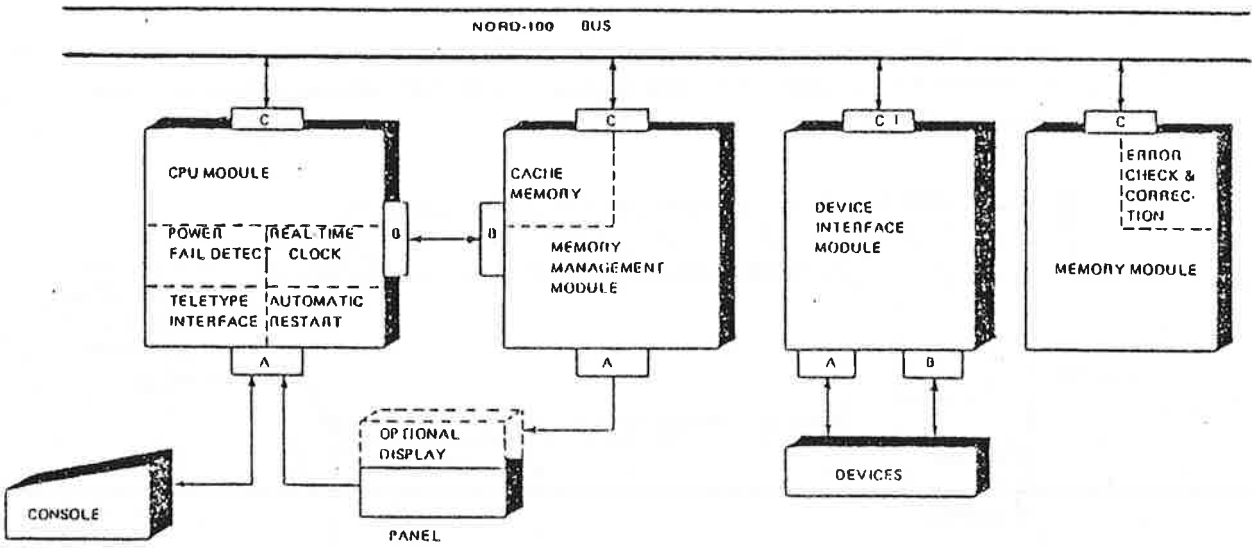


Figure I.2.1: NORD-100 Bus Structure

All communication between NORD-100 modules is provided by this bus, except CPU, MMS and Cache communication. Therefore, the NORD-100 bus is general purpose, to allow all classes of NORD-100 modules simply to be "plugged" into the system.

I.2.1.1 *Physical Arrangement of the NORD-100 Bus and Card Crate*

Physically, the NORD-100 bus is available as a printed backplane. The backplane is available in two versions containing either 12 or 21 positions for module connection. Besides the different numbers of positions, there is no difference between the two versions as far as the connection of hardware modules is concerned.

However, the difference between a 12 or 21 position system is easily visualized by the organization of the card crates and the size of the cabinets.

In the 12 position version, the required power is supplied by a power supply located within the card crate (refer to Figure I.2.2). This approach leads to a very compact system.

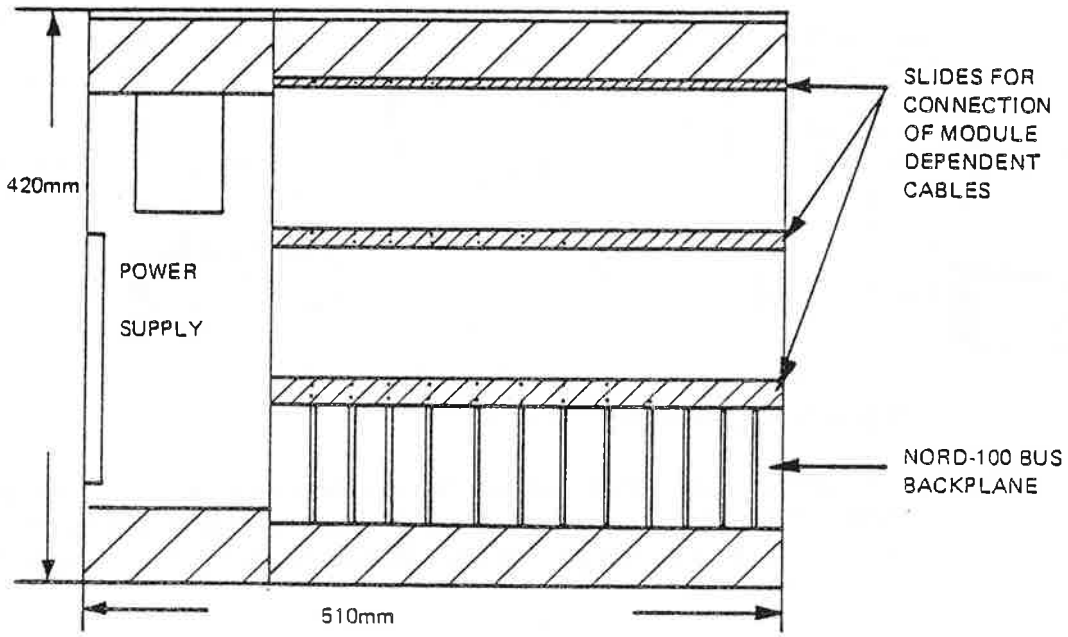


Figure 1.2.2: 12 Position NORD-100 Crate Layout (Top View)

In the 21 position version, the power supply is removed from the card crate and located in the upper part of the cabinet (refer to Figures I.2.3 and I.2.4). Thus, the cabinet must be bigger that the cabinet for a 12 position crate.

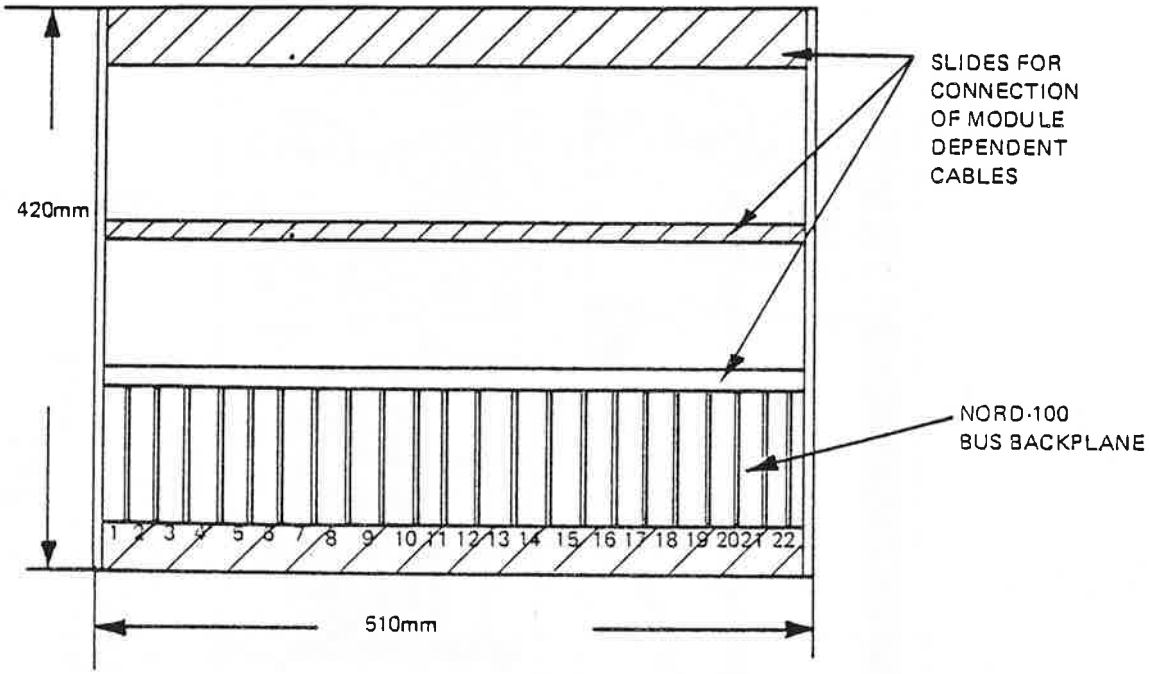


Figure I.2.3: 21 Position NORD-100 Crate Layout (Front View)

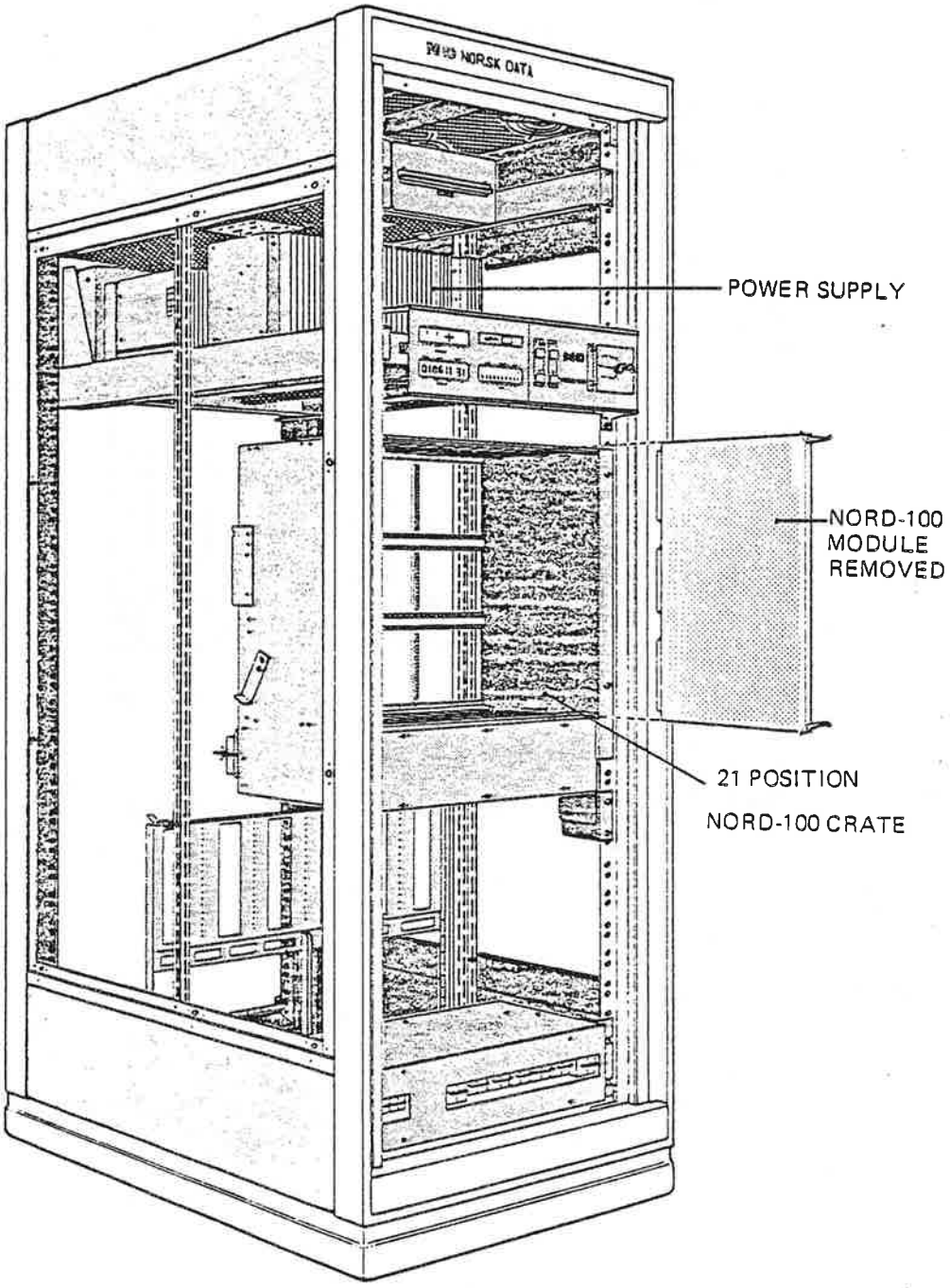


Figure I.2.4: Cabinet Layout for a 21 Position NORD-100 Backplane System.

The NORD-100 bus backplane of either 12 or 21 positions is, therefore, firstly selected from the number of modules required in an actual system.

On the other hand, physical dimensions in the cabinet and possible needs for future extension could also be taken into account.

In some cases, a configuration may require space for even more than 21 modules. In such cases, NORD-100 card crates are linked together by Bus Expander modules. The number of crates to be linked is set to a theoretical maximum of 8 crates.

The properties of the Bus Expander module are explained in Section VI.

Each NORD-100 bus backplane position contains a total of 96 lines (power and ground lines included).

All positions in the backplane contain the same information, i.e., they are equal.

This allows flexible configuration or reconfiguration of hardware.

The actual placement of different modules follow various rules which are described later.

I.2.2 *FUNCTIONS OF THE NORD-100 BUS*

I.2.2.1 *General*

As already mentioned, the NORD-100 bus serves as a general purpose highway for data and addresses in the NORD-100 computer system.

The actual function and use of the bus depends on what kind of transfer is being performed.

I.2.2.2 *NORD-100 Bus and the I/O System*

I.2.2.2.1 General

As illustrated in Figure I.2.1, NORD-100 I/O device controllers are connected directly to the NORD-100 bus. This includes some important advantages:

- The I/O device controllers are connected to the same printed backplane as the CPU and the memory system;
  - no external wiring
  - increased realiability
- only one bus to connect between source and destination of a transfer makes a fast system.
- I/O device controllers and slot number in the NORD-100 bus is *not* predefined:
  - easy to expand
  - easy to reconfigure

I.2.2.2.2 I/O System Information Exchange and NORD-100 Bus Usage

External devices may be classified as:

1. Slow character/word oriented devices (for example, terminals)
2. High speed block oriented mass storaged devices (for example, disks, mag. tape)

Data exchange between these classes of peripherals and NORD-100 falls into one of two categories.

Programmed Input/Output (PIO)

The first class is completely controlled by program. This is called Programmed Input/Output (PIO).

In programmed data transfers, each word or byte to be exchanged has to be programmed between the selected I/O device controller (interface) and the CPU A register.

This is illustrated in Figure I.2.3.

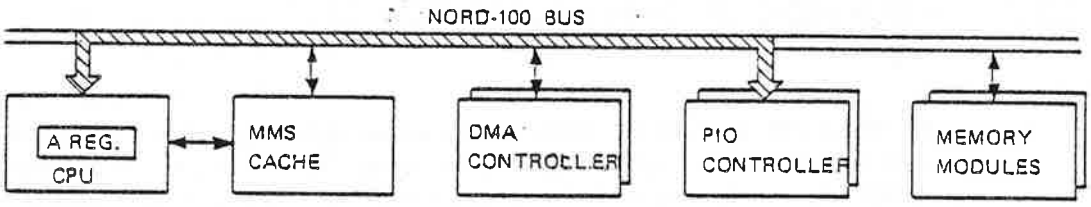


Figure I.2.3: PIO Data Exchange and Bus Usage

The CPU is busy for every word/byte to exchange. The NORD-100 bus is allocated to carry data being exchanged.

Direct Memory Access (DMA)

High speed block oriented peripherals exchange data directly with the NORD-100 memory system. This is called Direct Memory Access (DMA).

The I/O interface which controls the transfer, is activated by a driver program in SINTRAN III. The activation includes telling the I/O interface where to place/find data in memory, where to find/place data on the external device and the number of words to exchange.

The number of words specified is then transferred automatically without program (CPU) interaction.

DMA data exchange and bus usage is illustrated in Figure I.2.4.

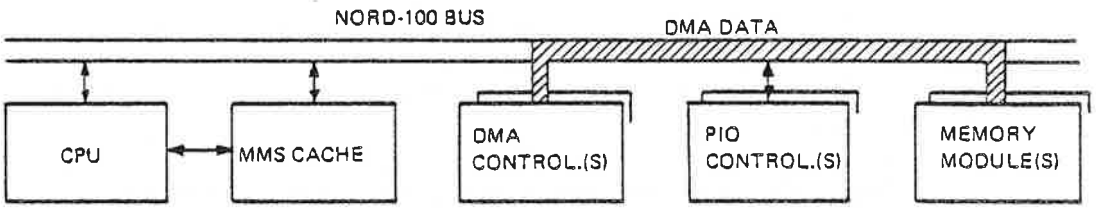


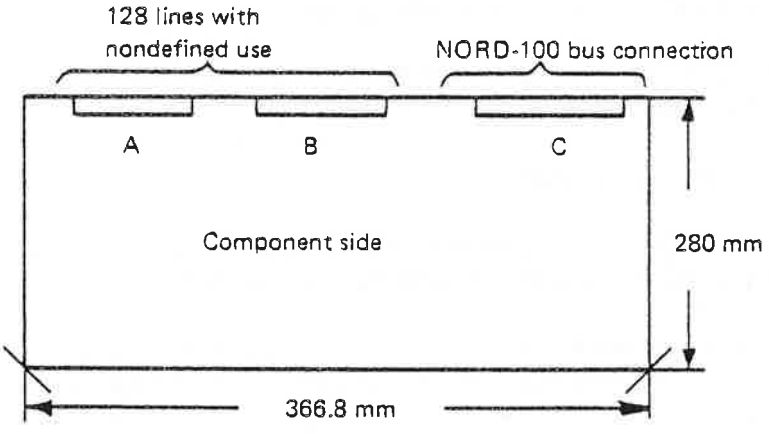
Figure I.2.4: DMA Data Exchange and Bus Usage

The CPU may run any activity not waiting for the DMA transfer to be completed.

The NORD-100 bus is allocated for a DMA transfer when a word is ready to be exchanged. The connection between a DMA controller and the memory system is referred to as a "DMA channel". More than one DMA controller (interface) may be activated at the same time, thus sharing the DMA channels total band width.

I.2.3 *ORGANIZATION OF NORD-100 MODULES*

All NORD-100 modules are made to a common standard. Every module has at least one connector used for connection to the NORD-100 bus. In addition, a NORD-100 module may have one or two extra connectors carrying a total of 128 lines (64 lines in each connector).



The plugs are assigned an identification letter as illustrated. The plug used for connection to the NORD-100 bus is assigned the letter C. This plug contains 96 pins; each of them defined in accordance to the NORD-100 backplane (bus) standard.

The plugs A and B each carry 64 lines with nondefined use. In the design of I/O device controllers these plugs are used for connection to the external devices.

In Figure I.2.5 the organization of an I/O device controller is shown.

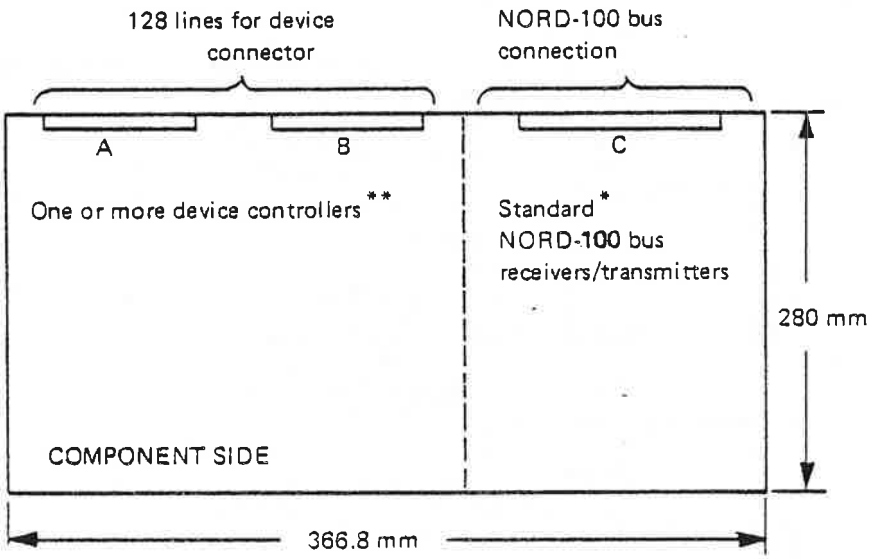


Figure I.2.5: Organization of an I/O Device Controller Module

* The standard part includes bus handshake control logic. This part is standarized for:

- all PIO device controllers
- all medium speed DMA controllers (10 Mb disk, mag. tape)

** The device dependent part may handle up to four different PIO devices or one DMA controller. A DMA controller may handle up to four units.

Example — one module:

- four terminals and floppy disk
- 8 terminals
- one 10Mb disk controller (up to 4 units)
- one mag. tape controller (up to 4 units)

I.2.4 *NORD-100 CONFIGURATION EXAMPLES*

Figure I.2.6 shows a typical medium sized NORD-100 single processor system.

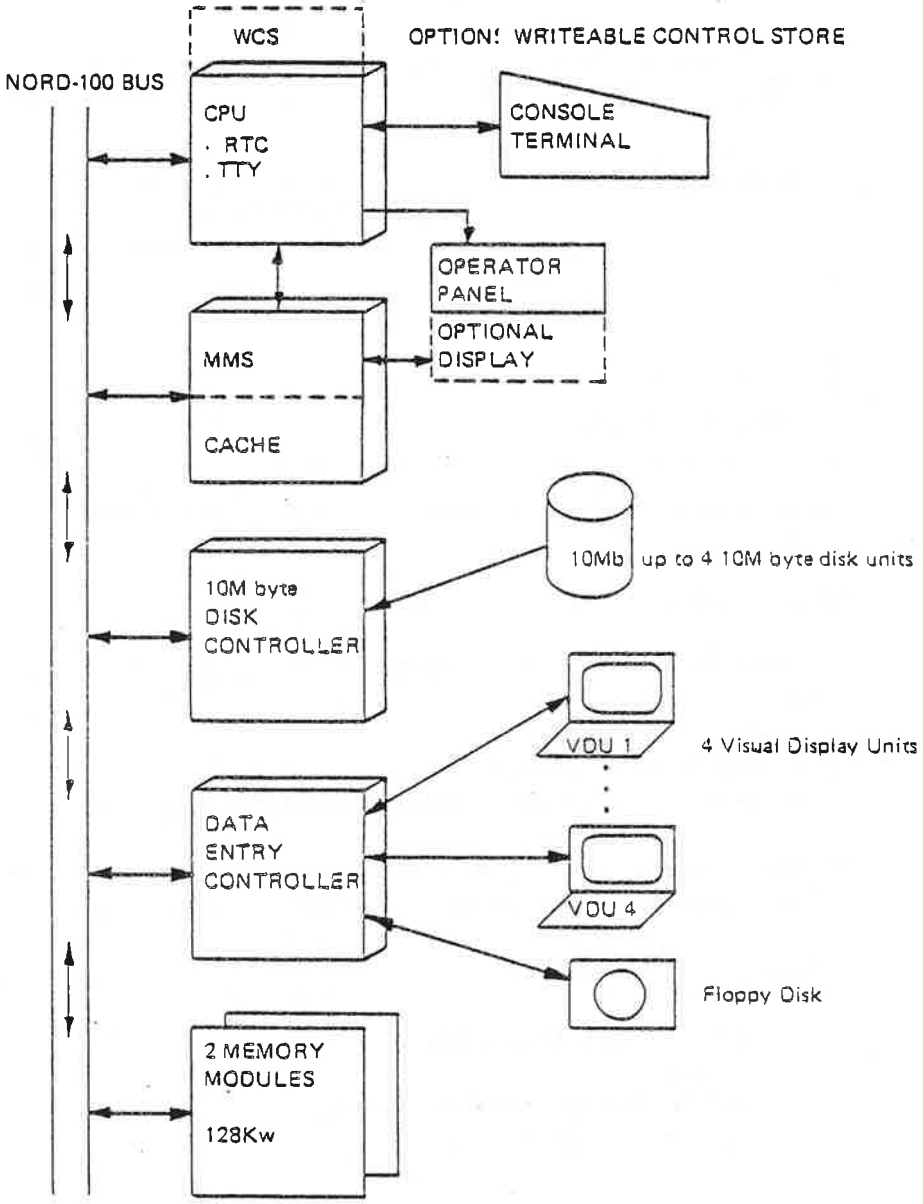


Figure I.2.6: NORD-100 Single Processor System

Physical layout in the card crate of the configuration shown in Figure I.2.6 is given in Figure I.2.7.

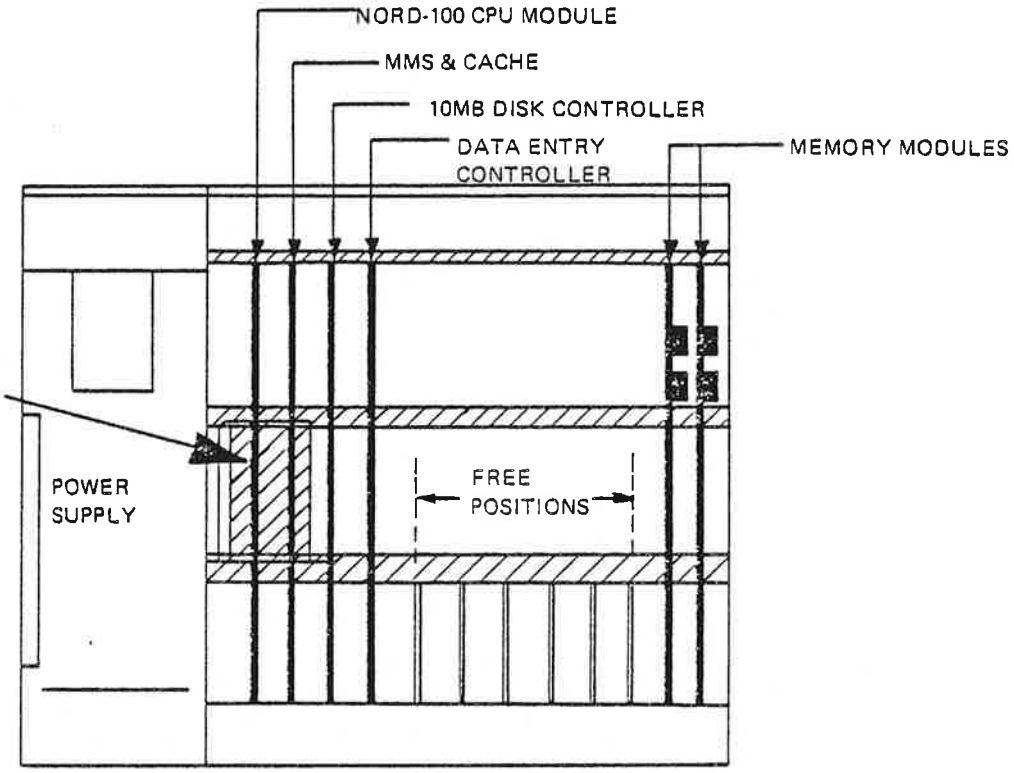


Figure I.2.7: Configuration Example — Physical Layout in Card Crate

I.3 PROGRAMMING OF I/O DEVICE CONTROLLERS (INTERFACES) – THE INPUT/OUTPUT INSTRUCTIONS IOX/IOXT

I.3.1 *GENERAL*

Neither a PIO interface nor a DMA controller starts an I/O transfer of their own. They have to be activated. The activation is done by the relevant device driver program when a transfer is requested (either from user program or from an I/O device controller through a hardware interrupt).

The control of an I/O device interface includes programmed access to physical hardware registers on the interfaces. Two special machine instructions are implemented for this purpose.

I.3.2 *INTRODUCTION TO IOX/IOXT*

In the NORD-100 instruction set there are two instructions usable for information exchange between the hardware device controllers and the CPU.

The mnemonics of the instructions are IOX and IOXT recognized by the MAC assembler.

NOTE: IOX and IOXT are privileged instructions. That is, under a running system (SINTRAN III running) only privileged users may use IOX/IOXT. This is normally the device driver programs belonging to SINTRAN III itself.

Nonprivileged use is detected and error message given.

If SINTRAN III is not running and paging is off, IOX and IOXT is available as other nonprivileged instructions.

Programs containing IOX/IOXT may then be entered in executable format either via the Microprogramed Operators Communication (MOPC) or from a mass storage device as floppy disk.

1.3.3 *FORMAT AND FUNCTIONS OF THE IOX AND IOXT INSTRUCTIONS*

In the NORD-100 instruction set, IOX and IOXT are the only instructions useable in information exchange between CPU and I/O device controllers.

Neither the IOX nor the IOXT instruction perform any predefined functions on the I/O device controllers.

Their only purpose is to carry information between the CPU A register and a specified I/O device register.

The actual function of the IOX/IOXT instructions *depend on the selected I/O device register.*

All I/O device controllers are assigned a group of registers, each of them having a special meaning on the interfaces. Therefore, the programmer, by specifying an appropriate register on a device, assigns the exchanged information a special function.

Thus, all classes of information may be exchanged:

- data (byte or word) to or from PIO interface data registers (not DMA interfaces)
- transfer control information to PIO and DMA interfaces control registers
- transfer status information from PIO and DMA interface status registers.

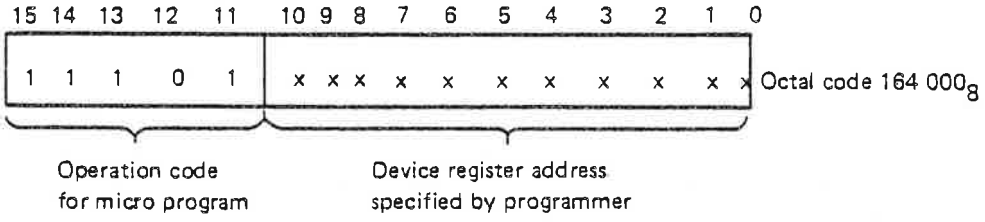
Instruction Formats

All I/O device registers are assigned an address referred to as "device register address". That is, both the IOX and IOXT instructions access an I/O device register by its address.

IOX Instruction Format

In the IOX instruction, the address of the I/O device register to access is specified in the 11 lower bits of the instruction itself.

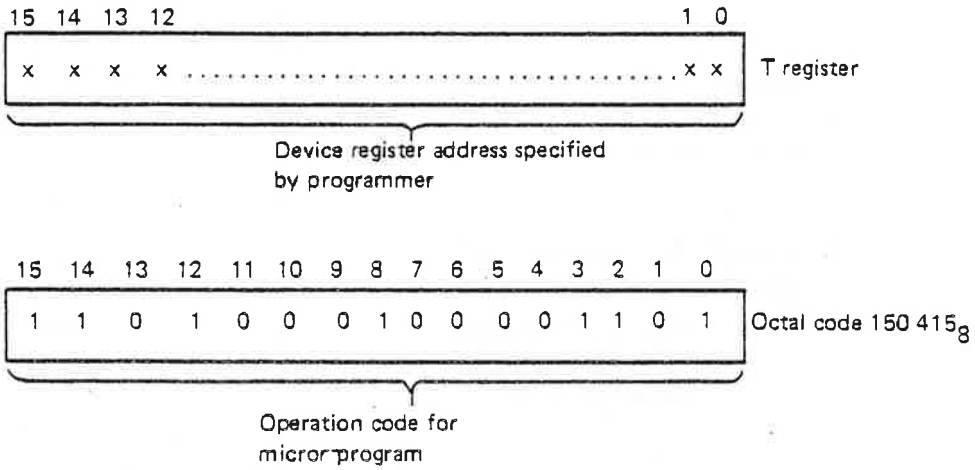
IOX <device register address>



IOXT Instruction Format

In the IOXT instruction, the device register address should be loaded into the T register prior to executing IOXT.

```
LDT <device register address>
IOXT
```



I.3.3.1 *Definition of IOX/IOXT Transfer Direction*

The IOX and IOXT instructions handle both input and output transfers. An input transfer in this context means input to the CPU A register from a specified I/O device register. An output transfer means output from the CPU A register to a specified I/O device register.

The actual transfer direction of the IOX and IOXT instructions is decoded from the device register address based on the following convention:

- The transfer direction is *input* if the device register address is *even*. That is, bit 0 of the address is "0".
- The transfer direction is *output* if the device register is *odd*. That is, bit 0 of the address is "1".

The programmer is not affected by this convention. All I/O device registers which should be loaded from the CPU A register (output transfer) are assigned an odd device register address.

All I/O device registers which should be read into the CPU A register (input transfer) are assigned an even device register address.

How hardware uses bit 0 in the device register address to select the transfer direction is described in later sections.

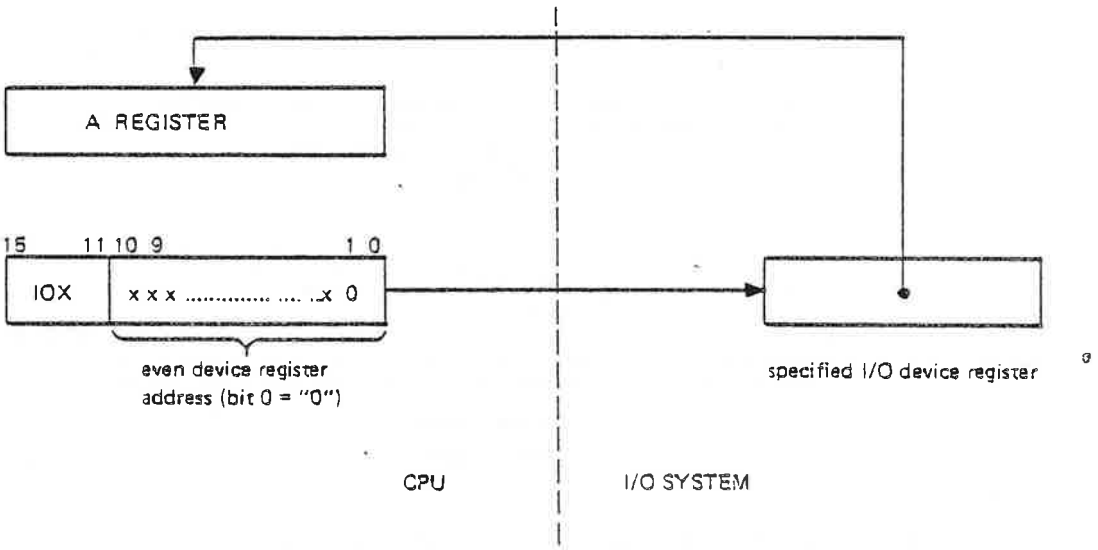
IOX/IOXT Transfer Direction Illustrations

Input Transfer:

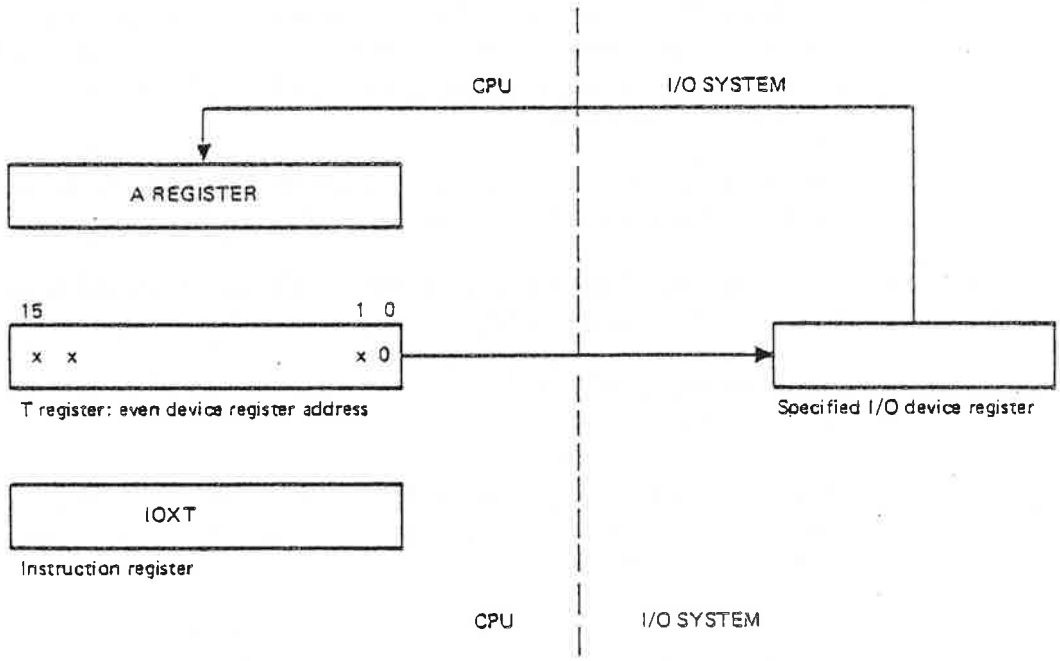
An I/O device register specified with an *even* device register address is loaded into the A register.

Illustration:

IOX Input:



IOXT Input:

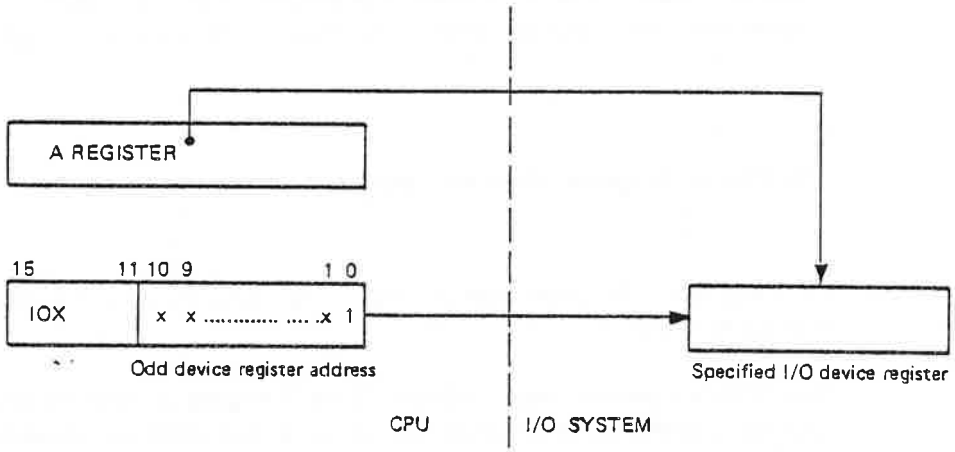


Output Transfer:

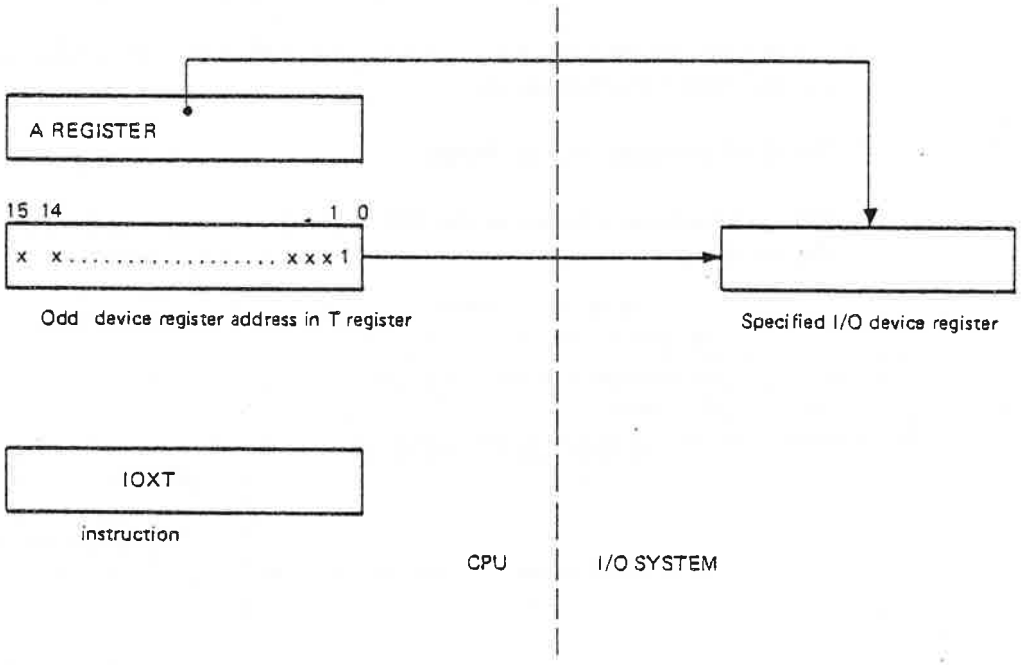
The CPU A register is written to an I/O device register specified by an *odd* device register address.

Illustrations:

IOX Output:



IOXT Output:



As indicated, the IOX and IOXT instructions are general purpose I/O instructions.

The actual register on a device to access is specified in "device register address". The information to exchange and the transfer direction (input or output) is given implicitly through the selected device register address.

I.3.4 *CALCULATION OF THE IOX/IOXT DEVICE REGISTER ADDRESS*

In order to form the device register address, a basic understanding of the organization of the device registers is needed.

Logically, the collection of all I/O device registers may be thought of as a contiguous register file.

Physically, each I/O device controller is assigned a group of registers. Thus, the registers are located and accessed on the modules where they are relevant.

I.3.4.1 *The Device Register Address Range*

By using the IOX instruction a total of 2K registers may be specified, i.e., addresses from 0 - 3777_8.

The IOXT instructions, which use the 16 bits T register to hold the device register address, may hence theoretically address up to 64K registers (addresses from 0 - 177777_8).

However, the collection of all device registers implemented on interfaces designed at Norsk Data only cover the address area 0 - 1777_8, i.e., 1K register.

The remaining addresses, which may be specified by the IOX or IOXT instructions, are defined as described below.

The IOX Instruction Address Range

The address range covered by the IOX instruction is organized as illustrated below (Figure I.3.1).

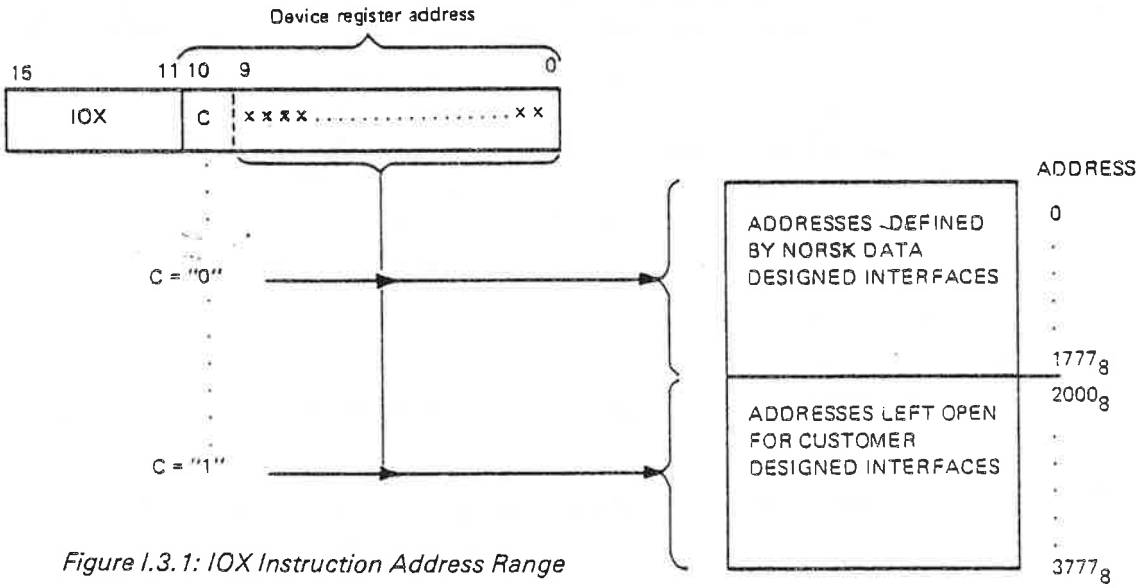


Figure I.3.1: IOX Instruction Address Range

As indicated, bit 10 in the IOX device register address, equal to "0", defines an address defined by Norsk Data's equipment. If bit 10 is "1", the specified address has to be defined by some customer designed equipment.

The IOXT Device Register Address Range

As already mentioned, the T register should be initiated with a relevant device register address prior to the exeuction of IOXT.

The address range covered by the IOXT instruction is organized as shown below (Figure I.3.2).

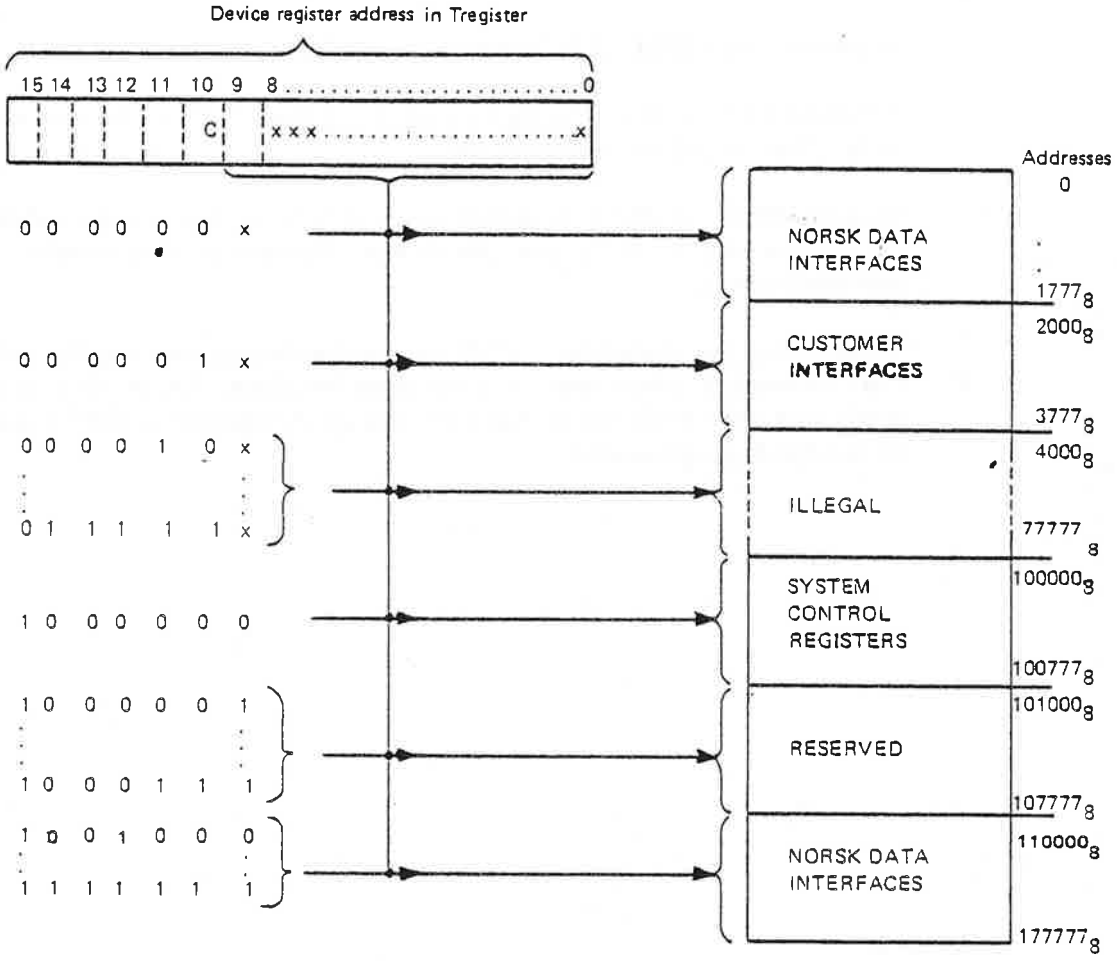


Figure I.3.2: Instruction IOXT Address Range

As indicated, if the bits 11-15 are all zero, the device register address of the IOXT instruction overlap in the IOX instruction's address range.

If one of the bits 11-14 is "1" and bit 15 is "0", the address is illegal.

If bit 15 is set ("1"), registers not accessible by the IOX instruction may be specified.

Addresses from 100000_8 - 100777_8 are used to specify system control registers which have to be accessed via the NORD-100 bus. An example is the Error Correction Control Register (ECCR), physically located on the memory modules.

ECCR is loaded by the TRR instruction. However, since ECCR is accessed via the NORD-100 bus, the micro program converts the TRR instruction to an IOXT instruction.

The "device register address" is generated by the micro program in the area 100000_8 - 100777_8 (TRR ECCR is converted to IOXT 100115_8).

The registers in this address area is not relevant to the I/O system and are therefore not discussed any further in this manual.

Addresses from 101000_8 - 107777_8 are reserved by Norsk Data for future needs.

Addresses from 110000_8 - 177777_8 are reserved by Norsk Data for future extention of the I/O device register address range.

Since all present I/O device controllers designed at Norsk Data may be specified in the address area 0 - 1777_8, only this area is supposed to be accessible in the following sections.

Further, since the addresses 0 - 1777_8 may be specified by both the IOX and the IOXT instructions which both have the same functions, IOX is the instruction which most often is referred to. However, convertion from IOX to IOXT is given in some programming examples.

I.3.4.2 *Specification of an I/O Device Register Address for Norsk Data Designed Interfaces*

As already mentioned, each I/O device controller is assigned a group of registers with concecutive device register addresses. The total number of registers assigned one I/O interface may be from 4 to 16 depending on the control functions needed on a device.

Therefore, the device register address may be divided up into two parts:

— device selection address (hardware device number — dev. no.)

— register selection address to select a register within the selected device (register number - reg. no. — in selected device)

The IOX/IOXT device register address is then calculated by the formula:

<dev. reg. addr.> = dev. no. + reg. no.

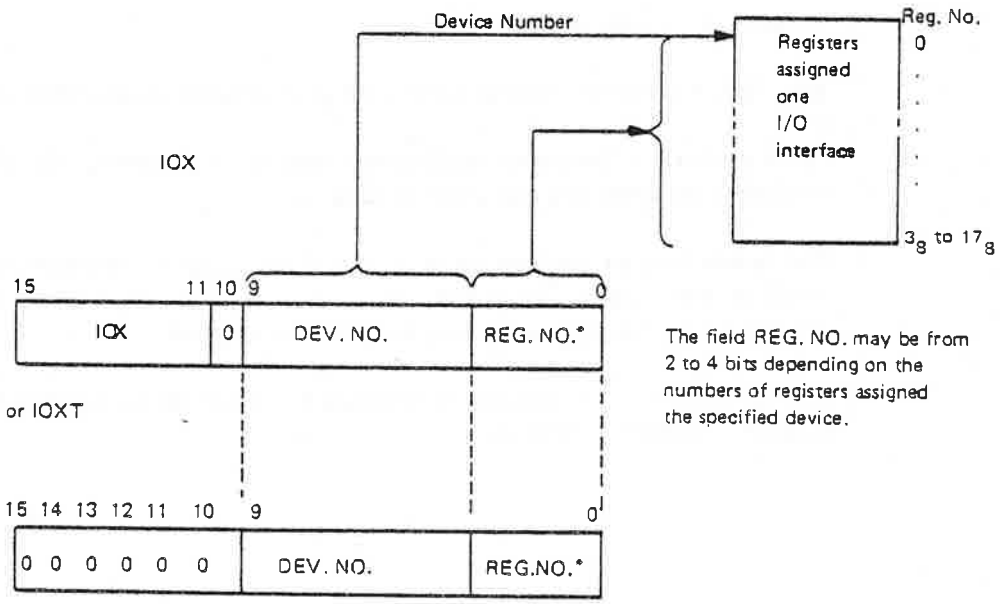


Figure I.3.3: Device Register Address Calculation — Illustration

For Norsk Data produced device controllers, both the device number and the register number have been standarized. The hardware device numbers (dev. no.) are available in a table. The numbers assigned the various registers on an I/O interface are given in the specifications following each I/O interface.

How to use these tools in calculation of the device register address is given in the following sections.

I.3.4.3 *Device Selection — the Hardware Device Number (Dev. No.)*

As indicated in Figure I.3.3.

The start (least) address of the block of registers on a device is referred to as the device's hardware device number (dev. no.).

The device number for Norsk Data produced interfaces are standarized and put in table form (in Appendix A). In Appendix A, the device register address range for the most common I/O interfaces is included.

NORD-100 Standard Device Numbers (Appendix A):

Device Register Address range	Device
.	.
.	.
.	.
300_8 - 307_8	Terminal 1

↑
device number (dev. no. = 300_8)

300_8 - 307_8 is the device register address range (8 registers are assigned terminal 1).
A
There is always a unique correspondence between a peripheral, the I/O interface controlling the peripheral and a device number.

The device number corresponding to an I/O interface is selectable by a thumb-wheel on the module. This is done to allow equal hardware modules to cover all possible device numbers assigned one class of peripherals.

Note that there is no relationship between a module device number and its slot number in the NORD-100 bus.

I.3.4.4 *Device Register (reg. no.) Selection on an I/O Interface*

Each register implemented on an I/O interface is assigned a unique number in the interface (register number — reg. no.).

Each register is related to a special function on the interfaces. That is, when both the device number and a register number is specified, the information exchanged by IOX/IOXT has a defined meaning, given by the funciton of the selected device register.

Each I/O device controller is described in a specification referred to as "the programming specification". In the programming specification of an interface, the functions and the associated numbers of each register is described. (Programming specifications of some I/O interfaces is given in Appendix B.)

In order to understand the programming speicfications of an interface, the organization and register assignment of Norsk Data produced interfaces is needed.

I/O interface channels and registers:

An I/O interface is said to have two channels if it can handle both input and output transfers simultaneously. A one channel interface may handle either input or output transfers.

Examples:

One channel devices:

- line printer interface (output only)
- card reader interface (input only)

Two channel devices:

- terminal interface (output to terminal and input from keyboard)
- communication controllers (output to line and input from line)

Bidirectional (one channel) Device Controllers:

- disk interfaces (may handle both input and ouput but not simultaneously)
- mag. tape interface (same as for disk)

PIO INTERFACE REGISTERS ACCESSIBLE BY IOX/IOXT:

On a PIO interface designed at Norsk Data, each channel is assigned at least three registers:

- a control register
- a status register
- a data register

The Control Register

The control register is a "write only" register (IOX/IOXT output). Commands (start/stop transfer, mode of operation) from a device driver program to an I/O interface channel is given through this register.

```
LDA <command>      % initiate A register
IOX <dev. reg. addr.> % write content of A reg. to specified control reg.
```

The Status Register

The status register is a "read only" register (IOX/IOXT input). By reading this register, the status of an I/O interface channel (ready for transfer, busy, errors, etc.) may be investigated.

```
LDT <dev. reg. addr.> % initiate T reg. with dev. reg. addr. of the status reg. to
                      % access
IOXT                  % specified status reg. — A reg.
```

The Data Register

The data registers is "write only" if it belongs to an output channel or "read only" if it belongs to an input channel.

Output Data:

```
LDA <data>          % initiate A reg. with data to output
IOX <dev. reg. addr.> % dev. reg. addr. of data reg. to access
```

Input Data:

```
IOX <dev. reg. addr.> % dev. reg. addr. of data reg. to access
                      % i.e., data → A register
```

EXAMPLES OF DEVICE REGISTER ADDRESS CALCULATION FOR PIO DEVICES

Example 1: Line Printer Interface

The line printer interface has only one channel — the output channel.

The registers assigned the interface follows Norsk Data's standard and is defined in the line printer programming specification (Appendix B).

Register:	Register No.:
Output channel control register	3
Output channel status register	2
Output channel data register	1
Input data register for*	0

* "off line" loop back of data (for test purposes only)

Problem 1:

What is the IOX/IOXT device register address of the data register on line printer no. 1.

Solution:

The dev. reg. addr. is calculated form:

dev. reg. addr = dev. no. + reg. no.

The device number (dev. no.) should select line printer no. 1. In Appendix A that is found to be 430_8.

The register number should point out the data register on the line printer interface. The register number of the data register is found in the programming specifications of line printer interface (Appendix B) and is equal to 1.

That is:

dev. reg. addr. = dev. no. + reg. no.
= $430_8 + 1_8 = 431_8$

Assumed that line printer no. 1 is ready to accept data, the following program will write the location CHAR to the line printer data register.

```
        LDA  CHAR       % initiate A reg. with char. to output
        IOX  431        % write content of A reg. (CHAR) to
                        % line printer no. 1 data reg.
CHAR,   101             % 101 is ASCII value of A
        or by using IOXT;
```

```
        LDT  DEVNO      % initiate T register with dev. reg. addr.
        LDA  CHAR       % initiate A reg. with char. to output
        IOXT            % write content of A reg. (CHAR) to
                          line printer no. 1 data reg.

CHAR,   101
DEVNO,  431
```

Problem 2:

Write 7_8 to the control register of line printer no. 2.

Solution:

Dev. reg. addr. = Dev. No. (L-P no. 2) + Reg. No.
$= 434_8 + 3_8$
$= 437_8$

Program:

```
        SAA  7          % A: = 7₈
        IOX  437        % A reg. → dev. reg. 437₈

        or

        LDT  ADDR       % initiate T reg. with <dev. reg.
                        % addr.>
        SAA  7          % A: = 7₈
        IOXT            % A reg. → level reg. 437 ₈
        .
        .
        .
ADDR,   437
```

Example 2: Terminal Interface

A terminal interface has two channels – input from keyboard and output to terminal display. Each channel is controlled independently and contains its own set of control, status and data registers.

Input channel:

Register:	Register No.:
Input control register*	3
Input status register**	2
Input data register	0

Output channel:

Register:	Register No.:
Output control register*	7
Output status register**	6
Output data register	5

Common Register:	Register No.:
Speed selection (baud rate) register	1

* Note that the control register is always a write only register independent of which channel it is assigned

** Note that the status register is always a read only register independent of input/output channel

Problem 1:

What is the IOX/IOXT device register address for the output channel status register on terminal no. 1.

Solution:

From Appendix A, terminal no. 1 is found to have device number (dev. no.) equal to 300_8.

In Appendix B, programming specifications for a terminal interface, the register number (reg.no.) for output channel status register is 6_8.

Using the formula

Dev. reg. addr. = dev. no. + reg. no.
$= 300_8 + 6_8 = 306_8$

That is, by executing,

```
IOX  306
```

or

```
LDT  (306
IOXT
```

the status register for the output channel on terminal 1 is read into the CPU A register.

Problem 2:

Assume that terminal 1 have a data character available to be read and that terminal 2 is ready to accept a character.

Read the character on terminal 1, store it in a location called BUFF and write it back to terminal 2.

Solution:

The data character available on terminal 1 is located in the terminals input channel data register. The device register address of this register is (using Appendix A and B).

Dev. reg. addr. = dev. no. (terminal 1) + reg. no. (input data)
$= 300_8 + 0_8 = 300_8$

The character stored in BUFF should be loaded to the output channel data register on terminal 2. The device register address of this register is (using Appendix A and B).

Dev. reg. addr. = Dev. no. (terminal 2) + reg. no. (output data)
$= 310_8 + 5_8 = 315_8$

Correct Program Using IOX (Convert to IOXT privately):

```
INPUT,    IOX  300        % read input data form terminal 1
          STA  BUFF       % store A reg. in BUFF

OUTPUT,   LDA  BUFF       % initiate A reg.
          IOX  315        % write data to terminal 2
BUFF,     0
```

DMA INTERFACE REGISTERS ACCESSIBLE BY IOX/IOXT:

A DMA controller usually contains from 8 to 16 registers accessible by IOX/IOXT instrucitons.

The IOX/IOXT device register address is just as for PIO interfaces, calculated for device number and register number found in the device number table (Appendix A) and the relevant programming specification respectively.

Discussion of different registers on a DMA controller is left for later sections (programming of DMA controllers). However, the important difference from PIO interfaces is noted:

— DMA controllers have no data register accessible by IOX/IOXT. The data register is exchanged directly between controller and memory without program interaction.

I.3.5 *FORMAT OF THE CONTROL AND STATUS REGISTERS FOR NORSK DATA DESIGNED PIO AND DMA INTERFACES*

I.3.5.1 *General*

All information exchange, i.e., control, status and data, between PIO devices and NORD-100 CPU is programmed via the A register by means of IOX/IOXT instructions.

On DMA controllers, control and status are programmed while data is exchanged directly to memory.

Before programming, the format and functions of the different I/O interface registers must be known. This information is found in the programming specifications for an interface. (Some examples are given in Appendix B.)

The format of the control and status registers are device dependent and may be assigned by the designer of each device controller. However, on Norsk Data designed interfaces, both the functions and the formats of these registers have been standarized.

The format of the data register follows the format accepted by the external device (for example, ASCII to and from terminals).

I.3.5.2 *Format and Functions of the Status Register*

The information available in an I/O interface channels status register is set by the interface itself.

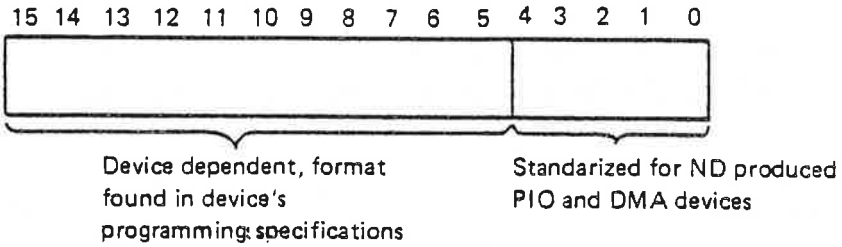
The register is read into CPU A register (IOX read status register) when the state of an interface channel is to be investigated.

The status register is read by

```
IOX <dev. reg. addr.>        % dev. reg. addr. of status reg. to access
```

The format of the status register has been standarized for all Norsk Data produced interfaces and is equal both for the input and output channel of a device.

Format of the status register (format of the A register after IOX read status register).



Status Register:

Bit 0	Ready for transfer, interrupt enabled
Bit 1	Error interrupt enabled
Bit 2	Device active
Bit 3	Device ready for transfer
Bit 4	Inclusive OR of errors
Bit 5	Error indicator
Bit 6	Error indicator
Bit 7	Error indicator
Bit 8	Error indicator
Bit 9	Selected unit
Bit 10	Selected unit
Bit 11	Operational mode of device
Bit 12	Operational mode of device
Bit 13	Operational mode of device
Bit 14	Operational mode of device
Bit 15	Operational mode of device

Bit 0 - 2:

The status register bits 0 - 2 are direct feedback of the corresponding bits in the control register of the same I/O interface channel (see control register format).

Bit 3 — Device Ready for Transfer

Status register bit 3 set or not set tells whether an I/O interface channel is ready to operate or not. What is meant by "ready for transfer" (bit 3 "1") differs from the input to the output channel.

PIO Device Input Channel:

Bit 3 equal to 1:

Input data register contains valid information ready to be read.

Bit 3 equal to 0:
Input data register does not contain valid information

PIO Device Output Channel:

Bit 3 equal to 1:

Output data register is empty and may accept the next output data character.

Bit 3 equal to 0:

Output data register is *not* empty, i.e., the data register should *not* be loaded.

DMA Controllers:

Bit 3 equal to "1":

DMA transfer completed.

Bit 3 equal to "0":

DMA transfer is going on.

Bit 4 — Inclusive OR of Errors

Bit 4 set to one: an error has occurred in the I/O interface channel investigated. More about the error is given in status register bits 5-15 (actual bits are described in the programming specifications).

Bit 5 - 15 — Nondefined

Status register bits 5 - 15 are assigned by the I/O interface designer and given in the I/O interface's programming specifications.

I.3.5.3 *Format and Function of the Control Register*

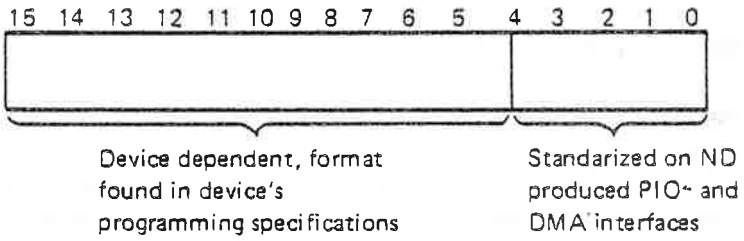
The information written into the control register of an I/O interface channel is always taken by the interface as a command. The command may be start/stop of transfer, mode of operation, etc.

The control register of an I/O interface channel is loaded by:

```
LDA <command>          % initiate A register
IOX <dev. reg. addr.>  % dev. reg. addr. of I/O interface
                         control reg.
```

The format of the control register has been standarized for all Norsk Data produced interfaces and is equal both for the input and output channel of a device.

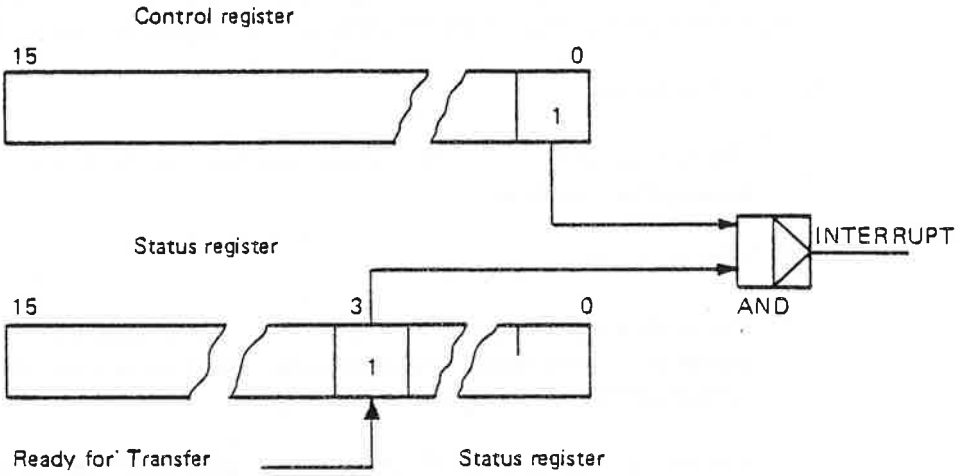
Format of the Control Register (Format of A Register before IOX Load Control Register)



Bit 0	Enable interrupt on device ready for transfer
Bit 1	Enable interrupt on errors
Bit 2	Activate device
Bit 3	Test mode
Bit 4	Device clear
Bit 5	Not used
Bit 6	Not used
Bit 7	Not assigned
Bit 8	Not assigned
Bit 9	Unit Multiple unit control interfaces
Bit 10	Unit (floppy disk)
Bit 11	Device operation
Bit 12	Device operation
Bit 13	Device operation
Bit 14	Device operation
Bit 15	Device operation

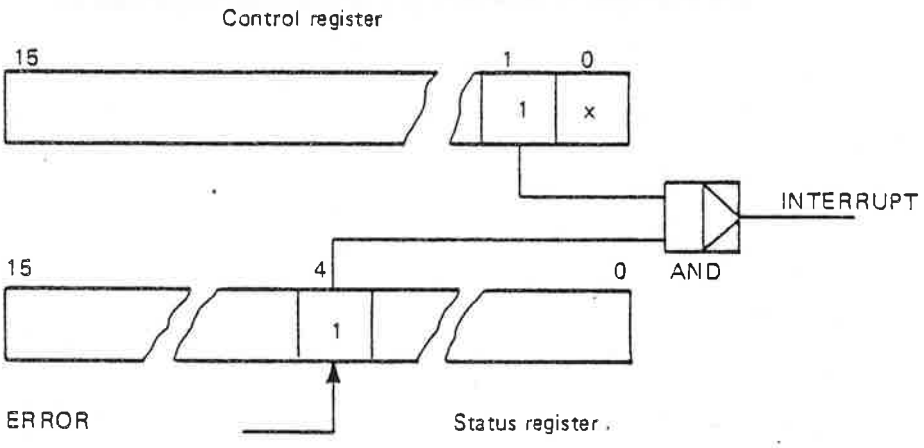
Bit 0 — Enable Interrupt on Device Ready for Transfer

Bit 0 set to "1" in an I/O interface channel enables for interrupt on device ready for transfer, i.e., if status bit 3 = "1".



Bit 1 — Enable Interrupt of Errors

Bit 1 set to "1" in an I/O interface channel enables for interrupt on errors in the channel.



Bit 2 — Activate Device

The control register bit 2 set to "1" on an interface will:

- in the input channel, enable reception of incoming data from external devices
- in the output channel, start output of the content in output channel data register
- start a DMA transfer (DMA controllers)

This bit is not always specified in a device programming specification. In such cases the activation is not needed.

The activate bit is normally static, i.e., it will remain on until Master Clear, Device Clear (control register bit 4) or a loading of the control register with bit 2 = "0" is done. Other reasons for resetting are given in the programming specifications.

Bit 3 — Test Mode

This bit set to "1" will loop output data back as input data in "off line" testing of an interface.

Bit 4 — Device Clear

The control register loaded with bit 4 = "1", generates a reset pulse on the accessed I/O interface. The reset pulse will clear all bits set in both the control and the status registers.

The device clear bit is usually implemented only in the input channel control register of a device. However, the bit resets both channels on a two channel device.

Bit 5 - 15 — Nondefined

Bits 5 - 15 in the control word are left for free use by the interface designer and is specified in the interface's programming specifications.

I.3.6 *PROGRAMMING OF A PIO INTERFACE*

I.3.6.1 *General*

Reading or writing data to a PIO interface has no meaning unless the interface is ready for transfer (RFT), i.e., status bit 3 = "1".

As indicated in the description of the control and status registers, RFT and control register bit 0 = "1" gives interrupt.

That is, there are two methods useable to check whether an interface is ready or not.

1. Polling for status register bit 3 = "1"
2. Enable for interrupt and wait (or do other useful things) until interrupt occurs

SINTRAN III (of course!) uses method 2. However, in the following description and programming examples method 1 will be used. Use and programming examples with interrupt is given in the section "The I/O System and the Interrupt System".

I.3.6.2 *Programmed Input from a PIO Interface*

Reading a byte or word from a PIO interface may be divided up into three steps:

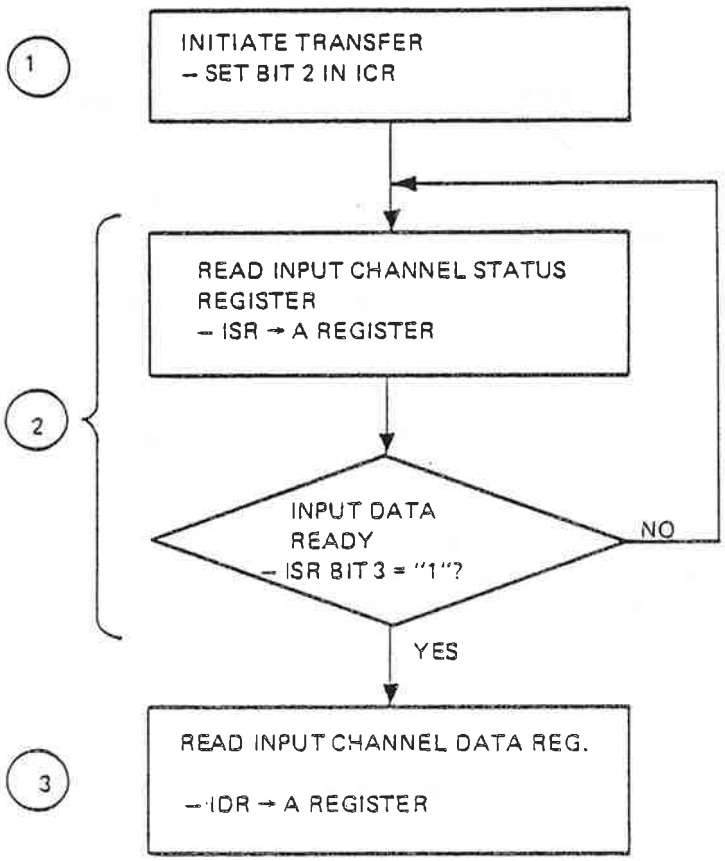
1. Enable the input channel for reception of incoming data

2. Check whether the input data register contains valid data ready to be read.

   If YES:

3. Read the input data register.

Due to standarization of both functions and formats of the status and control registers, this sequence is applicable on all Norsk Data produced PIO interfaces. The sequence is illustrated in the flow chart below.



ICR: Input channel control register

ISR: Input channel status register

IDR: Input channel data register

The flow chart is implemented by the following program:

```
DEVNO, <Device number>          % Device no. to interface to access
ICR = DEVNO + <ICR no.>         % Dev. reg. addr. = Dev. no. + ICR no.
ISR = DEVNO + <ISR no.>         % Dev. reg. addr. = Dev. no. + ISR no.
IDR = DEVNO + <IDR no.>         % Dev. reg. addr. = Dev. no. + IDR no.

INPUT,      SAA 4
            IOX ICR
LOOP,       IOX ISR
            BSKP ONE 30 DA
            JMP LOOP
READ,       IOX IDR
```

Example:

Input from Terminal no. 1.

```
DEVNO, 300                      % device no. terminal no. 1
ICR = DEVNO + 3                 % dev. reg. addr. for input control reg.
                                % (303)
ISR = DEVNO + 2                 % dev. reg. addr. for input status reg.
                                % (302)
IDR = DEVNO + 0                 % dev. reg. addr. for input data reg.
                                % (300)

INPUT,      SAA 4               % set bit 2 in A reg.
            IOX ICR             % A reg. → ICR
LOOP,       IOX ISR             % ISR → A reg.
            BSKP ONE 30 DA      % Is ISR bit 3 set?
            JMP LOOP            % No. i.e., not ready
READ,       IOX IDR             % Yes, i.e., ready. read data
            STA BUFF            % store data in BUFF
            JMP INPUT           % return
BUFF,       0
```

If the program above should be entered for execution via MOPC, it may be written as shown below.

```
                                executable:

20/         SAA 4                          20/170404
            IOX 303                          164303
            IOX 302                          164302
            BSKP ONE 30 DA                   175235
            JMP * –2                         124376
            IOX 300                          164300
            STA + 3                          004003
            JMP I * + 1                      125001
            20                                   20
BUFF,       0                                     0
```

I.3.6.3 *Programmed Output to a PIO Interface*

As for input, writing a byte or word of data to a PIO interface may be divided up into three steps:

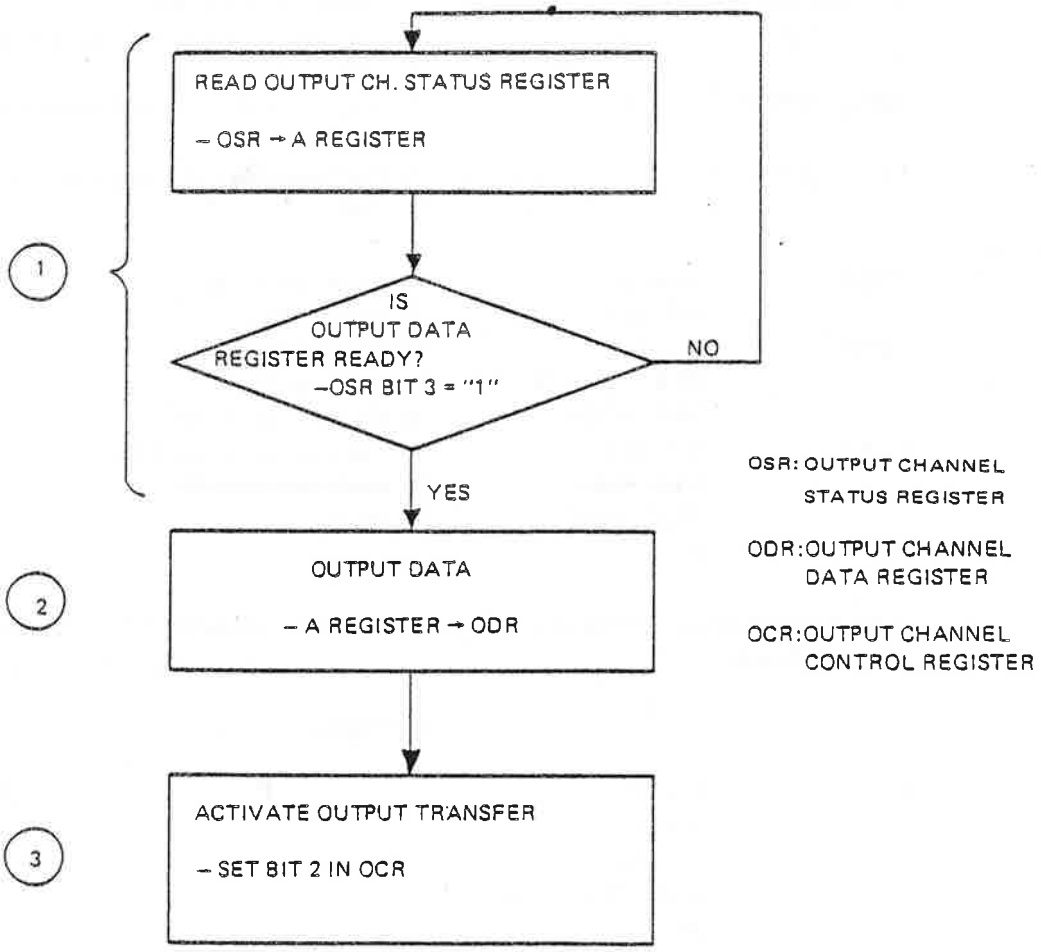
1. Check whether the output data register is empty and may accept data.

   If YES:

2. Write data to output for the output data register.

3. Initiate the output transfer by activating the output channel.

This sequence, applicable for output to all Norsk Data produced PIO interfaces, is illustrated in the flow chart below.



The flow chart is implemented by the following program:

```
DEVNO,  <device number>          % device no. of I/O interface to access
OSR =   DEVNO + <OSR no.>        % address of OSR
ODR =   DEVNO + <ODR no.>        % address of ODR
OCR = DEVNO + <OCR no.>          % address of OCR

OUTPUT,     IOX  OSR             % read status
            BSKP ONE 30 DA       % Ready?
            JMP  * –2            % NO
            LDA  BUFF            % Yes, initiate A register
            IOX  ODR             % A reg. → ODR
            SAA  4               % Initiate A reg. control
            IOX  OCR             % A → OCR
            JMP  OUTPUT          % Jump return
```

I.3.7 *PROGRAMMING OF DMA CONTROLLERS*

I.3.7.1 *DMA Controller Operation*

A DMA transfer may be divided into three steps:

- Initialization
- Transfer
- Termination and status check

Refer to the following descriptions and illustrations.

Initialization

A DMA controller has to be initialized before a transfer can be started. The initialization is done by a device driver program activated by the operating system when a transfer is needed.

The driver program accesses the DMA controller by means of IOX instructions. Through different transfer parameters, the driver tells the DMA interface what to do. The transfer parameters are written into physical device registers located on the controllers.

Typical transfer parameters and registers are:

- Memory Address Register (MAR) holds the first memory address to read from (DMA output) or write into (DMA input).

- Block Address Register (BAR) holds the first address to read (DMA input) from or write (DMA output) to on the physical device.

- Word Count Register holds the number of words to be transferred.

- Control register gives device function (read, write, etc.) and start (bit 2: activate device).

In addition, a status register, the MAR and the BAR, may be read for status check and test purposes.

The format of the registers and their associated register numbers are given in the hardware programming specifications.

Transfer

After initialization and start is given, the data transfer takes place. Data is exchanged between the DMA controller and memory at the speed determined by the device. CPU may run any activity not waiting for the DMA transfer to be completed.

Termination and Status Check

The DMA transfer is completed when the word counter is zero. On the DMA controller, status register bit 3 (ready for transfer) is turned on. If the interrupt system is on (ION) and interrupt is enabled on the controller, this causes interrupt on level 11. If the interrupt system is not on, a complete transfer is found by polling (continuous reading) on status register bit 3.

More about DMA controller operation is given in Section V.

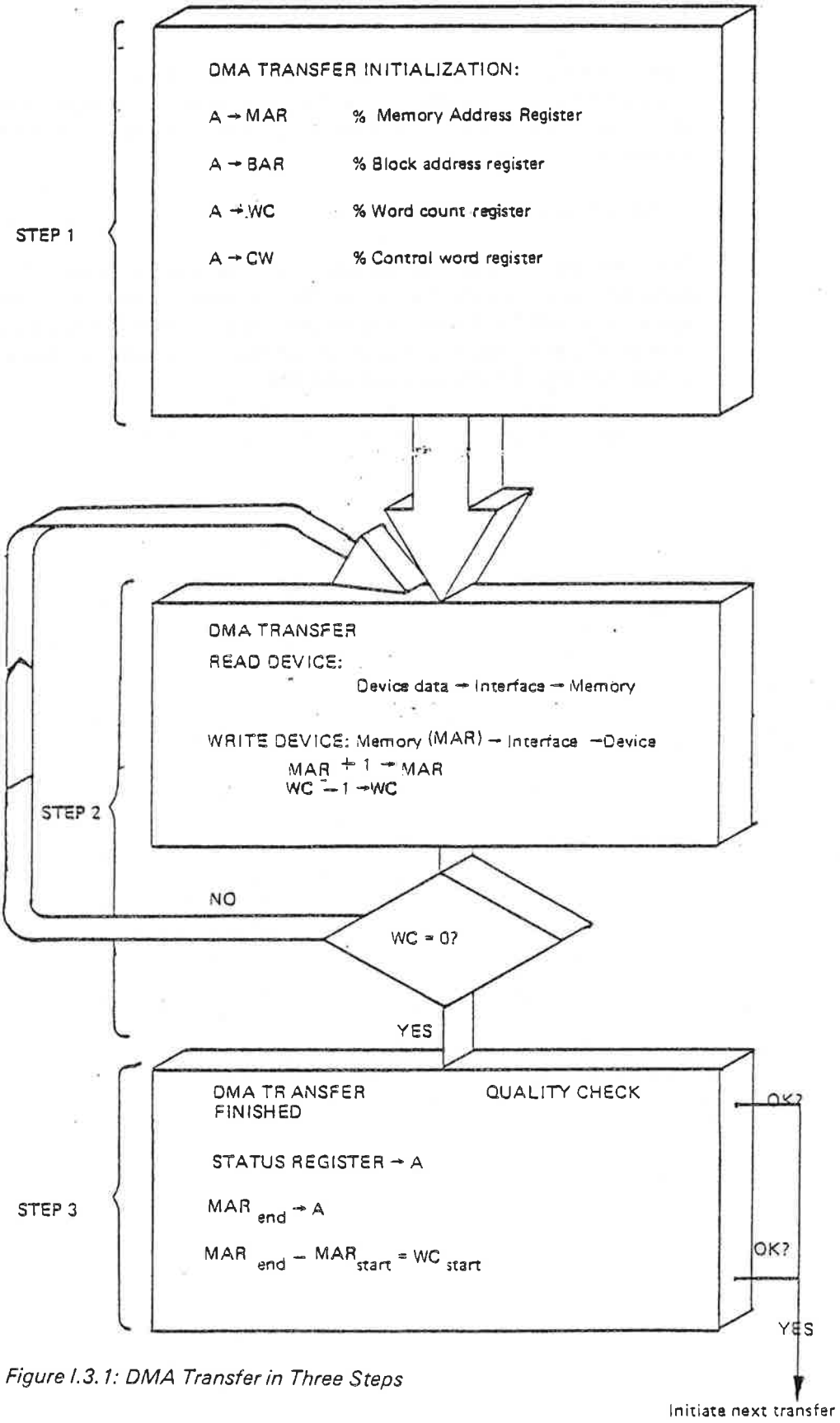


Figure I.3.1: DMA Transfer in Three Steps

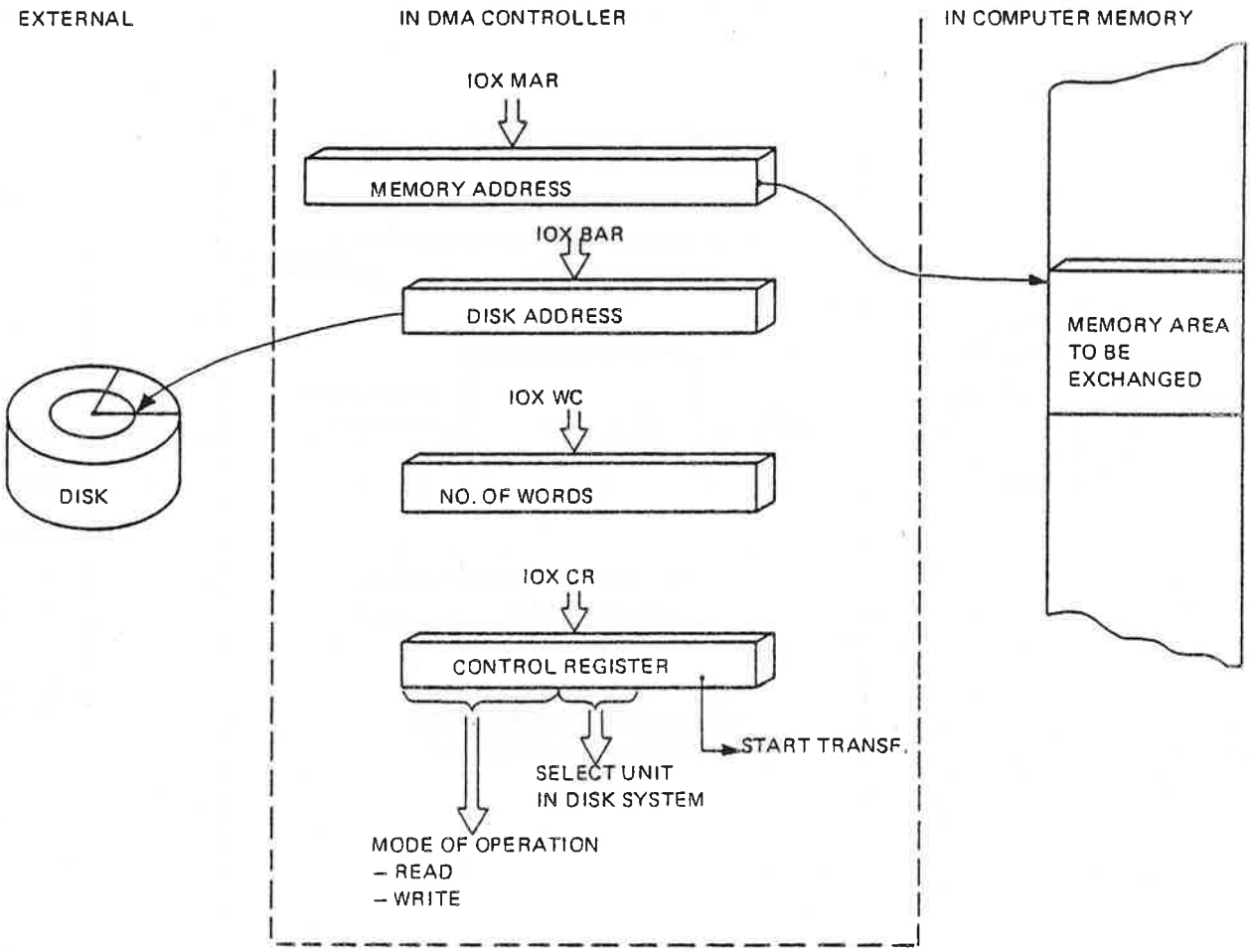


Figure I.3.2: DMA Initialization Illustration (Disk)

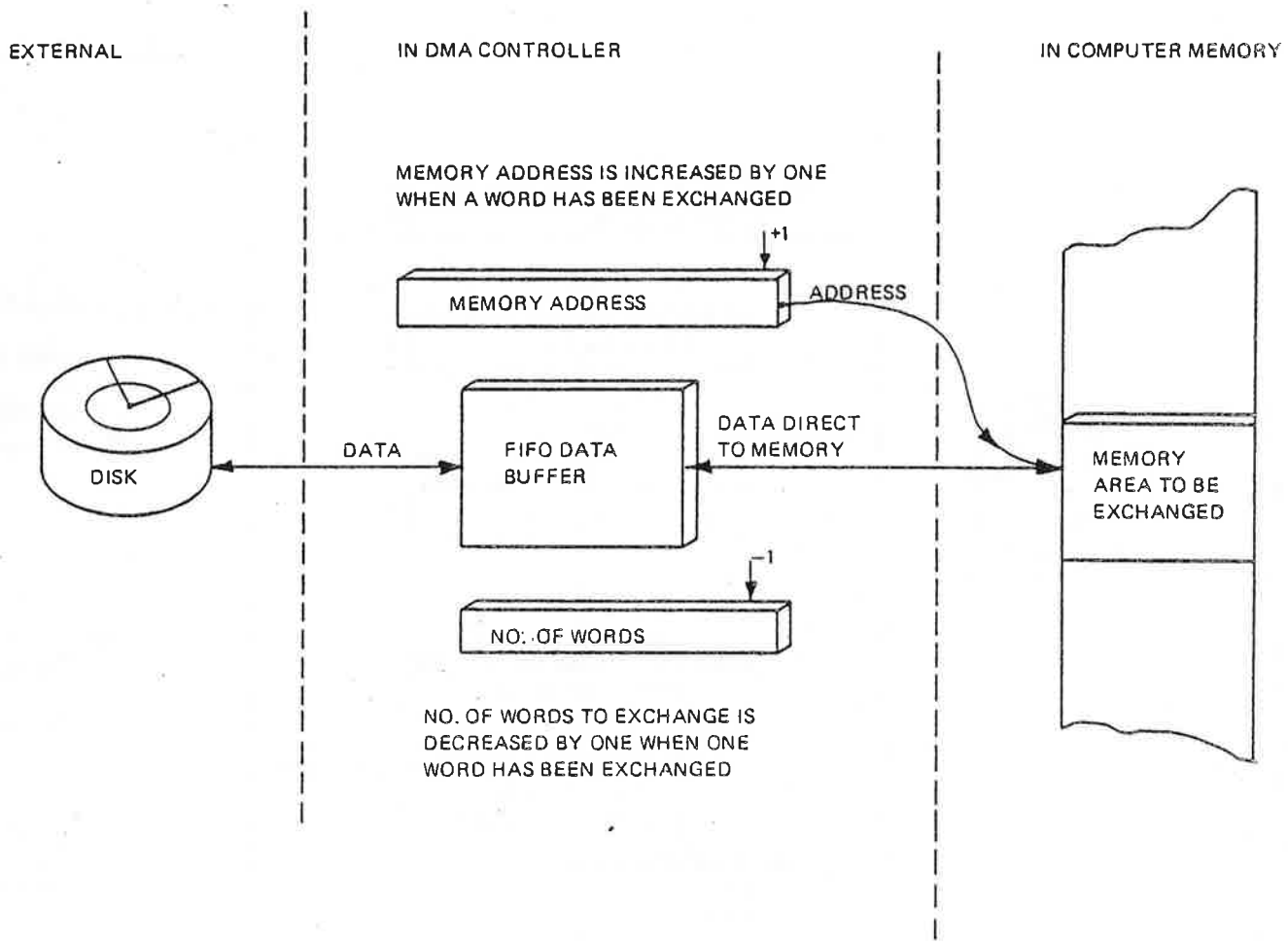


Figure I.3.3: DMA Transfer Illustration (Disk)

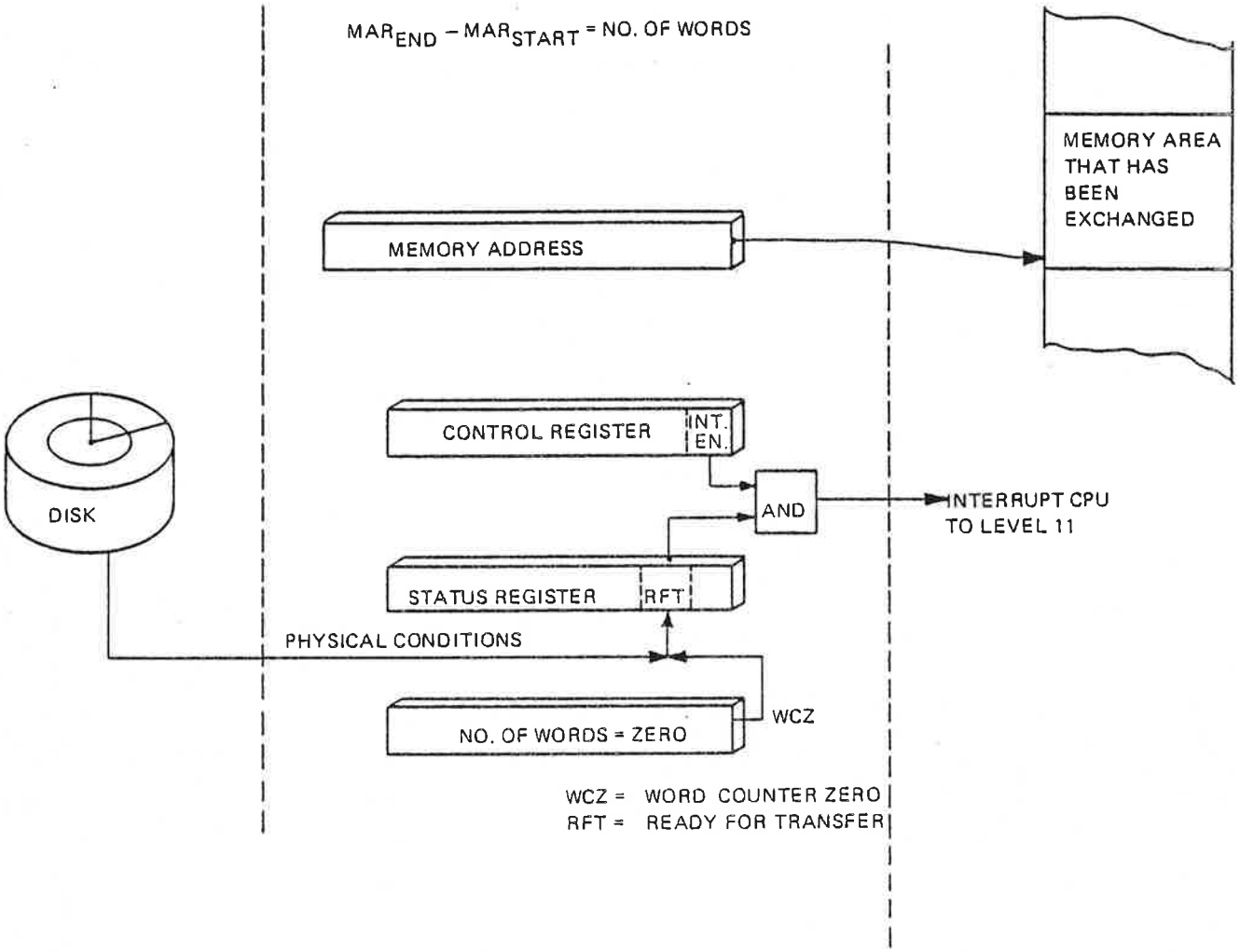


Figure I.3.4: DMA Transfer Completed (Disk)

I.4 THE I/O SYSTEM AND THE INTERRUPT SYSTEM

I.4.1 *GENERAL*

Under a running system (SINTRAN III), all I/O devices connected to NORD-100 will be prepared for operation and then allowed to operate asynchronously with respect to the CPU. That means that the I/O controllers activate themselves through an interrupt to the CPU if a status change occurs.

Possible status changes in the I/O system that may cause interrupt are:

1. End of operation interrupt

   and

2. Error interrupt

Which of the two status changes that actually caused an interrupt is found by reading the status register of the interrupting channel on the interrupting device.

1. End of operation interrupt occurs if a device is ready for transfer (status bit 3 = "1") and control register bit 0 = "1".

   This means for a PIO interface in the:

   - output channel
     data has been transmitted, next character to output may be loaded to the output data register

   - input channel
     input data is available

   or for a DMA controller:

   - DMA transfer is completed, i.e., word counter is zero.

2. Error interrupt occurs if an interface's status bit 4 = "1" (device error) and control register bit 1 = "1".

   Further details about the error is found in status register bit 5 - 15 (see programming specifications).

1.4.2 *NORD-100 INTERRUPT SYSTEM GENERAL DESCRIPTION*

The NORD-100 interrupt system consists of 16 program levels. The program levels are numbered from 0 - 15 with increasing priority; program level 15 has highest priority, program level 0 lowest.

At any time, the highest active level is running. This level is referred to as "current program level" (PL).

Interrupt occurs, if a level with higher priority than the one currently running is activated or if the current program level gives up its priority. The level change is a micro programmed procedure which purpose is to establish the interrupting level as the new currently running (PL), while the previous PL is put into a waiting state.

To ensure fast level changing (context switching), each of the 16 program levels have their own set of registers and status indicators located in a high speed register file.

The control of the NORD-100 priority interrupt system is based on two registers:

- the Priority Interrupt Enable register (PIE)
- the Priority Interrupt Detect register (PID)

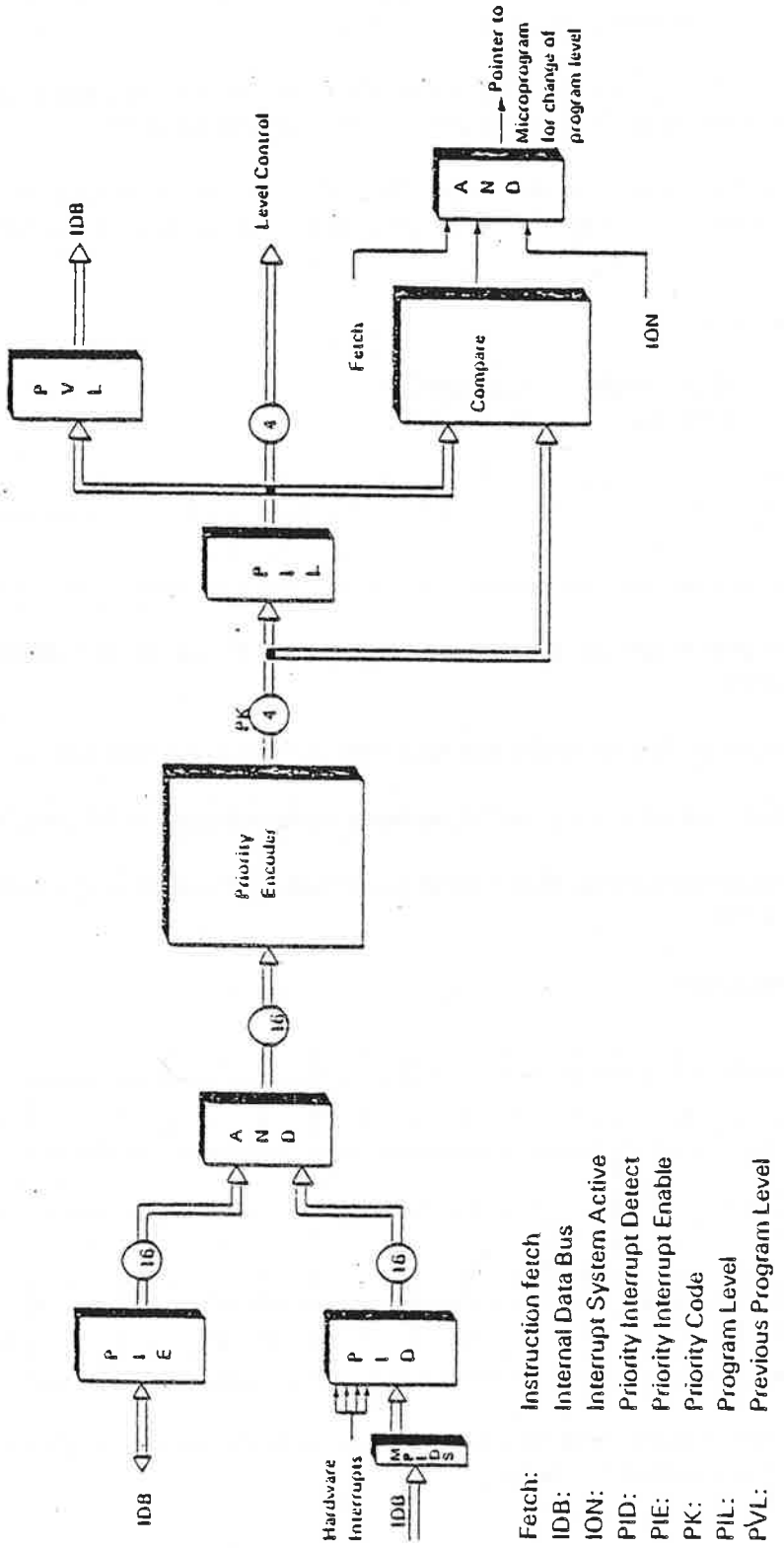


Figure I.4.1: External Interrupt System

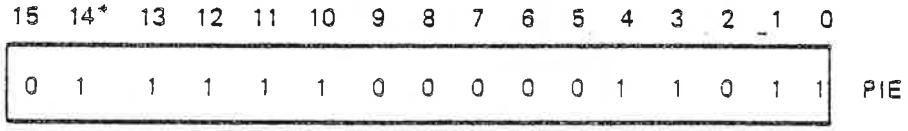
Both the PIE and PID are 16 bit registers where each bit corresponds to a program level (for example, PIE and PID register bit 10 corresponds to program level 10).

The PIE register is used as a static mask register; the PID register is dynamically set/reset depending on the activity on enabled program levels.

The PIE register is loaded by the TRR PIE instruction. A bit set in the PIE register, enables for interrupt on the program level corresponding to the bit's position in PIE.

Example 1:

```
LDA (76033      % 76033 → A
TRR PIE         % A → PIE
```



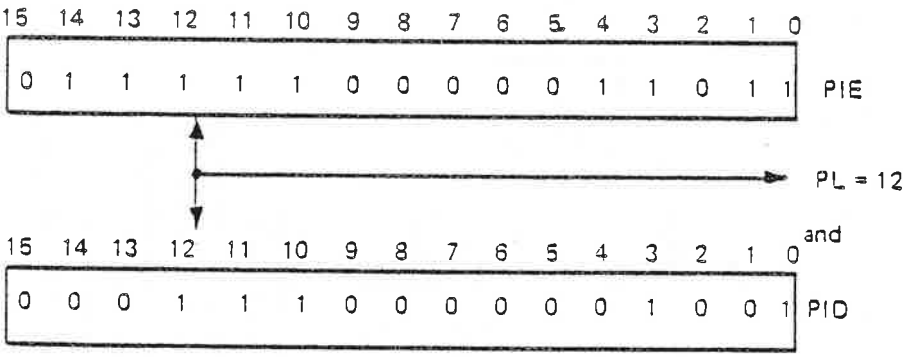
Interrupt is enabled on the levels 0, 1, 3, 4, 10, 11, 12, 13, 14*. All other levels are disabled.

* Level 14 requires speical attention, refer to the following description.

Each bit position set in the PID register corresponds to an active program level.

The current running level is given by the highest corresponding bits set both in PIE and PID.

Example 2:



In this example, level 12 will be the one currently running, while levels 11, 10, 3 and 0 are waiting for service.

The microprogrammed level change routine activated when an interrupt occurs does the following things.

1. The interrupt system is temporarily blocked to prevent false interrupts.

2. The program counter (CP) is copied to the saved program counter (SP) on the current level.

3. The PL (program level) register is copied into the PVL (previous program level) register.

4. The PK (new level priority code) register is copied into the PL (program level) register. (The CPU has, at this moment, changed level.)

5. The SP (saved program counter) on the new level is copied to the CP (current program counter).

6. A fetch is issued, i.e., the first machince instruction on the new level is asked for.

I.4.3 *NORD-100 INTERRUPT SYSTEM LEVEL ASSIGNMENT UNDER SINTRAN III*

The interrupt priority level assigned for the differnt activities under SINTRAN III is given in Figure I.4.2.

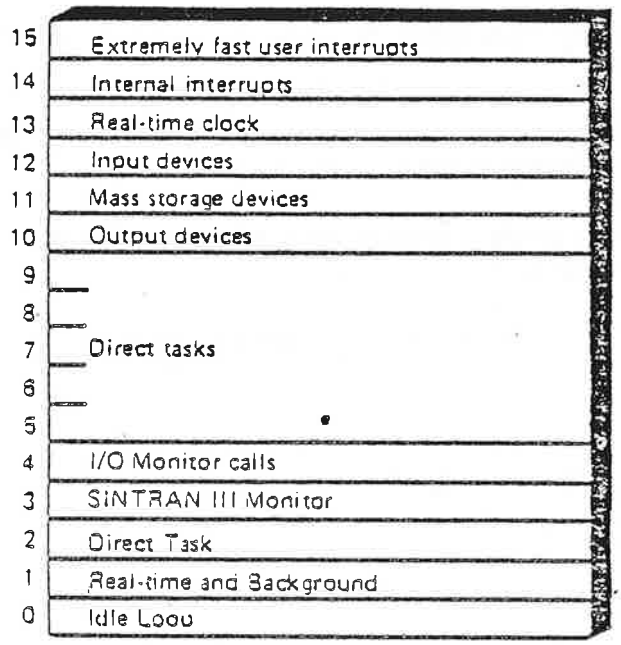


Figure I.4.2

Seen from a hardware point of view, all levels may be used by appropriate setting of the PIE and PID registers.

The actual levels used by SINTRAN III are 0, 1, 3, 4 and 10 to 14.

The program levels from 0 to 9 are completely controlled by software, i.e., interrupt on these levels has to be programmed by means of TRR PID/MST PID instructions.

Interrupt on the levels 10 to 15 may be set either by program or by hardware.

Each of the levels 10 - 13 and 15 are assigned a unique interrupt line in the NORD-100 bus which may be activated from any slot position.

Level 15 is not used by Norsk Data equipment but may be used by users requiring immediate access to the CPU. The levels 10-13 are used by Norsk Data produced input/output device controllers.

Level 14 is used for interrupts on internal CPU trap conditions. Level 14 is associated with an Internal Interrupt Enable (IIE) register and Internal Interrupt Detect (IID) register.

The IIE register may enable/disable interrupt generation on interrupt requests detected by IID (more information about the Internal Interrupt system is given in the manual "NORD-100 Functional Description").

I.4.4 *NORD-100 INPUT/OUTPUT DEVICE CONTROLLERS LEVEL USAGE*

As already mentioned, interrupt levels 10-13 and 15 may be activated by hardware through physical lines available in the NORD-100 bus. These lines go directly to the interrupt detect controller (PID register) in the CPU.

For Norsk Data produced equipment, the use of these lines have been standarized:

- Level 10 is used by the output channel of all PIO interfaces
- Level 11 is used by all DMA controllers
- Level 12 is used by the input channel of all PIO interfaces
- Level 13 is used by the real-time clock(s) and PIO devices which need special attention
- Level 15 is not used by Norsk Data produced hardware or software, but is available for special purposes needing immediate access. Note that level 15 has even higher priority than power failure (level 14).

  This is illustrated in Figure I.4.3.

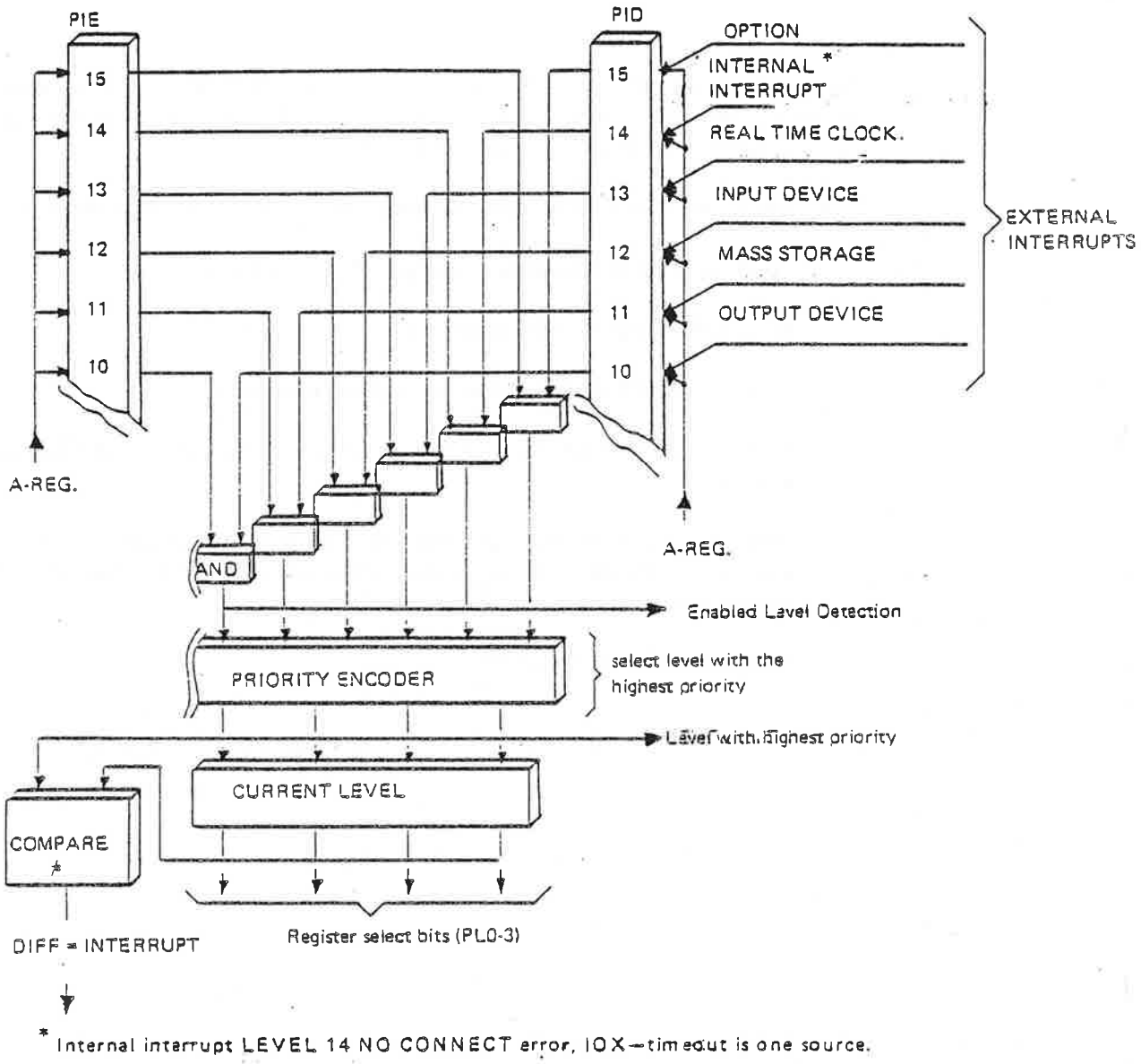


Figure I.4.3: NORD-100 External Interrupt Handling

1.4.5 *IDENTIFICATION OF AN INTERRUPTING I/O DEVICE CONTROLLER*

1.4.5.1 *General*

As indicated, more than one device may use the same interrupt line.

That is, when an interrupt on one of the levels 10 - 13 is detected by the CPU, the only known thing is that some kind of status change, in the I/O system, has occurred. In order to find the reason for the interrupt, the status register of the interrupting device has to be read (IOX) and investigated. But since each of the interrupt lines 10 - 13 are shared between all devices, the I/O interface causing the interrupt is unknown. That is, the device number is unknown and the IOX device register address is not possible to define.

Therefore, a device interrupt identification is needed.

1.4.5.2 *The Ident Code*

Each I/O device controller is assigned an interrupt vector referred to as "ident code" (id code). The ident code related to Norsk Data produced interfaces has been standarized and is given in Appendix A.

There is a unique correspondence between an external device, the device number and the ident code.

On the interfaces, the ident code (as the device number) is selectable by a thumbwheel to allow equal hardware modules to cover all ident codes related to one class of peripherals.

I.4.5.3 *The Ident Instruction*

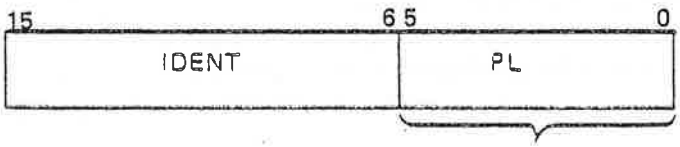
The ident instruction is a privileged machine instruction used in device interrupt identification.

The ident instruction, when executed, searches for interfaces with interrupt condition set, and returns the interfaces' ident code to the A register.

To maintain the interrupt priority the ident instruction searches only for interrupts on a specified level. The level to search on is specified in the ident instruction format.

Ident Instruction Format

IDENT PL xx % $10 \leqslant xx \leqslant 13$



Example:

The instruction IDENT PL 12 will only search for interfaces driving interrupt line level 12 (BINT12). A possible existing interrupt on level 10 or 11 is ignored and handled later by IDENT PL 10 and IDENT PL 11 respectively.

The ident code, which is unique for each device, is used to via an ident code table, to enter the interrupting device's data field. The data field contains an address pointer to the device's driver program which is started. The device driver program accesses the interrupting interface by means of IOX instructions which device register addresses are found in the device data field.

The driver program resets the interrupt condition on the interface, initiate next exchange and returns control to the program interrupted by the I/O interface.

I.4.5.4 *The Ident Search Mechanism*

The search for interrupt performed by the ident instruction may be divided up into three steps.

First, all I/O device controllers are presented the level on which the searching is going to be performed. The interfaces which have interrupt set on the specified level turns on a "flag" to signify this. The others do nothing.

Second, the CPU generates a search signal (INIDENT). The search signal is daisy chained via the module nearest to the CPU over to the next and so on. When the search signal finds the "flag" (interrupt on specified level) set, the search is stopped.

Third, the ident code from the interface which stopped the search is returned to the A register.

NOTE: In step three the interrupt is removed from interface by resetting of the interrupt enable bit, i.e., bit 0 or 1 in the channel's control register. That is, an interface channel should be reenabled after being served by the ident instruction.

The search mechanism used by the ident instruction includes some important notes.

Note 1: There should never be empty positions in the NORD-100 bus between the CPU and any I/O device controller. An empty position will stop the search signal and never release interrupts on modules in higher slot position numbers than the empty one.

Note 2: Between interfaces generating interrupt on the same level, the interface nearest the CPU has highest priority within the level (for slow devices such as terminals this has no practical effect).

I.4.5.5 *Input/Output Interrupt Programming*

I.4.5.5.1 Initialization of the Interrupt System

As indicated in the previous discussion, an interrupt is generated on I/O interfaces which are "ready for transfer" (RFT - status bit 3 = "1").

However, these interrupts are *not* noticed by the CPU unless the interrupt system is initialized and turned "on" (by the ION instruction).

After power up, PID, PIE and PL will be zero. The registers on level zero will be in use. The interrupt initialization must include the following:

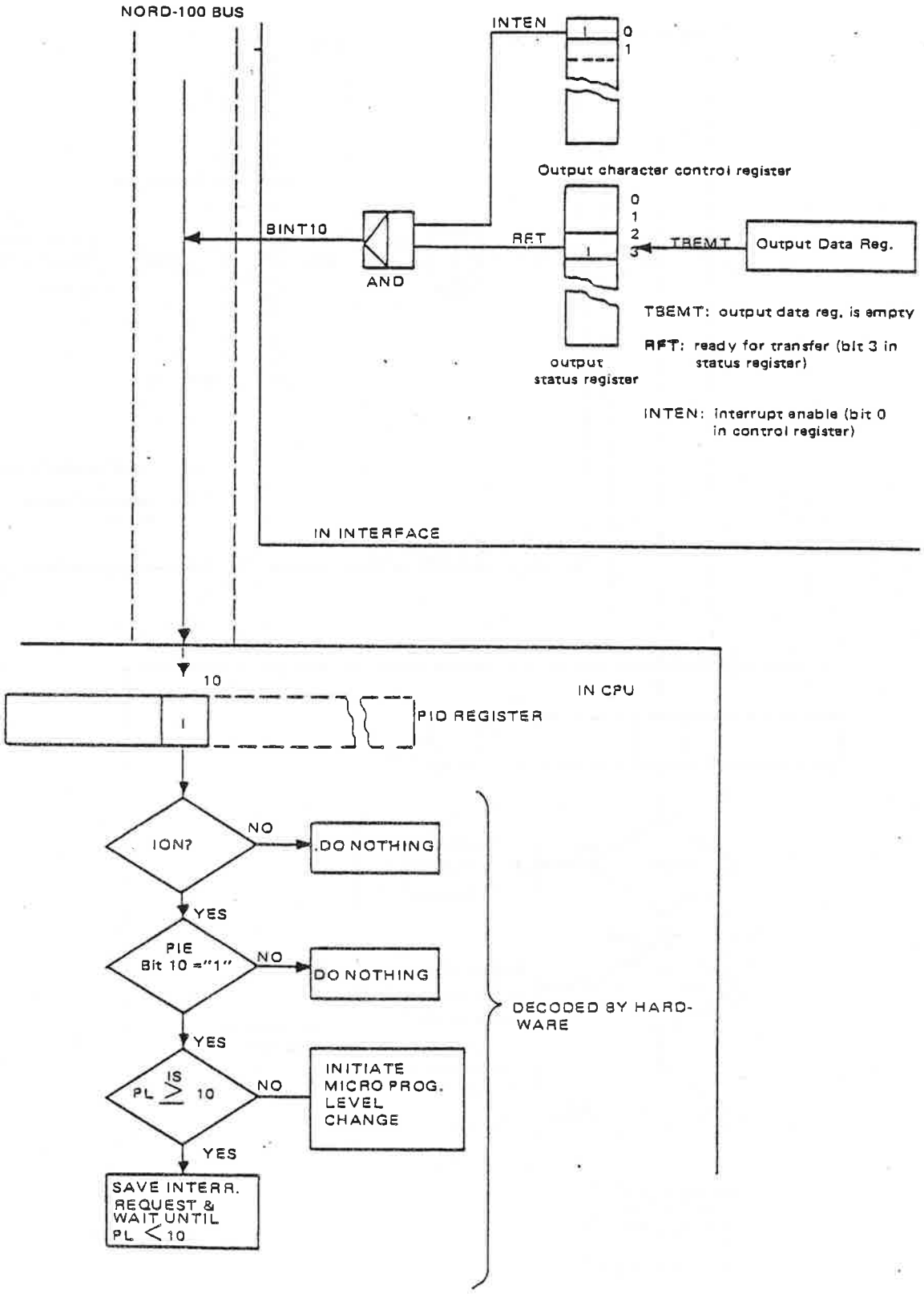
- Enabling of the desired program levels by proper mask setting in PIE (Priority Interrupt Enable).
- Enabling of the desired internal interrupt sources by proper mask setting in IIE (Internal Interrupt Enable register).
- The SP, saved program counters, on the levels to be used must be initialized, i.e., they must all point to the program to be executed on the different levels.
- If the Z (error) indicator is enabled for interrupt (IIE bit number 5), care should be taken that this indicator is cleared in the status register (bit number 3) for all levels being initialized.
- The IIC (Internal Interrupt Code) register, the PES (Parity Error Status) register and the PEA (Parity Error Address) register might be blocked after power up.
- By performing a TRA instruction for IIC and PES, all three registers will be unblocked and ready for use.
- The interrupt system is turned ON.

I.4.5.5.2 I/O Interface Interrupt Generation

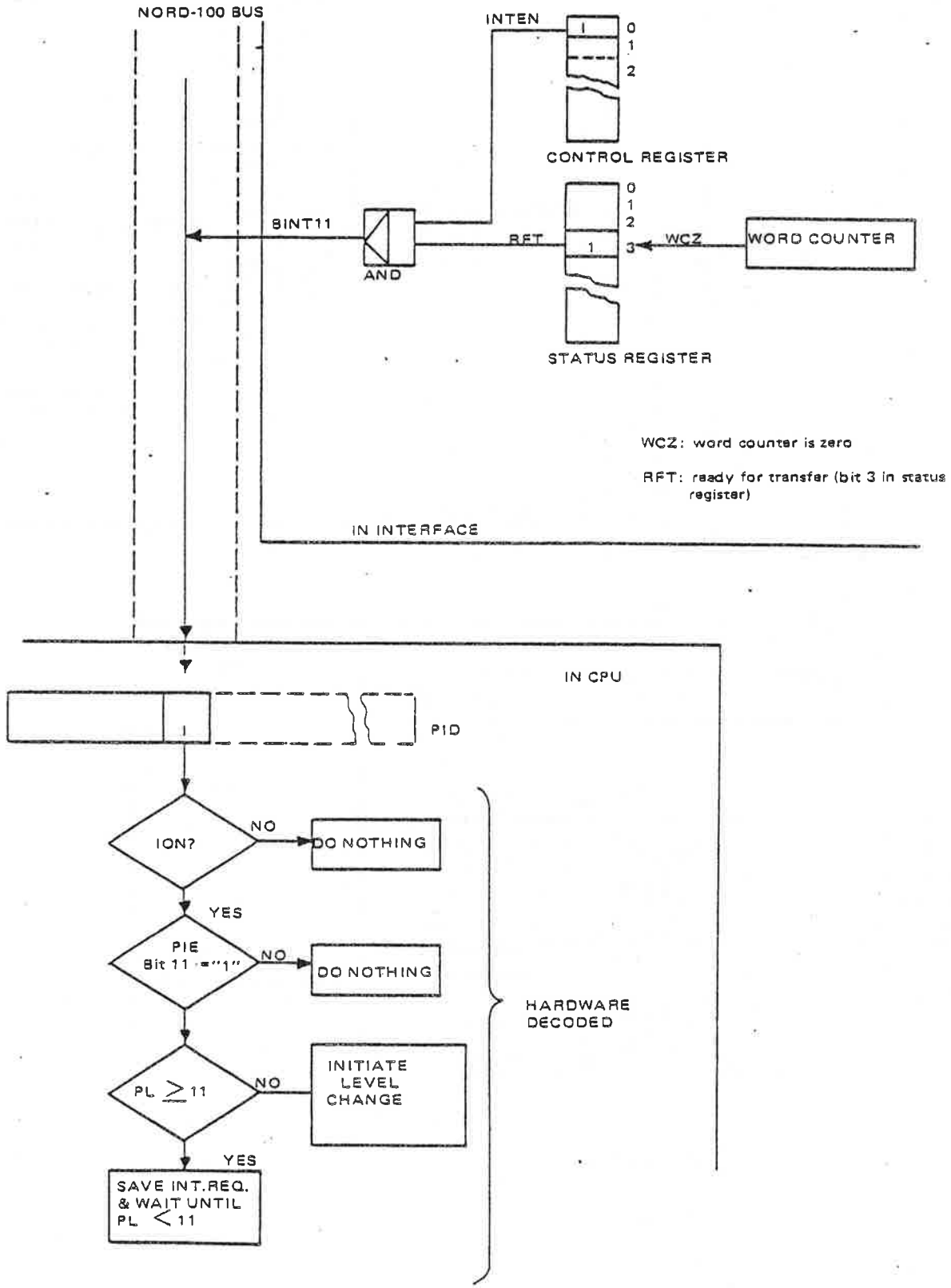
After the initialization of the interrupt system, the I/O interfaces should be enabled for interrupt generation. This is done by setting bit 0 (and 1) in the interfaces' control registers.

The following illustrations show how enabled interfaces generate interrupts on the levels 10, 11 and 12.

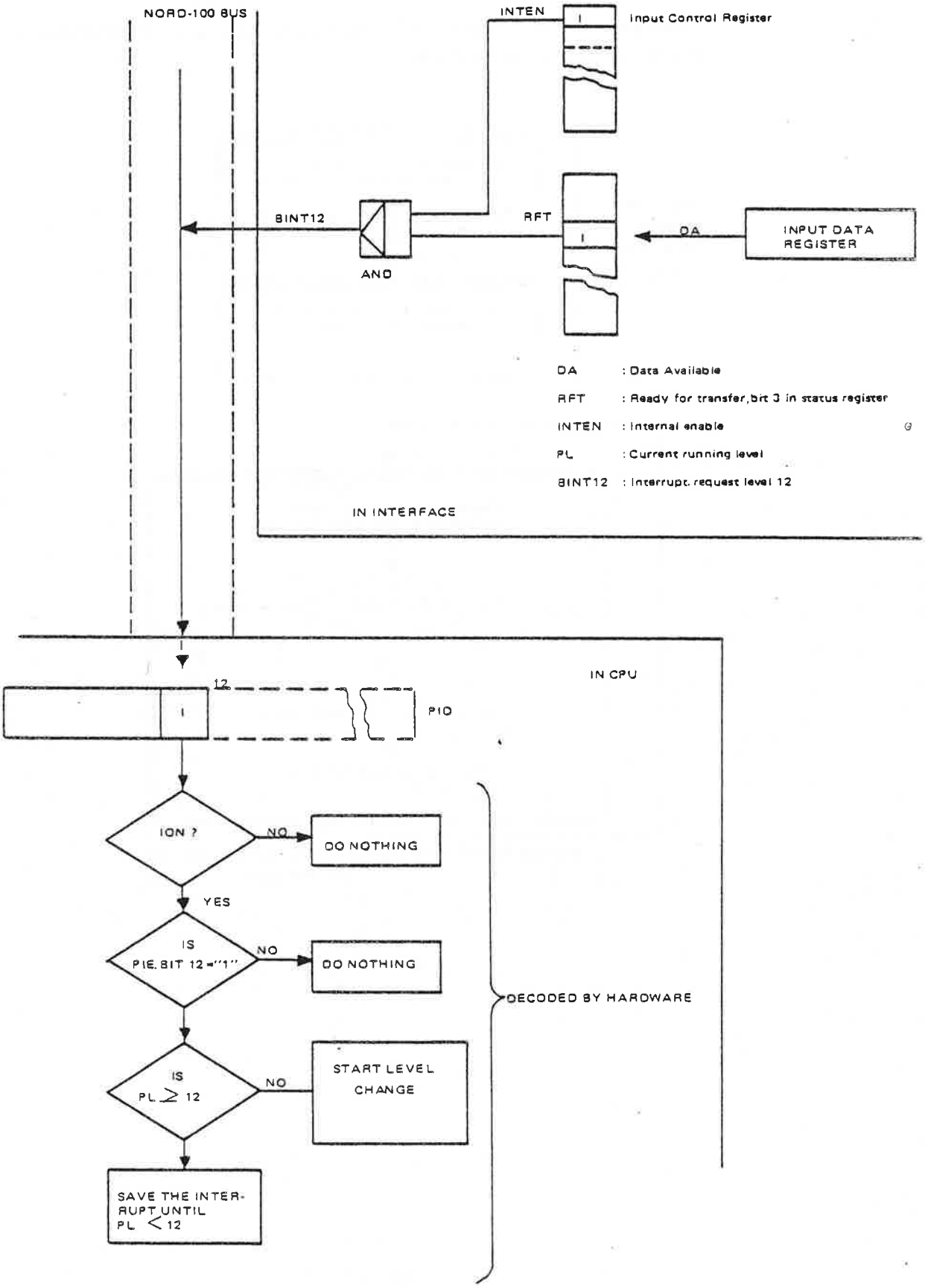
Interrupt Generation Level 10



Interrupt Generation Level 11

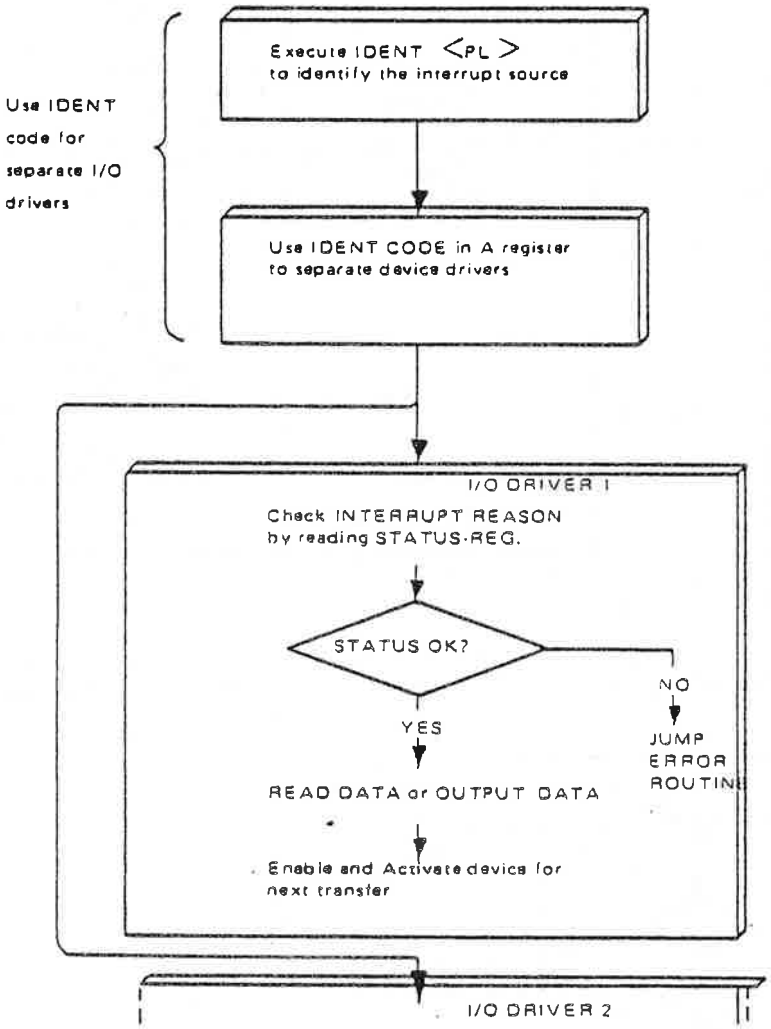


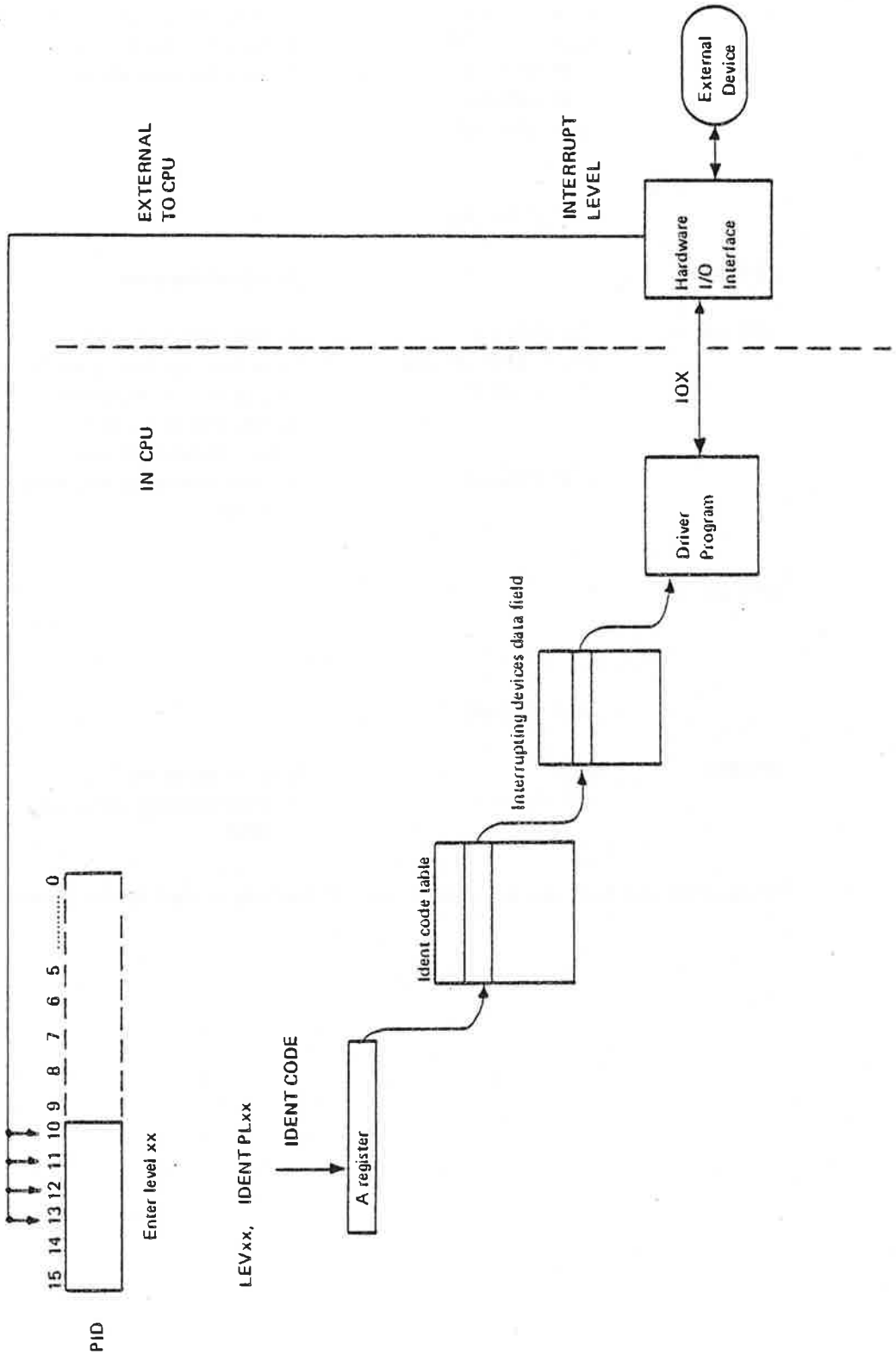
Interrupt Generation Level 12



I.4.5.5.3 Handling of I/O Interface Interrupts

When an I/O interrupt has occurred and the CPU has entered the interrupting level the flow chart below should be followed.




External Device
EXTERNAL TO CPU
INTERRUPT LEVEL
Hardware I/O Interface
IOX
IN CPU
Driver Program
Interrupting devices data field
Ident code table
IDENT CODE
A register
LEVxx, IDENT PLxx
Enter level xx
15 14 13 12 11 10 9 8 7 6 5 0
PID

A simplified implementation of the flow chart is given in the following program.

```
LEV xx,       IDENT  PL xx          % read ident code
              RADD   SA  DP         % modify program counter
              JMP    ERROR          % with the ident code
              JMP    DRIVER 1
              JMP    DRIVER 2
                .
                .
              JMP    DRIVER n

ERROR,        .....                 % report the error

DRIVER 1,     IOX    RSTAT          % read status register
              BSKP   ZRO 40 DA      % is interrupt reason error?
              JMP    ERROR          % Yes, bit 4 (dev. error) = "1"
                .                   % No, serve the ready
                .                     for transfer condition
              JMP    FINISH         % jump to routine for giving up
                                      priority
.
.
DRIVER n,     ....
                .
                .
                .
              JMP    FINISH

FINISH,       WAIT                  % give up priority
              JMP    LEV xx         % entry point to LEVxx after
                                      WAIT
```

More about the hardware involved in the I/O interrupt system is described in Part IV.

II THE NORD-100 BUS

II.1 GENERAL

As already stated in Part I, the NORD-100 bus efficiently organizes the interconnections and transactions between the hardware modules in the NORD-100 computer system.

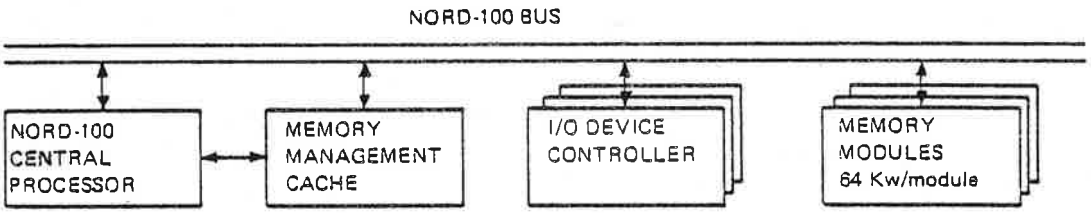


Figure II.1: The NORD-100 Bus

The NORD-100 CPU uses the bus for memory transfers to and from the memory system, and for I/O transfers to and from its connected peripherals.

In addition, some NORD-100 bus peripherals (DMA controllers) can initiate transfers directly to and from the memory system.

II.2 THE NORD-100 BUS — BUS REQUESTORS

As indicated introductionally, the NORD-100 bus may be thought of as a common resource, shared between:

- the NORD-100 CPU
  and
- DMA controllers

In addition, refresh of the NORD-100 memory system requires access to the NORD-100 bus.

As a common resource, the NORD-100 bus has to be allocated before it may be used. Once allocated, the bus is busy to other activities, while the granted bus user may transfer one word. As the transfer is completed the NORD-100 bus should be released for next eventually requesting bus user.

II.3 A NORD-100 BUS CYCLE — GENERAL DESCRIPTION

Before going into details, it could be convenient to look at some of the fundamental concepts behind the organization of the NORD-100 bus activities.

The basic activity is, not to forget, to exchange information between a source and a destination.

The event of exchanging one word is referred to as one bus cycle. In case of refresh, one bus cycle is an address and dataless memory refresh cycle.

A NORD-100 bus cycle is always preceded by an allocation and terminated with what here is called release.

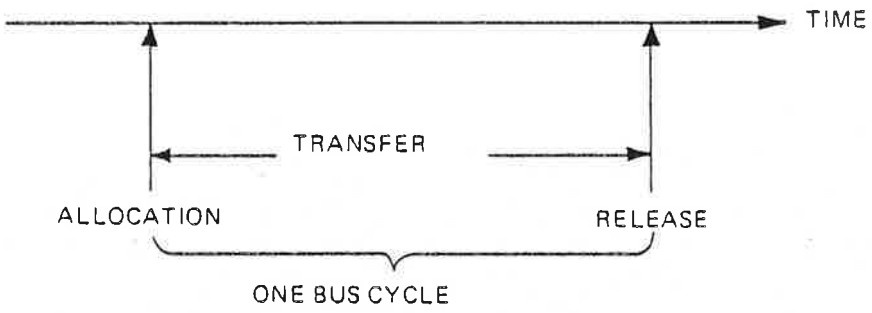


Figure II.3.1: A NORD-100 Bus Cycle

Refer to the illustration (Figure II.3.1).

A NORD-100 bus cycle consists of three main events:

- allocation of a requesting bus user
- transfer one word
- termination and release of the bus

The allocation is handled by a Bus Control Unit (BCU) physically implemented on the CPU module.

That is, the BCU decides which of the three possible bus users that should be given the next bus cycle.

The transfer and termination of a bus cycle is organized as a handshake between the selected bus user (CPU, DMA or refresh) and it's accessed device (I/O interface or memory).

The concept allows completely asynchronously operation within a bus cycle.

II.4 FUNCTIONS OF THE BUS CONTROL LOGIC (BCU)

II.4.1 *ALLOCATION OF THE NORD-100 BUS*

As a common system source the NORD-100 bus has to be allocated before it can be used. The three possible bus users that may ask for access to the bus is:

- the NORD-100 CPU
- DMA controllers
- memory refresh

Access to the NORD-100 bus has to be asked for at the Bus Control Unit (BCU) through a bus allocation request.

The above mentioned bus users are operating completely asynchronously. That is, the bus allocation requests may be passed to the BCU completely asynchronously, even simultaneously causing competition.

Conflicts are avoided by the BCU through a priority allocation arbiter. Based on a priority allocation algorithm and the present active bus allocation requests, one requestor is selected for the next bus cycle while possible other active requests must wait.

II.4.1.1 *The Allocation Requests*

Each of the three possible bus requestors have their own unique request line input to the allocation arbiter in the BCU. A request is issued by activation of the request lines.

II.4.1.1.1 NORD-100 CPU Allocation Request

The NORD-100 CPU may request access to the system bus for one of six reasons:

- instruction fetch
- operand read
- indirect address read
- operand store

} memory access

- programmed access to the I/O system

} I/O system access

- programmed access to external system control registers*

} system control registers

* Control registers not located on the CPU or MMS module (for example, Error Correction Control Register on the memory modules — TRR ECCR).

If any of the six above mentioned operations are in progress, the CPU microprogram activates the signal BUSRQ (CPU bus request) input to the CPU allocation arbiter.

In addition to the signal BUSRQ passed to the BCU, the microprogram informs the CPU bus-handshake-logic which transfer that is to be performed.

The bus-handshake-logic is then ready to start the transfer when it, from the BCU, receives acknowledge on the allocation request.

II.4.1.1.2 DMA Allocation Request

A DMA controller requests allocation of the NORD-100 bus to get direct access to the NORD-100 memory system.

When a DMA controller needs more data to output or have a word ready to be written to computer memory, it generates a bus allocation request to the BCU.

Then the controller has to wait for the BCU to acknowledge the request before the exchange may be performed.

When there are several DMA controllers, each controller's request signal is "wired or" (connected) to one bus allocation request line input to the BCU.

This signal is named BREQ (Bus REQuest) and may be driven from any slot position in the NORD-100 bus.

In order to select only one DMA controller as granted bus user at the time, the DMA request acknowledge signal is daisy chained in the NORD-100 bus backplane. The daisy chain then establishes a sequential priority scheme between DMA controllers.

Physical implementation of the daisy chain is described in later sections.

II.4.1.1.3 Memory Refresh Allocation Request

Refresh is a periodical operation needed by the dynamic MOS memory circuits used in the NORD-100 main memory. By allocating the NORD-100 bus during the refresh period two problems are solved in a simple and efficient way.

- The refresh cycle is synchronized to other bus activities.
- The memory system is unaccessible since the system bus is allocated, i.e., blocked.

The refersh allocation request is initiated every 15 μ s by an oscillator on the CPU module activating the signal RFREQ (refresh request) input to the BCU.

II.4.1.2 *The Bus Control Unit (BCU) Allocation Priority Rules*

From the above discussion of the potential bus allocation request signals, Figure II.4.1 shows what is the request inputs to the BCU allocation logic.

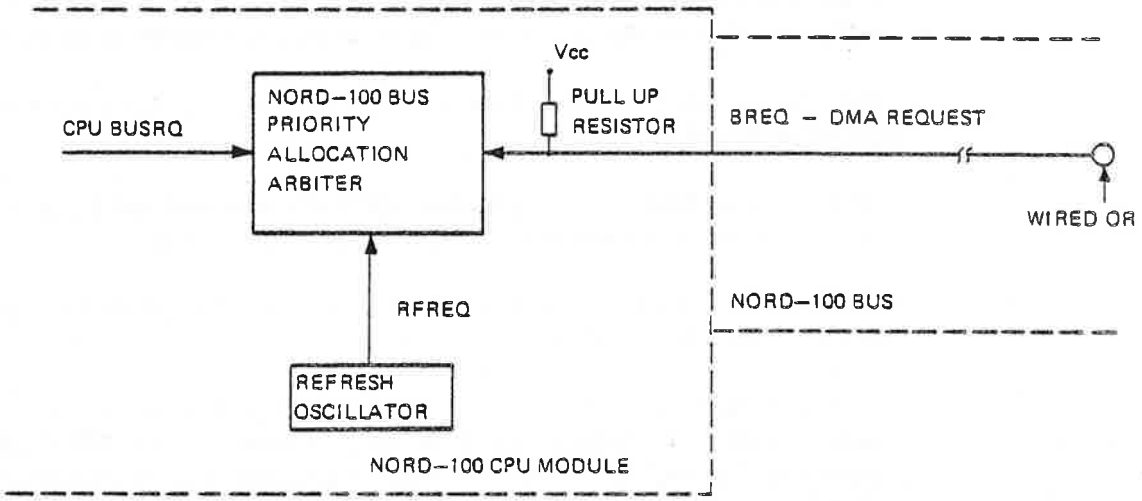


Figure II.4.1: NORD-100 Priority Allocation Arbiter and the Bus Requests

The BCU priority arbiter controls the access to the NORD-100 bus based on the following rules:

- Already allocated bus is not interruptable.
- The bus cycle going on is aborted if it exceeds a time limit of 8 μ s.

If the bus is idle and an allocation request appears alone, the rule is:

- *first come, first served.*

Since all the request signals appear asynchronously, conflicts between simultaneous requests may happen. In such cases, the following priority rules are used:

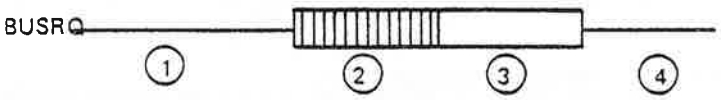
- RFREQ and BREQ are both handled as DMA requests but RFREQ is given highest priority.

- If both DMA requests (RFREQ and BREQ) and CPU request (BUSRQ) are present, priority is given to the one not having the previous cycle (toggled priority).

The bus control priority arbiter and the bus allocation is illustrated in the following examples.

In the examples a simplified presentation of the request lines and the bus usage is used.

Example:



1 CPU does not try to allocate the NORD-100 bus

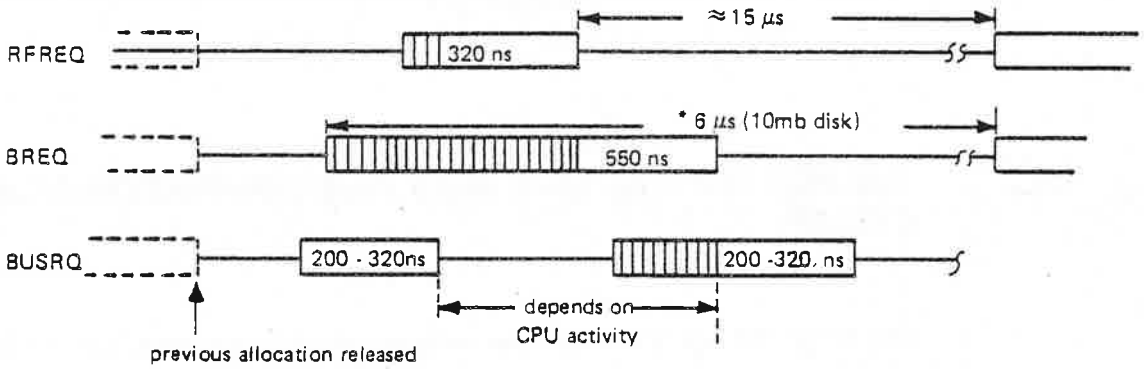
2 CPU tries to allocate the bus but has to wait for another bus activity

3 The bus is allocated to the CPU

4 CPU has finished its bus cycle and releases the bus.

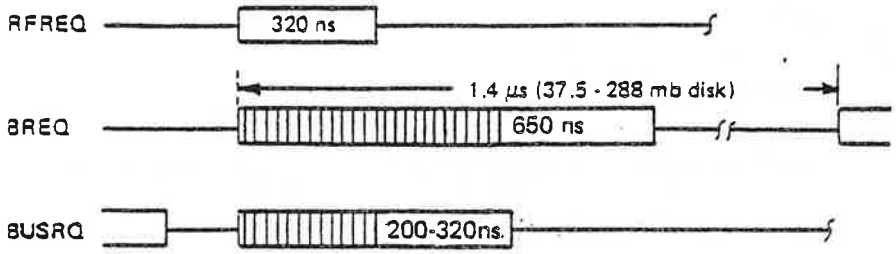
Example 1:

CPU requests the NORD-100 bus first, after previous allocation is released.



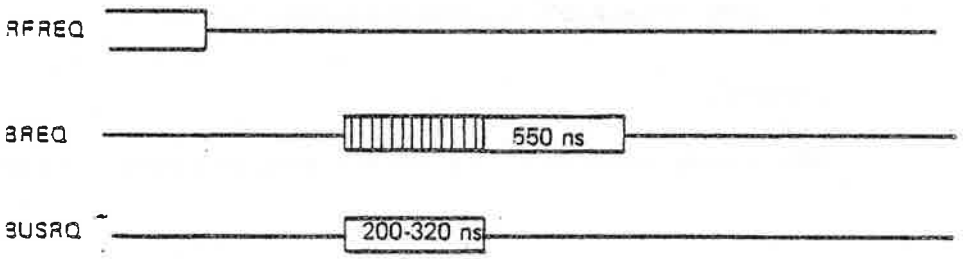
Example 2:

All requests appear simultaneously but CPU had the previous cycle.



Example 3:

CPU bus request and DMA request appears simultaneously (DMA cycle steal) but previous cycle was for DMA (RFREQ).



II.4.2 *THE BUS CONTROL UNIT (BCU) AND TERMINATION OF BUS CYCLES*

The NORD-100 bus is allocated and released on one cycle basis, i.e., for every word to exchange or for one memory refresh cycle.

One bus cycle should not last for more than 8 µ s. This is monitored by the BCU. At the time of allocation, the BCU starts a timer. This timer is reset by a handshake mechanism between granted bus user and its accessed device signifying transfer completed. The transfer completion signal is named BDRY (Bus Data ReadY) and will be explained in proper context later.

If the NORD-100 bus is not released it causes system hang-up. To prevent such a situation, a bus cycle that exceeds 8 µ s is aborted by the BCU timeout timer.

The faulty cycle is reported to the CPU as an internal interrupt (level 14). Refer to Figure II.4.2 for illustration.

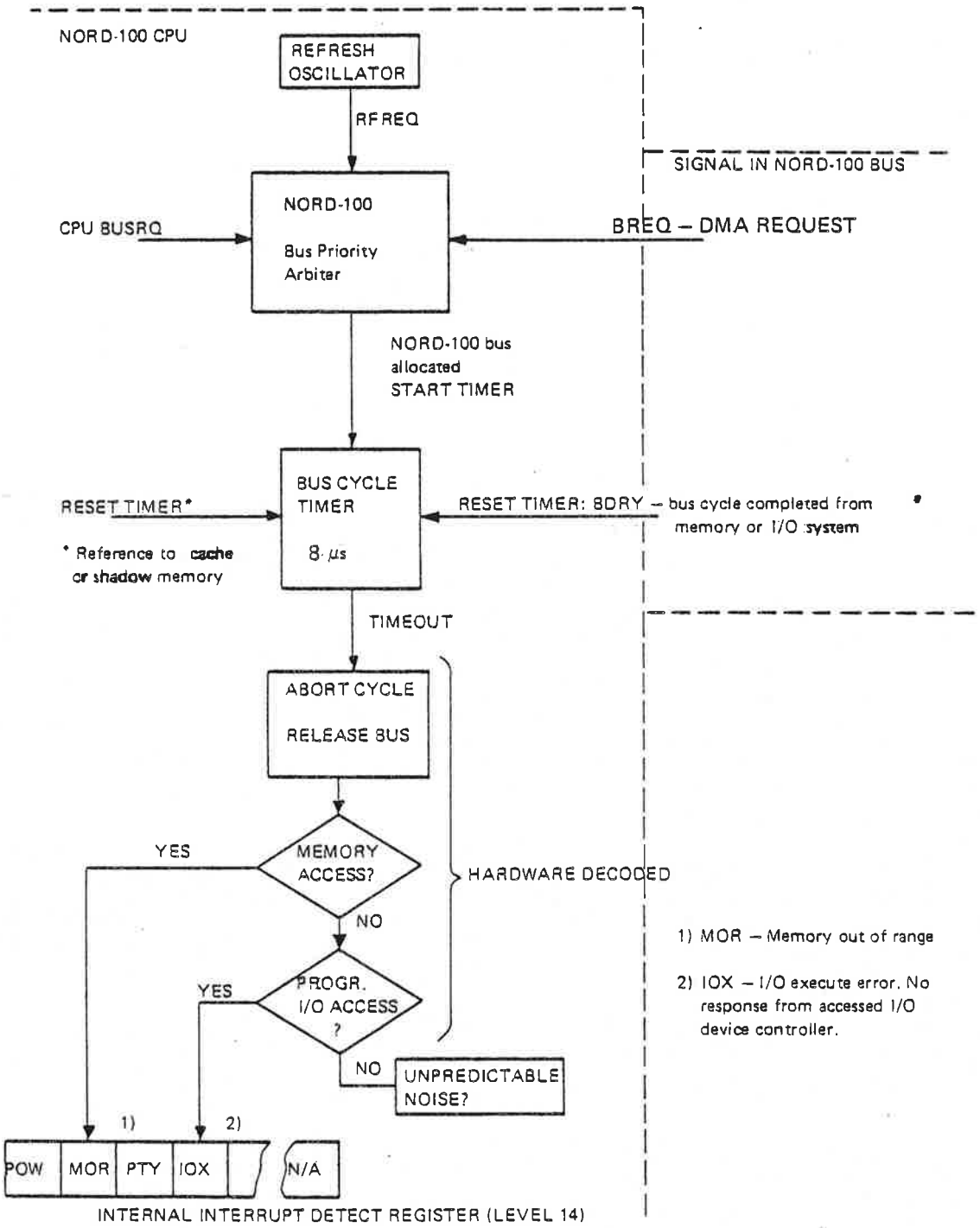


Figure II.4.2: NORD-100 Bus Termination Circuitry

II.5 DATA TRANSFER ON THE NORD-100 BUS — GENERAL DESCRIPTION

When a requesting bus user receives the allocation acknowledge signal from the BCU, the requesting bus user is granted and given access to the NORD-100 bus. The granted user may then initiate a transfer.

As previously mentioned, the BCU is not involved in the control of data exchange on the NORD-100 bus. The data exchange is completely controlled by the granted bus user in handshake with its accessed device. This allows asynchronous operation independent of the CPU speed and clock frequencies.

Although the granted bus user may either be the CPU, a DMA controller or memory refresh, only CPU and DMA cycles include any data exchange. Therefore, granted bus user when talking about data transfers is either the CPU or a DMA controller.

II.5.1 *ORGANIZATION OF A NORD-100 BUS CYCLE*

Due to the multiplexing of addresses and data on the same physical bus lines, one NORD-100 bus cycle may be divided up into two subcycles.

- First, an address cycle
- Second, a data cycle

See Figure II.5.1 for illustration.

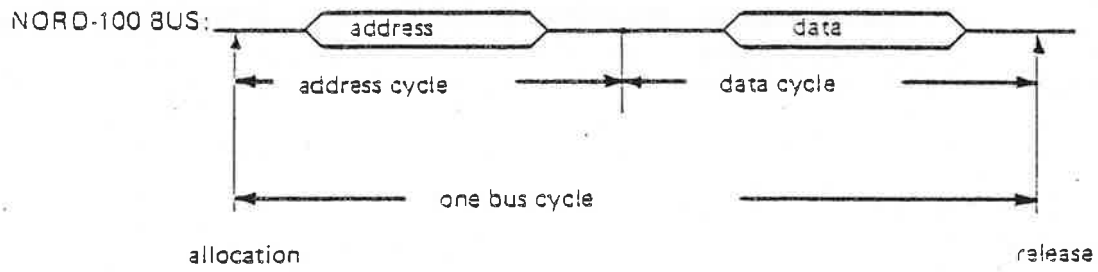


Figure II.5.1: NORD-100 Bus Cycle Illustration

Independent of the transfer (CPU or DMA transfer) actually going on, the sequence is always as given in the illustration. The Address cycle precedes the data cycle.

II.5.1.1 *The Address Cycle*

When combined with the address strobe $BAPR_0$ (Bus Address PResent) the multiplexed address/data line BD0-23 carry addresses.

In the address cycle, of a transfer, the granted bus user presents an address to the device that is to be accessed. That is, physical memory address to the memory system or a device register address to the I/O interfaces. In addition, which of the interrupt levels 10, 11, 12 or 13 that are to be investigated by the IDENT PLxx instruction is specified in the address cycle.

II.5.1.2 *The Data Cycle*

In the data cycle, data is exchanged between the I/O register or memory location specified in the address cycle.

By contrast of the address cycle, the data cycle includes an asynchronous handshake between granted bus user and accessed device.

The handshake is initiated by the granted bus user issuing a control signal indicating "start of data cycle". Termination and release of the bus cycle is done by the accessed device activating a signal indicating "transfer completed".

Depending upon the transfer direction, the signals "start of data cycle" and "transfer completed" are also used as data strobes.

To get a better understanding of this handshake, the following sections provide a detailed description of both IOX/IOXT, IDENT PLxx and DMA controller usage of the NORD-100 bus.

III PROGRAMMED INFORMATION EXCHANGE BETWEEN I/O INTERFACES AND NORD-100 CPU — EXECUTION OF THE IOX/IOXT INSTRUCTIONS

III.1 INTRODUCTION

The only useable instructions in the NORD-100 instruction repertoire for information exchange between I/O interfaces and NORD-100 CPU are the IOX and IOXT instructions.

Both these instructions exchange information between the CPU A register and a specified I/O interface register.

Other sections in this manual have covered how and when to use the IOX/IOXT instructions in programmed control of PIO and DMA interfaces.

Thus, this section is intended to give a description of how these instruction are executed. That is, how they are handled by the CPU and how they appear to the NORD-100 bus and I/O interfaces.

III.2 IOX/IOXT INSTRUCTION EXECUTION

III.2.1 *GENERAL*

All instructions in the NORD-100 instruction repertoire are carried out by microprogrammed routines. Each instruction has its own special routine which is entered based on the instruction's operation code. Thus, execution of the IOX or IOXT instruction appears to the CPU as an execution of a routine in the microprogram.

In this section, the IOX and IOXT instructions are described as they appear to the CPU. That is, how the microprogrammed routines for these instructions are entered and executed.

III.2.2 *IOX/IOXT INSTRUCTION ENTRY POINT GENERATION*

The microprogrammed routine for an instruction to be executed is entered based on the instruction's operation code. The operation code is used as a "look up" address in a map table where the microprogram address to the routine relevant to the actual instruction is found.

The basic theory behind the address generation to the microprogram is covered by the manual "NORD-100 Function Description" and is not repeated here. However, some remarks relevant to the IOX and IOXT, as privileged instructions, are given.

IOX and IOXT are privileged instructions. If MMS is on, privileged instructions may only be executed by programs executing with a ring – priority greater or equal to two (RING $\geq$ 2).

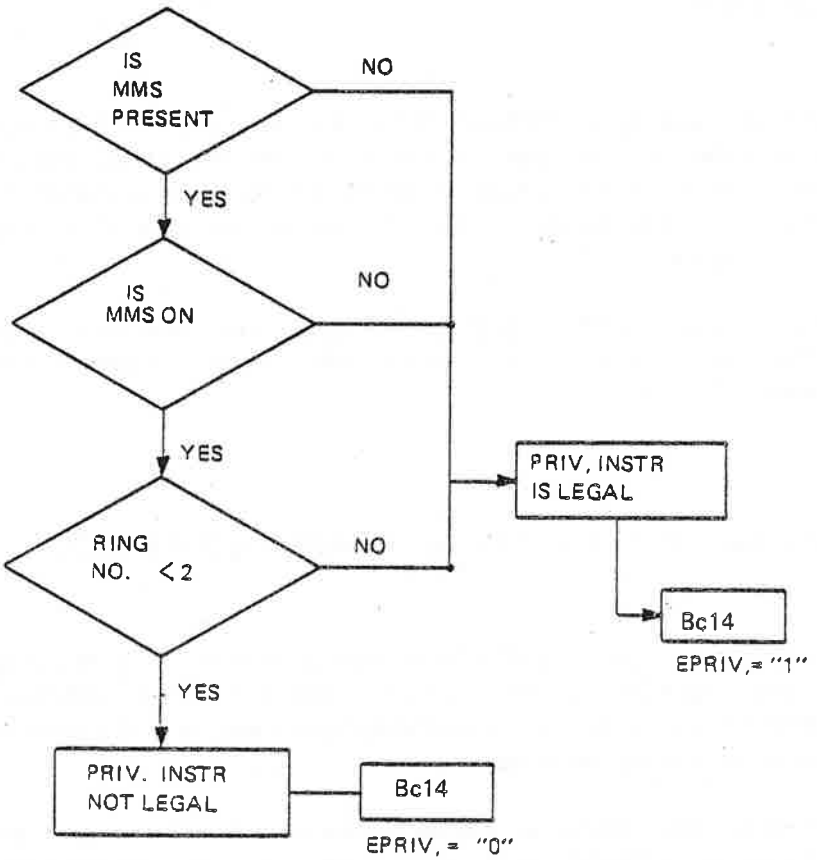
That is, IOX/IOXT and other privileged instructions are protected from unauthorized use.

A program's ring priority is known by the MMS through the used Paging Control Register (PCR). Thus, it is the MMS that informs the CPU whether a program is authorized to execute a privileged instruction or not.

The CPU uses this information to modify the address input to the microaddress map. That is, illegal (nonprivileged) use of a privileged instruction gives another address to the map than legal use of the same instruction. The map location pointed to by the illegal address points to a microprogram ERROR routine.

That is, IOX/IOXT and other privileged instruction are not started in the microprogram unless the use of the instructions is legal.

Legal use is decoded by the MMS and presented to the CPU as described below (Figure III.2.1).



EPRIV: Enable privileged instruction

Bc14: B connector line carrying EPRIV from MMS (Memory Management System) to CPU

Figure III.2.1:

The CPU handling of EPRIV, equal 0 or 1 is illustrated in Figure III.2.2.

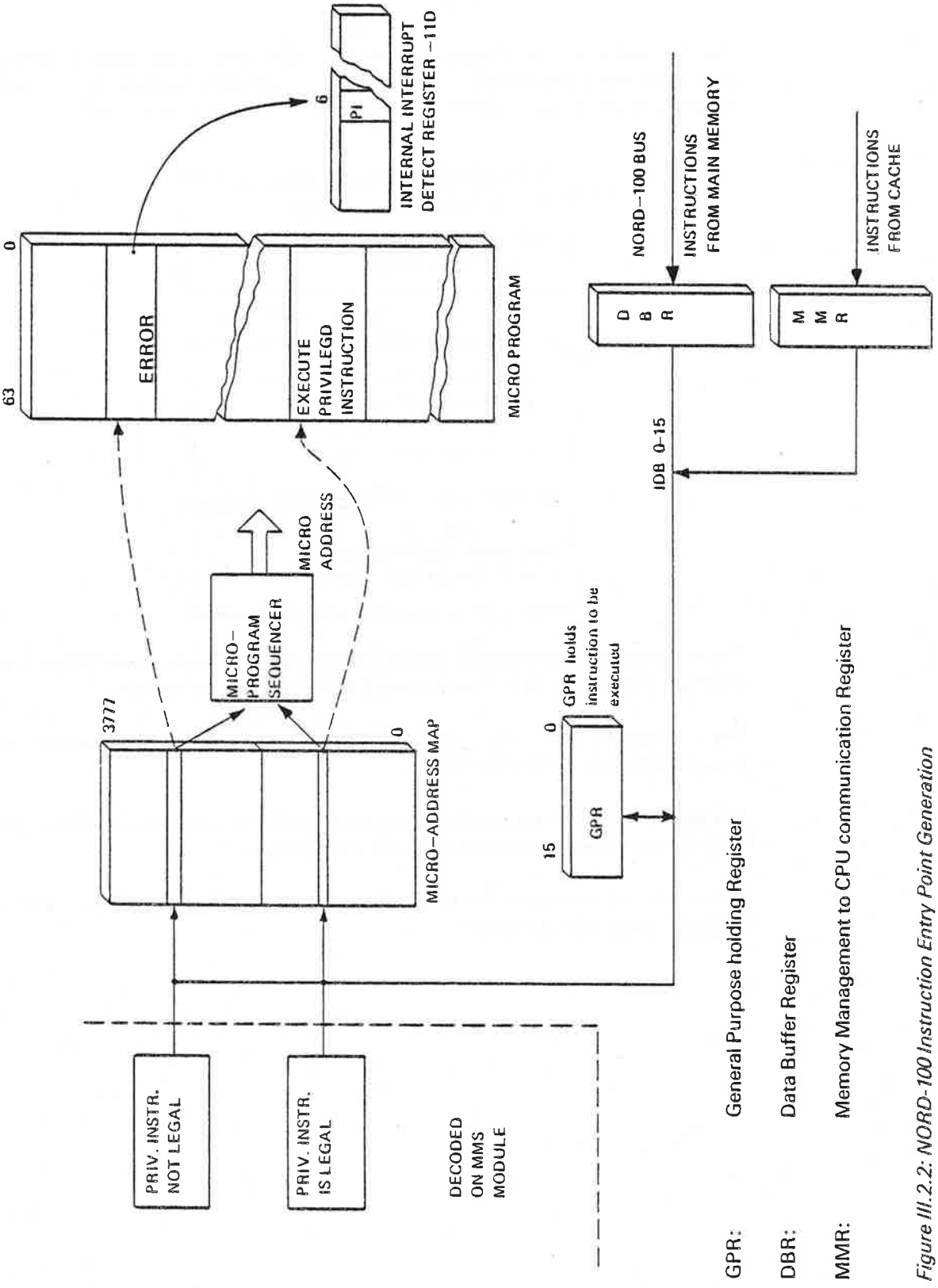
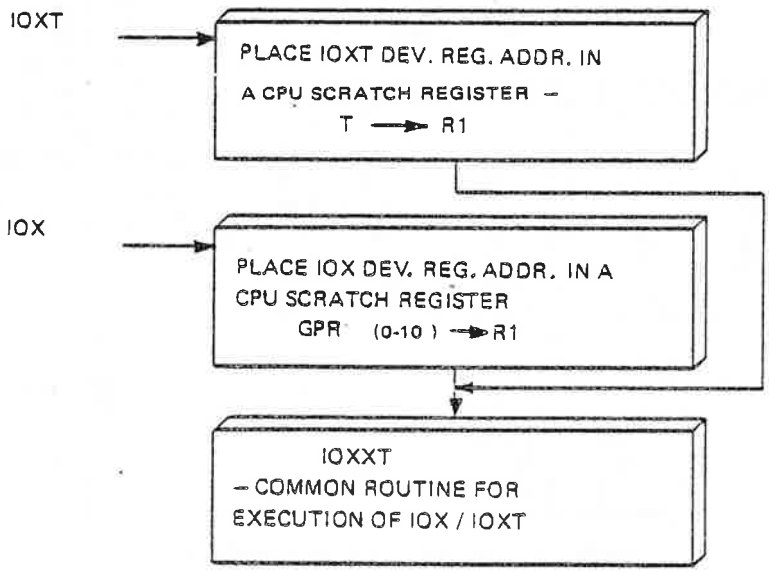


Figure III.2.2: NORD-100 Instruction Entry Point Generation

III.2.3 *IOX/IOXT MICROPROGRAM OPERATION*

The IOX and IOXT instructions have different operation codes. That is, although they have the same function, they will enter different routines in the micro-program. However, as shown below, they soon enter a common routine.

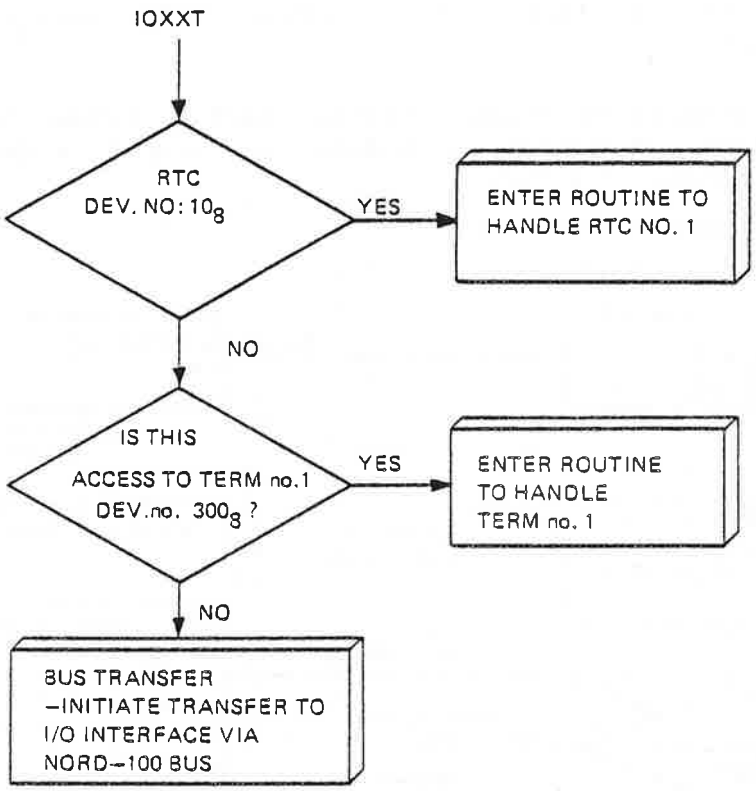


The functions of the common routine (IOXXT) is, of course, to exchange information between CPU A register and a specified I/O interface register.

The I/O register to be used, specified in the device register address, normally has to be accessed via the NORD-100 bus.

The only exceptions are access to terminal number 1 and the Real-Time Clock (RTC) which both are implemented on the CPU module.

That is, the microprogram has to seperate the handling of terminal no. 1 and the RTC from other I/O interfaces.



In this section it is assumed that the transfer is performed via the NORD-100 bus, i.e., IOX/IOXT device number unequal to 300_8 and 10_8.

A CPU bus transfer is controlled by the microprogram. However, the asynchronous handshake mechanism through various control signals included in a NORD-100 bus cycle is handled by a CPU Bus Handshake Logic (BHL). Figure III.2.3 illustrates the interaction between the CPU microprogram and CPU bus handshake logic during IOX/IOXT execution.

When a NORD-100 bus cycle is needed, the microprogram requests allocation of the NORD-100 bus. In addition, the microprogram informs the CPU bus handshake logic which cycle (IOX, IDENT or memory reference) it should handle.

In case of IOX, the device register address to be used in the IOX address cycle is enabled onto the internal CPU bus (IDB).

Refer to Figure III.2.3 and note how the microprogram execution is synchronized to the bus activities by the bus handshake logic through the "WAIT" and "RELEASE WAIT" mechanism.

When the address cycle is completed, the microprogram continues by enabling the CPU A register onto IDB. Note that this is done independent of the actual transfer direction.

At the completion of the data cycle, the microprogram loads the content of DBR into the CPU A register.

As indicated, the microprogram performs one output operation and one input operation. This is done to make the microprogram independent of the transfer direction.

The actual transfer direction is handled "privately" between the CPU bus handshake logic and accessed I/O device. How this is done is described in the next section.

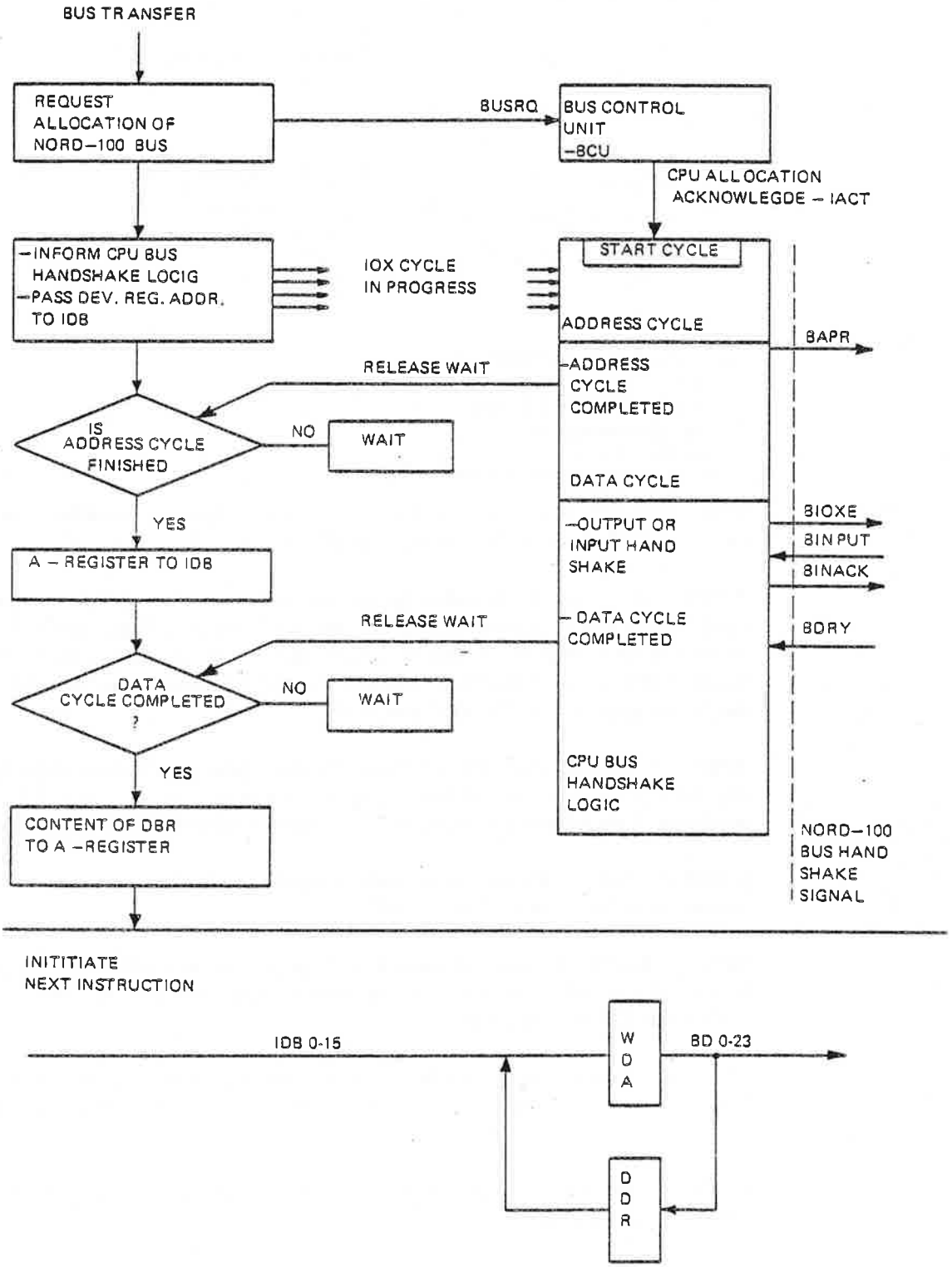


Figure III.2.3: IOX/IOXT Microprogram and CPU Bus Handshake Logic Interaction

III.2.4 *IOX/IOXT EXECUTION AND THE NORD-100 BUS*

This section is intended to give a description of how execution of the IOX/IOXT instructions appear to the NORD-100 bus. This includes the operation of the CPU bus handshake logic and the control signals involved during IOX/IOXT executions.

The background for signal responses from the accessed PIO or DMA controller is described in a later section.

As indicated in the previous section it is the microprogram that "knows" that an access to the NORD-100 bus is needed, and thus indicates the bus cycle.

The microprogram generates a bus allocation request to the BCU, informs the bus handshake logic that an IOX cycle is in progress and enables the device register address into IDB.

The bus control logic is then ready to operate when it receives the bus allocation signal from the BCU.

As other bus cycles, the IOX/IOXT cycle is handled by the bus handshake logic as an address cycle and a data cycle.

In the address cycle, all I/O interfaces, simultaneously, are presented the device register address. Based on this address, one I/O device register is selected to communicate with the CPU during the data cycle.

As indicated in the previous section, the microprogram is independent of the IOX/IOXT transfer direction. This is also the case for the IOX/IOXT address cycle. However, the data cycle of course is direction dependent.

Therefore, the following text and illustrations cover an output transfer and an input transfer as case 1 and 2 respectively.

Case 1: IOX/IOXT Output Transfer

After initialization by the microprogram, the bus control logic is waiting for the allocation acknowledge signal to start the bus cycle. The device register address resides in the Write Data Address (WDA) buffer.

Refer to Figure III.2.4.

NOTE: The convention for representation of the bus signals in timing diagrams is given in Appendix F.3.

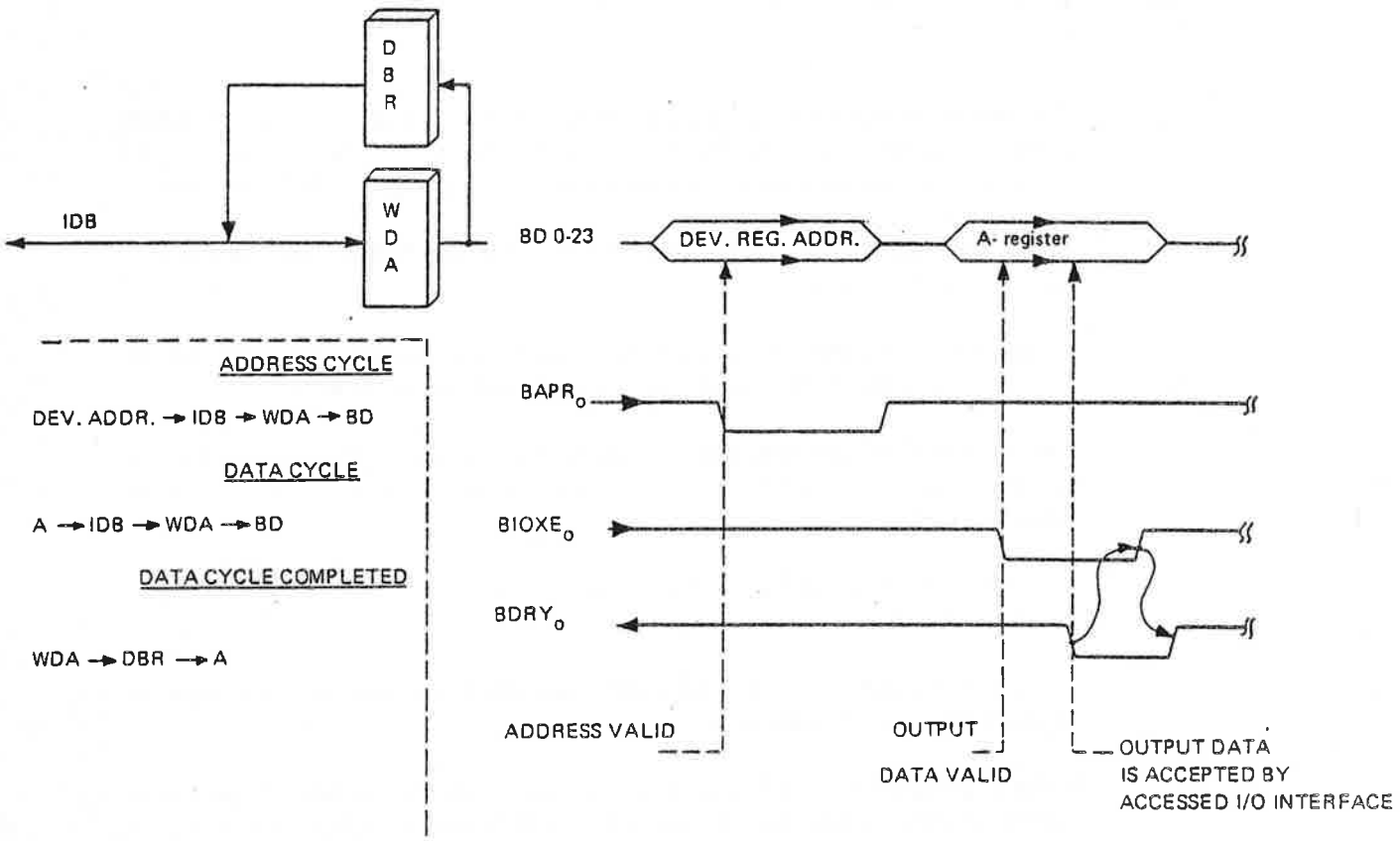


Figure III.2.4: IOX/IOXT Output Operation via the NORD-100 Bus

Upon receiving the allocation acknowledge signal, the handshake logic enables WDA onto the NORD-100 bus BD lines combined with the address strobe $BAPR_0$.

The CPU bus handshake logic holds the addresses about 50 ns after leading edge of $BAPR_0$ before it continues with the data cycle.

The microprogram now moves the A register, via IDB, to WDA. The bus handshake logic passes WDA to the BD lines.

When the data (the A register) is valid on the BD lines, the handshake logic activates the signal $BIOXE_0$.

Upon receiving BIOXE active, all I/O interfaces "look" at the device register address presented in the address cycle to see if it is equal with their own address.

The interface that finds equal address strobes the content of the BD lines (the A register) into the specified device register. The interface then activates the signal $BDRY_0$ to signify "data accepted". In other words, the transfer is completed.

The handshake logic uses the leading edge of $BDRY_0$ to strobe the content of the BD lines (still the A register) into DBR and restarts the microprogram. The microprogram terminates the IOX/IOXT execution by moving DBR to the A register. That is, the A register is left unchanged after an IOX/IOXT output transfer.

As indicated in Figure III.2.4, the handshake logic turns BIOXE off upon reception of BDRY. BIOXE off implies that the accessed I/O interface turns $BDRY_0$ off. That is, the NORD-100 bus is released and ready for the next bus cycle.

Case 2: IOX/IOXT Input Transfer

Refer to Figure III.2.5.

The IOX/IOXT address cycle is independent of the transfer direction, i.e., it follows the same scheme both for input and output transfers.

The data cycle also starts equal since neither the microprogram nor the CPU bus handshake logic knows what the direction actually is.

The IOX/IOXT transfer direction is selected by the I/O interface that finds "equal" after the reception of BIOXE.

If the accessed interface finds that the specified I/O register to be exchanged is an input register, the interface activates the signal $BINPUT_0$.

The bus handshake logic reacts on the active $BINPUT_0$ signal by closing the output from the WDA buffer and then activating the signal BINACK.

The signal BINACK (BINput ACKnowledge) is a direct feedback to the interface that generated BINPUT. BINACK means theat the BD lines are free to accept data from the interface. Thus, at the reception of $BINACK_0$ active, the interface enables the accessed input register onto the BD lines. When the content of the input register is valid on the NORD-100 bus, the interface generates the $BDRY_0$ signal.

As for output, the bus handshake logic uses the leading edge of $BDRY_0$ to strobe the content of the BD lines into DBR and restarts the microprogram.

The microprogram moves DBR, which now contains a specified I/O register, to the A register.

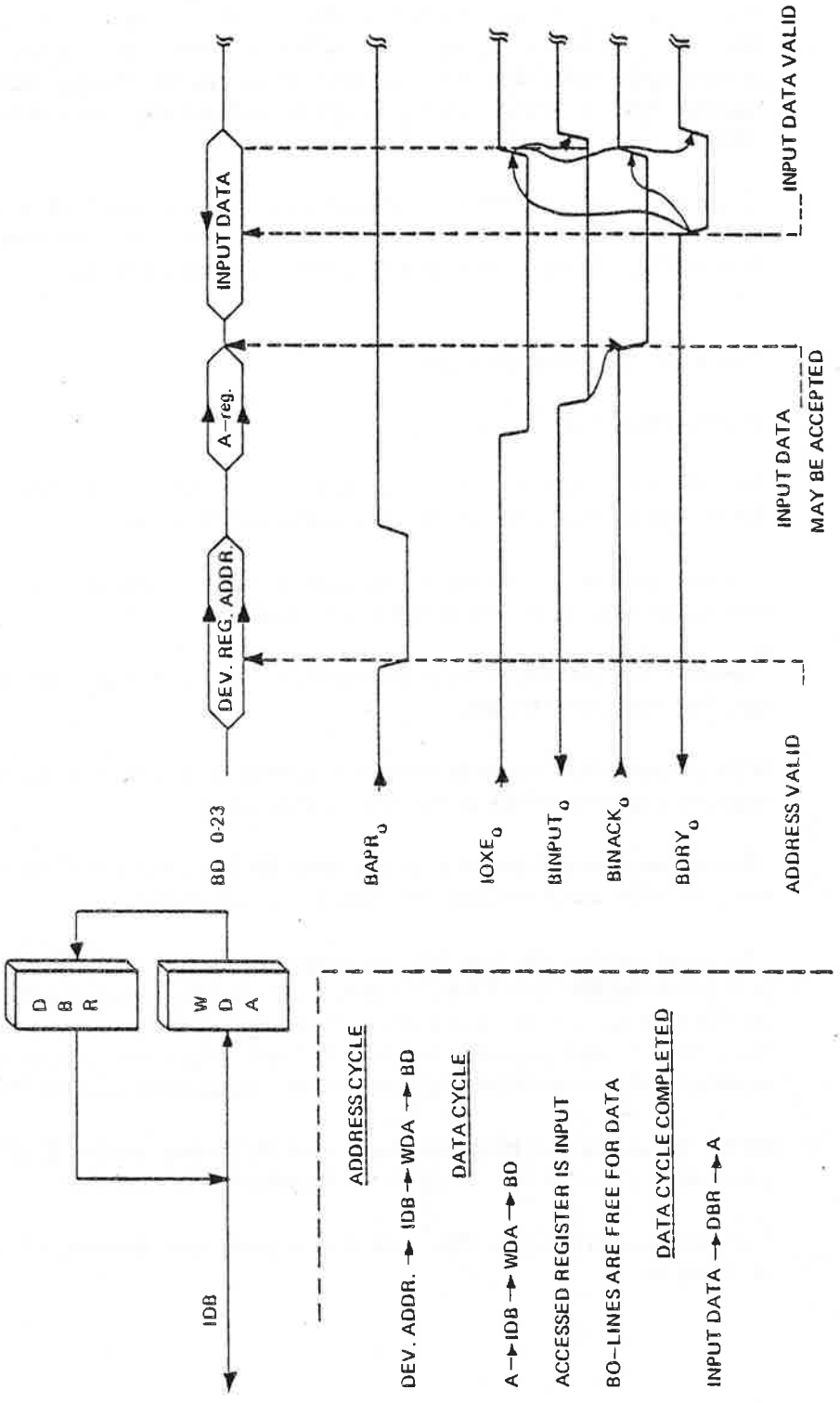


Figure II.2.5: IOX/IOXT Input Operation via the NORD-100 Bus

III.2.5 *IOX/IOXT EXECUTION AND THE I/O INTERFACES*

All I/O interfaces, both PIO and DMA, are controlled by program, i.e., accessed by IOX/IOXT instructions.

Thus, each I/O interface has a bus control logic specially dedicated to handle IOX/IOXT access to the interface.

The bus control logic is designed independent of the peripheral connected to the interface. This implies that all peripherals appear equal to the NORD-100 bus and could be handled equally by the CPU bus handshake logic.

On a PIO interface module, the bus control logic is standarized to handle up to four different devices accessed to the NORD-100 bus.

Refer to Figure III.2.6.

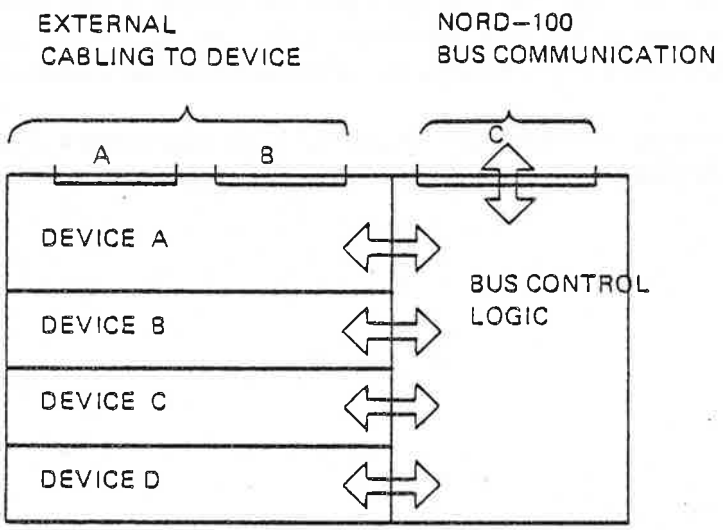


Figure III.2.6: PIO Interface Organization

III.2.6 *PIO INTERFACE MODULE ORGANIZATION*

The actual number of devices controlled by one interface module is limited to the physical space required for the peripheral device dependent control logic.

A DMA controller requires at least one module. Thus, it has no meaning talking about a standard part controlling several DMA controllers.

However, a DMA controller, of course, has an IOX/IOXT bus control logic with the same functions as for PIO interfaces.

The description of an I/O interface bus control logic is therefore applicable to both PIO and DMA interfaces.

The main function of the I/O interface's bus control logic is to handle the handshake with the CPU bus handshake logic during IOX/IOXT execution.

The address and data flow and the associated control signals invovled in this handshake have already been described in timing diagrams. Thus, the functions and the sequence in which the different control signals appear should be known.

The following text and illustration (Figure III.2.7) is based on this knowledge and is meant as an introduction to the I/O interface's bus control logic.

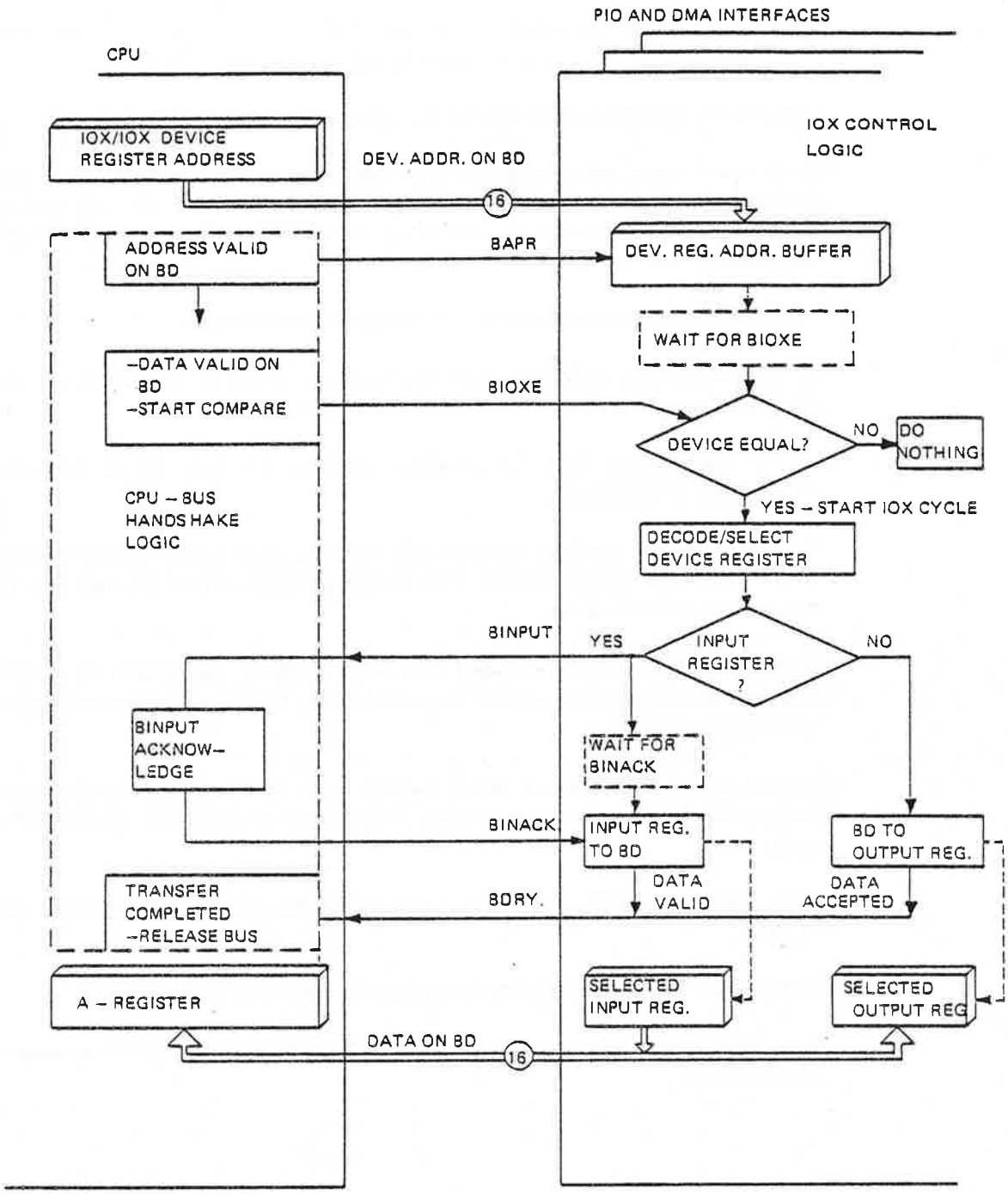


Figure III.2.7: CPU Bus Handshake Logic and I/O Interface Bus Control Logic Interaction

The first information presented on the NORD-100 bus during IOX/IOXT execution is the device register address combined with the address strobe $BAPR_0$.

This information is available to all modules connected to the bus simultaneously.

On all I/O modules, the leading edge of $BAPR_0$ is used as a strobe to buffer up the address presented on the BD lines. Note that the I/O modules at this stage do not know if the address is meant for memory or an I/O interface. Thus, the I/O interfaces have to wait for more information to proceed.

The $BIOXE_0$ signal active is what the I/O interfaces are waiting for.

At the leading edge of $BIOXE_0$, each I/O interface compares the device number part of the buffered address with its own device number.

The I/O interface that finds "device equal" starts an IOX cycle on the interface. Other interfaces do nothing.

The "device equal" interface starts the IOX cycle by decoding the register number bits in the device register address. This decoding selects one of the registers on the interface to be accessed.

If the device register to be accessed is an output register, the content of the BD lines, i.e., the A register, is written into the register. $BDRY_0$ is activated to signify "data accepted".

In cases where the accessed device register is an input register, the interface requests the BD lines for input of data. This is done by activating the $BINPUT_0$ signal.

Upon receiving $BINACK_0$, the interface enables the selected input register onto the BD lines.

$BDRY_0$ is now generated to signify the presence of valid data on the BD lines.

In the next section we will look at how the device identification logic is implemented in hardware.

III.2.7 *HARDWARE IMPLEMENTATION OF THE DEVICE IDENTIFICATION LOGIC*

The block diagrams given in Figure III.2.9 and III.2.10 illustrate how the device identification logic is implemented in hardware. As seen, both figures are very similar. Figure III.2.9 shows how the hardware is organized on a module controlling one peripheral type (DMA controller) while Figure III.2.10 shows the organization on a module controlling several peripheral types (PIO interfaces).

The device identification logic is based on a compare circuit.

The device register address issued during IOX execution, is in this circuit compared with the I/O interfaces' device numbers.

Device numbers relevant to the different I/O interfaces resides in "device number PROM's" (Programmable Read Only Memories).

There is one PROM for each class of peripherals controlled by equal hardware. The actual device number selected on an interface depends on the selected PROM location.

A PROM location is selected by a PROM address which is set by a thumbwheel. Thus, the device number of an interface is easily changed by turning the thumbwheel.

On I/O modules controlling more than one peripheral type with different device numbers, one seperate PROM, thumbwheel and comparator is available for each peripheral.

In order to make it easier to understand the device identification logic, it could be convenient to see how the hardware "looks" at the presented device register address.

On a two channel interface (i.e., three register number bits) the hardware is designed to handle the device register address format given in Figure III.2.8.

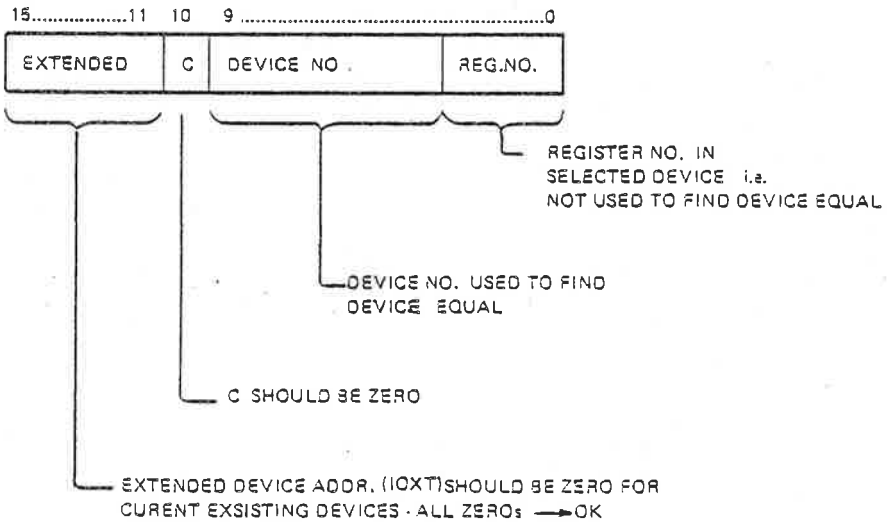


Figure III.2.8: IOX/IOXT Device Register Address Format as it Appears to Hardware

As indicated in the block diagram (Figure III.2.9) the extended address bits (bit 11-15) is passed through an "all zero" test.

If this test turns out in "ok" (all zero) *and* BIOXE goes active, the IOX device number is compared with interface's device numbers.

One and only one interface should now find "device equal". Device equal on an interface enables decoding of the register number field of the device register address. As indicated in the figures III.2.9 and III.2.10, this decoding may be performed either "sentralized" or in the device dependent part on single peripheral or multiperipheral controllers respectively.

The decoding is performed physically by a 3 to 8 line decoder. A given bit combination input to the decoder gives one output line active.

That is, the number specified in the register number field gives one line active which points out the register that is to be accessed.

The output from the decoder is used to clock the content of the BD lines into a device output register.

If an input register is specified, the active output from the decoder *and* BINACK true, enables the device input register onto the BD lines.

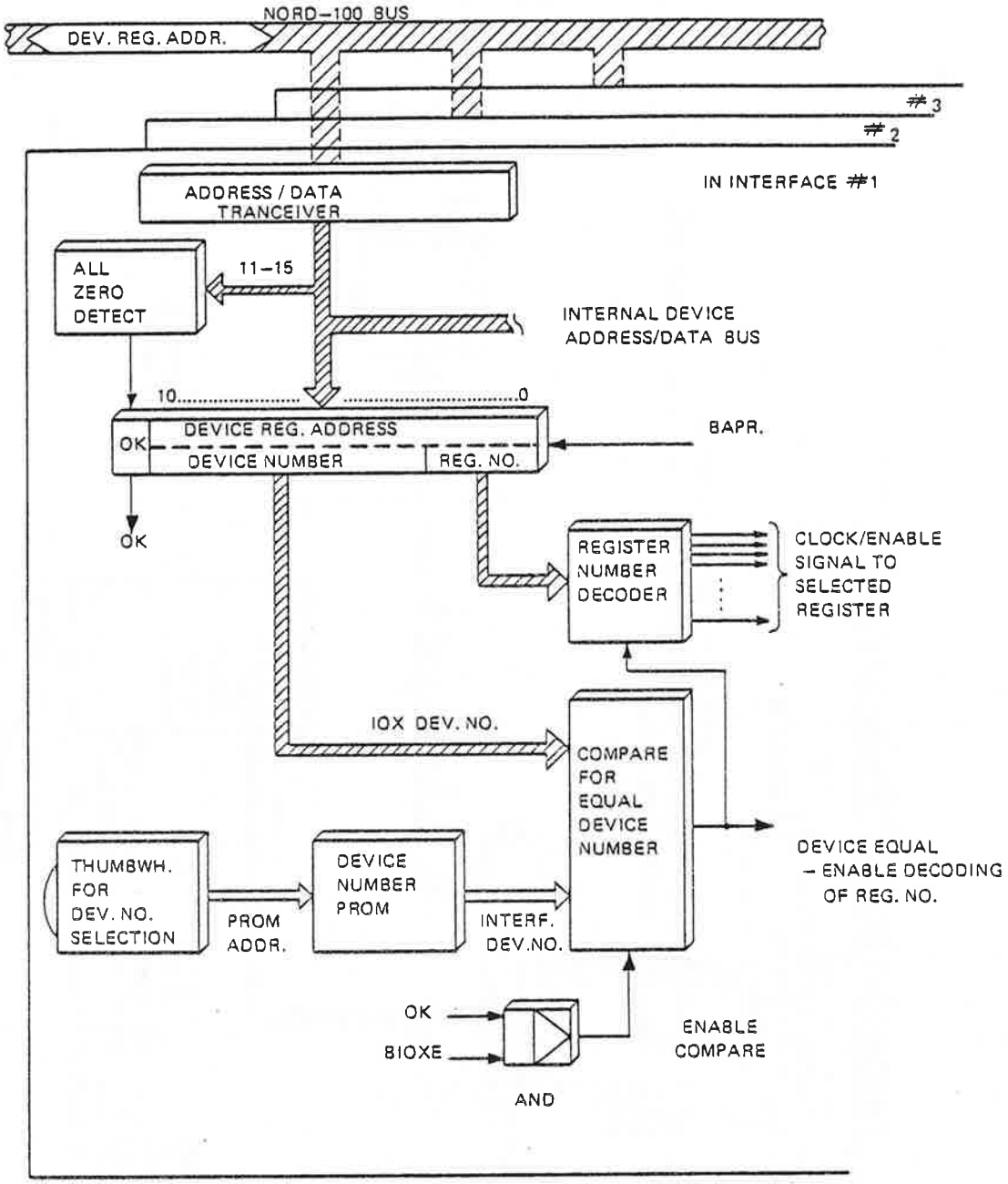


Figure III.2.9: Device Identification Logic on a Single Device I/O Interface (DMA Controllers)

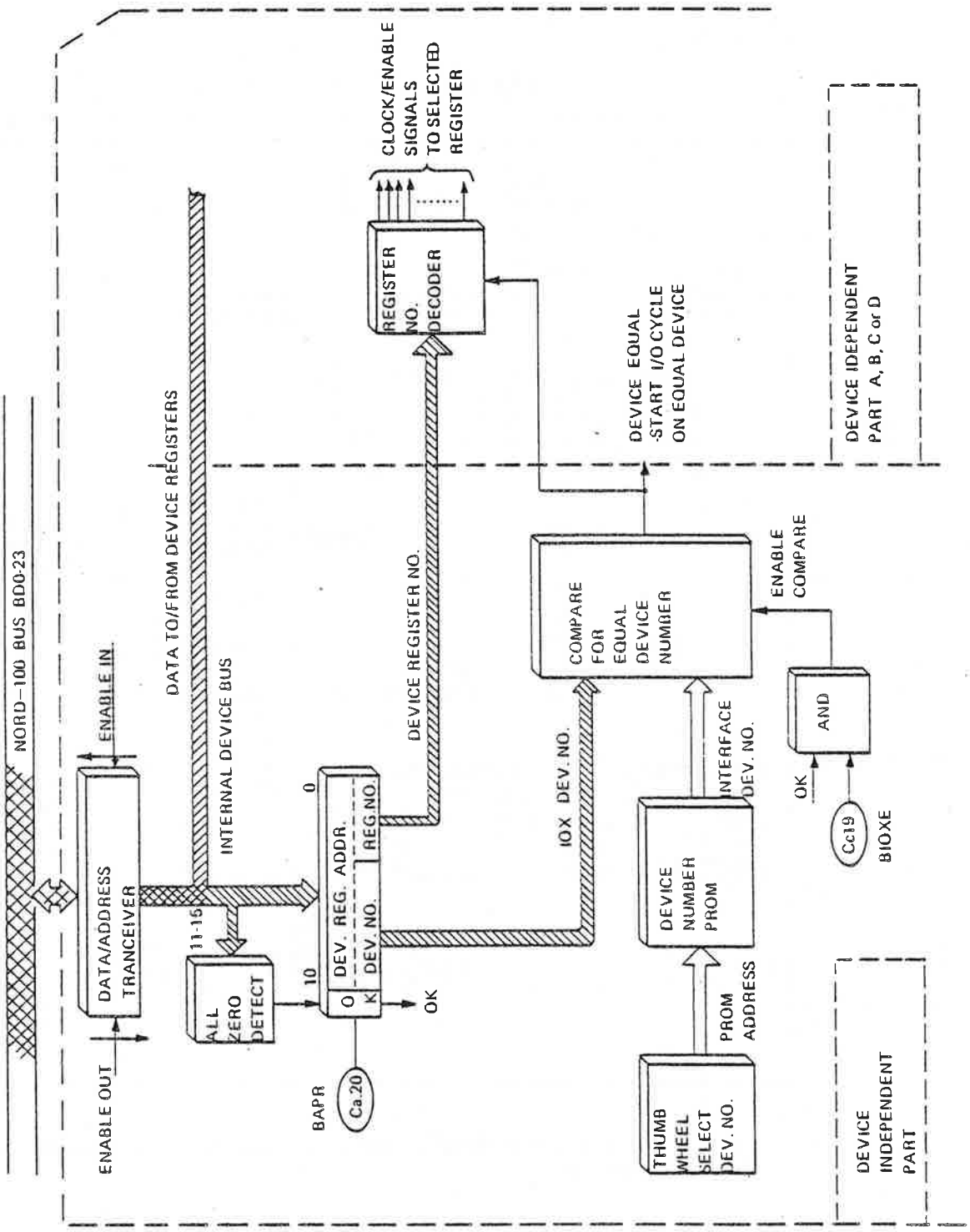


Figure III.2.10: Device Identification Logic on Multi Device I/O Interfaces (PIO Interfaces)

IV IDENTIFICATION OF INTERRUPTING I/O INTERFACES — EXECUTION OF THE IDENT PLxx INSTRUCTION

IV.1 INTRODUCTION

The event causing the IDENT PLxx instruction to be executed is an interrupt on one of the levels 10, 11, 12 or 13.

The background for interrupt generation on the I/O interfaces should be known from other parts of this manual.

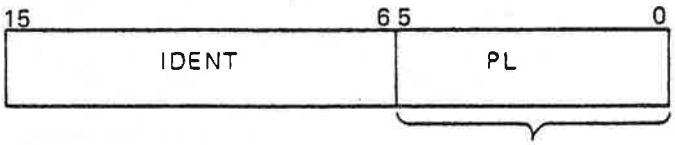
The need for and the function of the IDENT PLxx instruction has also been described:

- Its purpose is to search for an existing interrupt on a specified level (PLxx) and return the ident code from the first found device with interrupt on specified level to the CPU A register.

This section is intended to give a more detailed description of how the IDENT PLxx instruction is executed. That is, how it appears to the NORD-100 CPU, the NORD-100 bus and the I/O interfaces.

IV.2 EXECUTION OF IDENT PLxx

The format of the IDENT PLxx instruction entered for execution should be as shown below.



Since the IDENT PLxx is a privileged instruction the entry point generation in the microprogram is restricted in the same way as for IOX/IOXT execution. Non-privileged use is detected and error message is given.

This section is intended to give a functional description of the microprogrammed routine after the IDENT PLxx instruction is successfully entered.

IV.2.1 *IDENT MICROPROGRAM OPERATION*

The search for interrupts that is performed during IDENT PLxx execution is normally done in the NORD-100 bus.

However, as for the IOX/IOXT instruction, there are exceptions.

Terminal number 1 (device no. = 300_8) is physically implemented on the CPU module. Thus, interrupts (level 10 or 12) from this terminal are not detected by searching in the NORD-100 bus.

The real-time clock (interrupt level 13) is also designed as an integrated part of the CPU.

That is, if interrupts are detected on the levels 10, 12 or 13 it might be that the search should *not* go via the NORD-100 bus.

This has to be investigated by the micropgoram and it is done as illustrated in Figure IV.2.1.

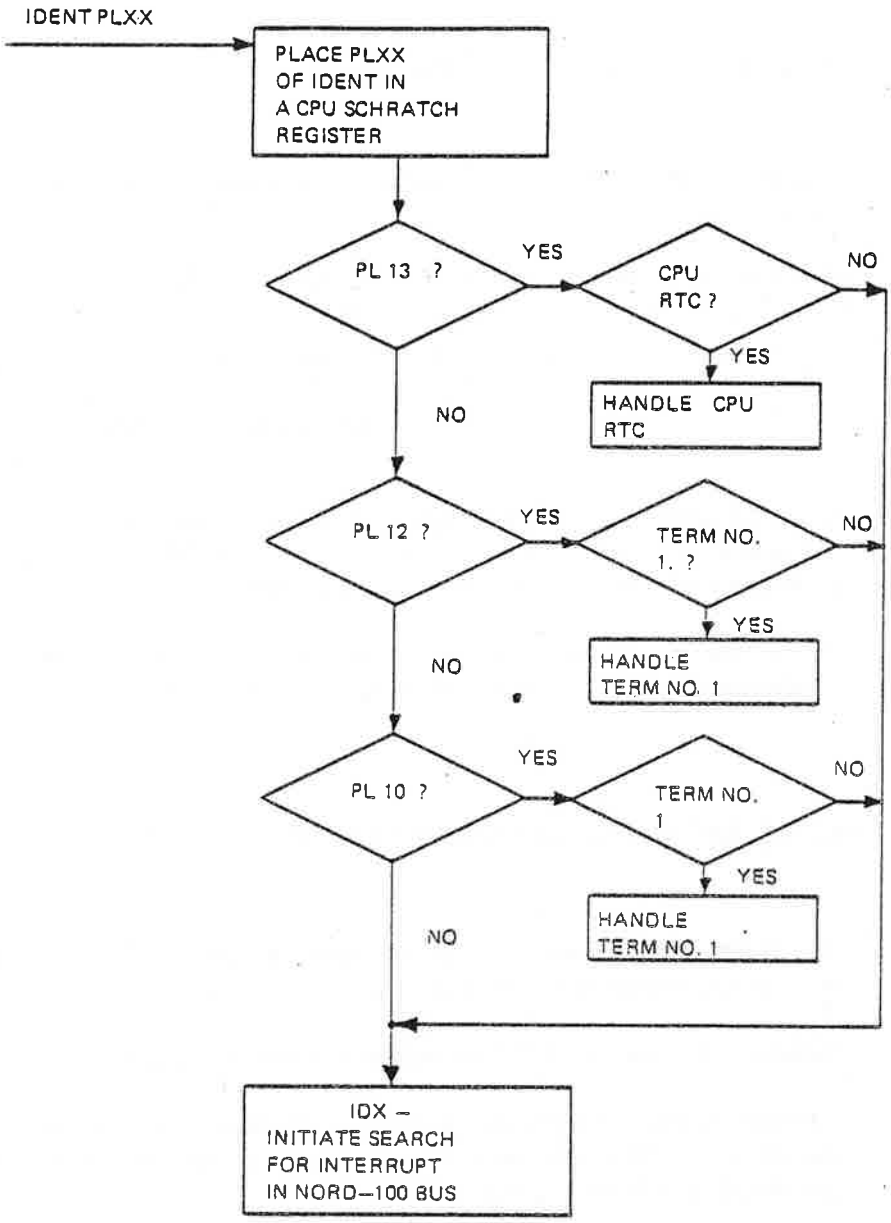


Figure IV.2.1: Microprogram Separation of Internal (CPU) or External Interrupts on Level 10, 12 or 13

In the following description of the IDENT PLxx instruction it is assumed that the search for interrupt has to be performed in the NORD-100 bus.

The microroutine designed to handle the IDENT PLxx search on the NORD-100 bus is principally equal to the IOX/IOXT Bus Transfer routine. The only difference is that in IDENT execution, the microprogram tells the CPU bus handshake control logic to perform an interrupt search instead of an IOX/IOXT information exchange.

Thus, the interaction between the microprogram and CPU bus control logic, with some minor adjustments, follows the same scheme as for IOX/IOXT execution.

Refer to Figure IV.2.2 for information relevant to IDENT PLxx execution.

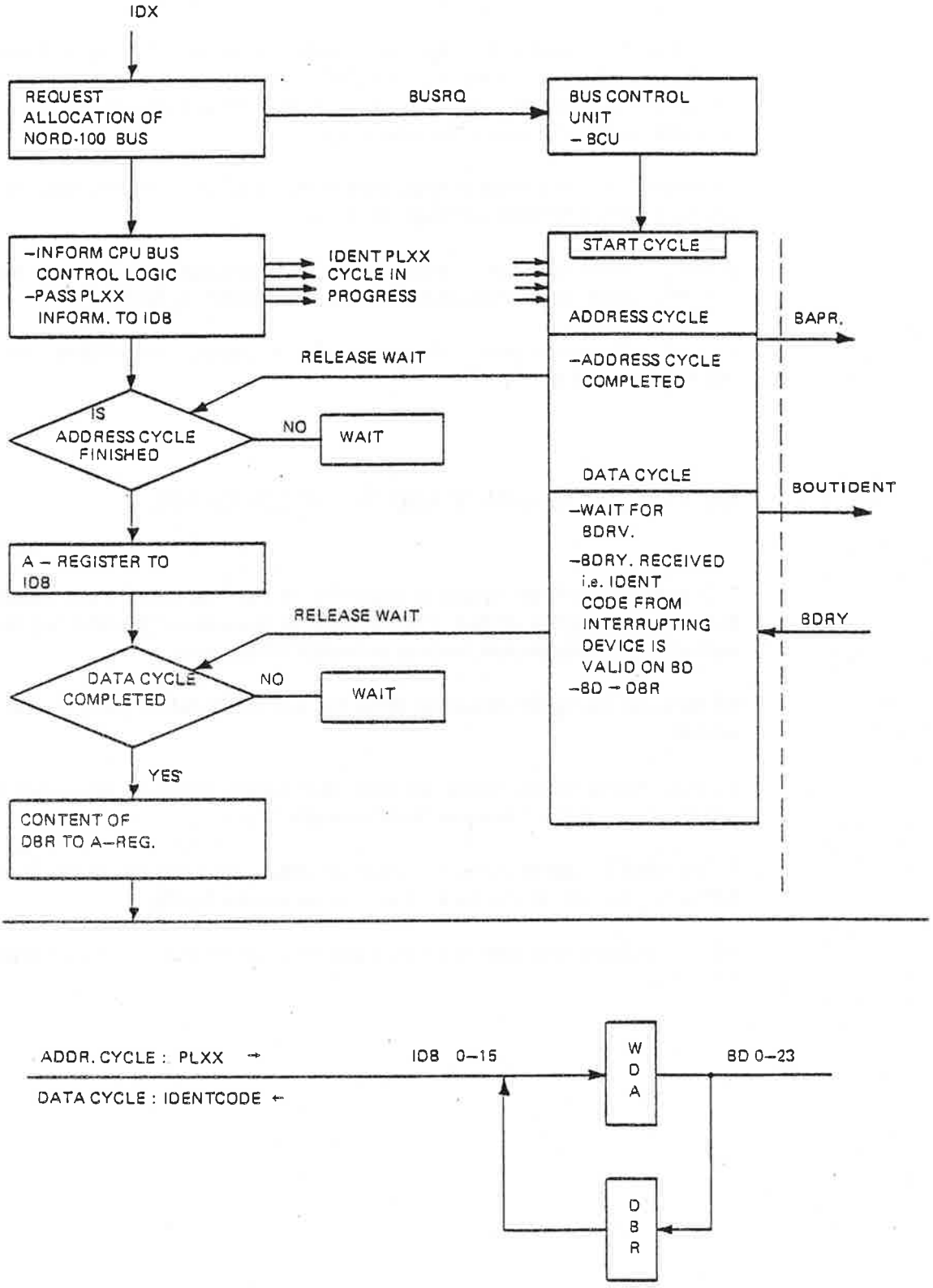


Figure IV.2.2: IDENT Micro Routine and CPU — Bus Handshake Logic Interaction

Figure IV.2.2: IDENT Micro Routine and CPU — Bus Handshake Logic Interaction

As indicated in Figure IV.2.2, the microprogram generates a bus allocation request (BUSRQ) to the BCU. In addition, the CPU bus handshake logic is informed that the cycle in progress is an IDENT cycle. The level specification (PLxx) is enabled onto IDB, ready to be used in the address cycle.

The IDENT PLxx exeuction is terminated at the reception of $BDRY_0$ from the first interface found with interrupt on specified level.

BDRY is, by the CPU bus handshake logic, used to clock the content of the BD lines (the interrupting interface's ident code) into the DBR register.

The microprogram moves DBR to the CPU A register and initiates the next machine instruction.

IV.2.2 *IDENT PLxx EXECUTION AND THE NORD-100 BUS*

This section is intended to give a description of how the IDENT PLxx instruction appears to the NORD-100 bus. This includes the operation of the CPU bus control logic and the control signals involved during IDENT execution.

Background for signal responses from the I/O interfaces is given in the next section.

As other NORD-100 bus cycles, an IDENT bus cycle is started by an address cycle and terminated with a data cycle. Refer to Figure IV.2.3.

In the IDENT address cycle, the level on which the interrupt search is to be performed, is presented on the BD lines, combined with $BAPR_0$.

All I/O interfaces having interrupt on the specified level turns on a "flag" to signify this.

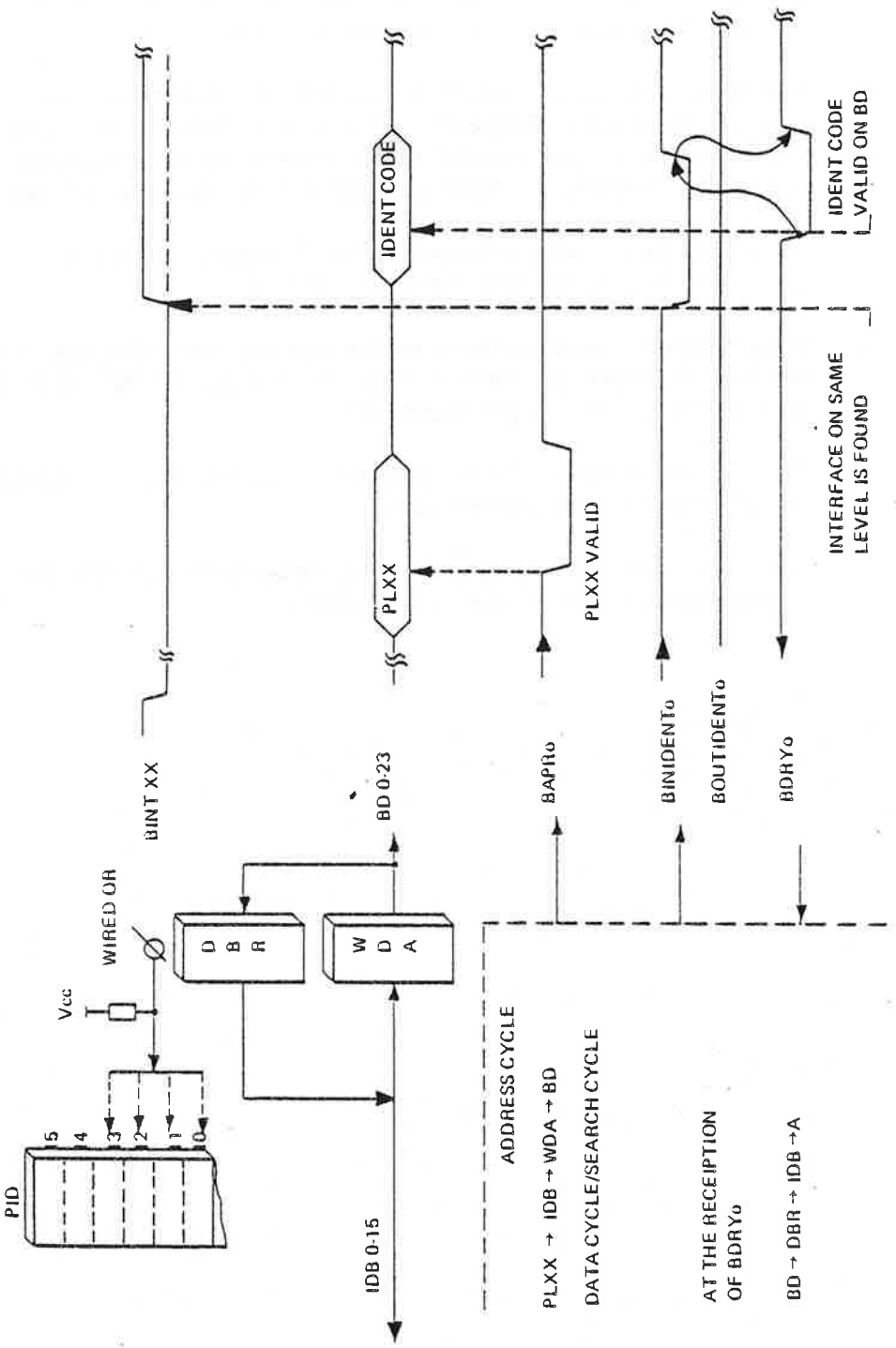


Figure IV.2.3: IDENT PLxx Interrupt Search via the NORD-100 Bus

Although several I/O interfaces, at the same time, may have interrupts waiting for service on the specified level, only one may be handled at a time.

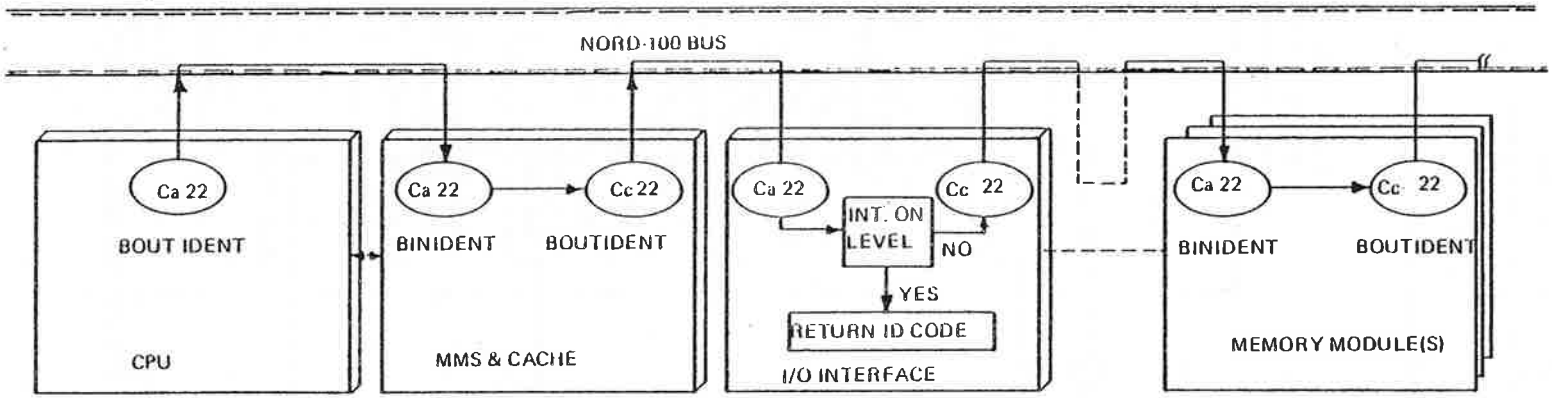
This potential problem is solved by the CPU bus handshake logic issuing an interrupt search signal (BINIDENT)* which is daisy chained in the NORD-100 bus. That is, the search signal is passed form one NORD-100 bus slot position (n) to the next (n + 1) via the module placed in position n. This is illustrated in Figure IV.2.4.

* The search signal is called BINIDENT when it is input to a module and BOUTIDENT when it is leaving a module.

If the BINIDENT signal reaches a module with the "flag" (interrupt on specified level) set, the search is stopped. The interface that stopped the search returns its ident code to the CPU, combined with $BDRY_0$.

Modules with no interrupt "flag" set should, with the smallest possible delay, pass BINIDENT further to the next slot position.

Thus, this daisy chain establishes a sequential priority scheme between I/O interfaces generating interrupt on the same level.



NOTE 1: EMPTY SLOT POSITIONS BREAK THE BINIDENT/BOUTIDENT SEARCH CHAIN

NOTE 2: MODULE PLACED NEAREST THE CPU HAS HIGHEST PRIORITY WITHIN ONE LEVEL

NOTE 3: MODULES NOT USING THE INTERRUPT SEARCH CHAIN SHOULD STRAP BINIDENT TO BOUTIDENT

Figure IV.2.4: The BINIDENT/BOUTIDENT Interrupt Identification Search Chain

IV.2.3 *IDENT PLxx EXECUTION AND THE I/O INTERFACES*

All I/O interfaces have an IDENT control logic. The IDENT control logic is normally implemented as an integrated part of the device independent IOX/IOXT control logic.

Figure IV.2.5 illustrates the basic functions of the control logic. Refer to the illustration.

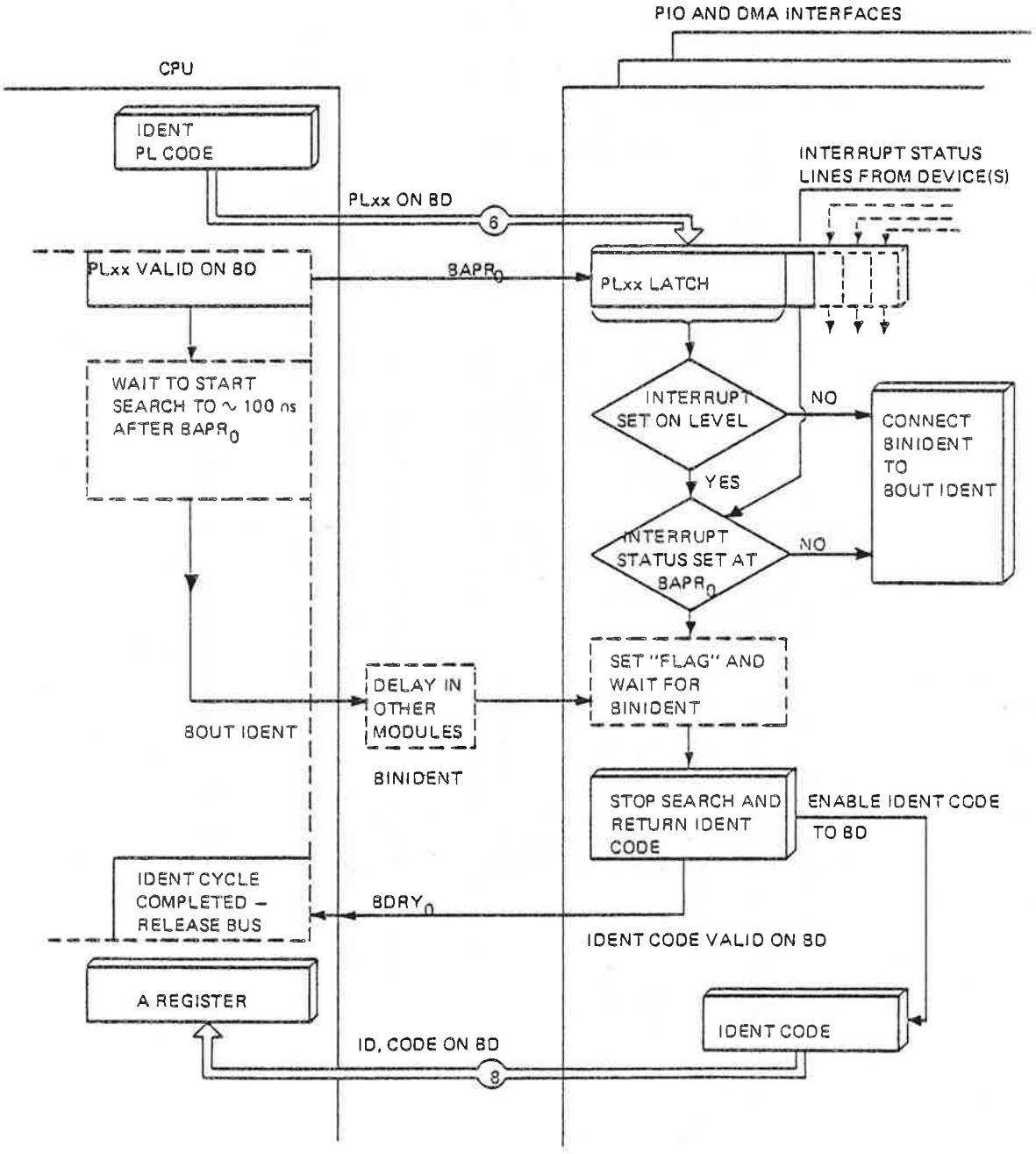


Figure IV.2.5: CPU — Bus Handshake Logic and I/O Interfaces Bus Control Logic

In the IDENT address cycle, all I/O modules, simultaneously, are presented PLxx on the BD lines, combined with $BAPR_0$.

$BAPR_0$ is used to buffer up the content (PLxx) of the BD lines.

In addition, the interrupt status on the interfaces are "frozen".

I/O interfaces which have interrupt on specified level *and* interrrupt status set at $BAPR_0$ are allowed to be set "flag" (interrupt on specified level).

This is done to ensure a stabilized test condition for the BINIDENT signal.

As indiciated, the CPU bus handshake logic, from leading edge of $BAPR_0$, gives the I/O interfaces about 100 ns to complete this procedure. The I/O interfaces should then be ready to either pass BINIDENT to BOUTIDENT or stop the search and return the IDENT CODE.

IV.2.4 *HARDWARE IMPLEMENTATION OF THE IDENT CONTROL LOGIC ON PIO AND DMA INTERFACES*

The block diagrams given in Figures IV.2.6 and IV.2.7 are used as illustrations to the text in this section.

Figure IV.2.6 illustrates the IDENT control logic on a single device controller interface (DMA controllers). Figure IV.2.7 illustrates how it is implemented on multi device interfaces (several PIO controllers on one module).

The implementation is principally and functionally equal for both single and multi device controllers.

One may look at one part, controlling one device, on a multi device interface and find it equal to the logic on a single device interface. Thus, the description of a single device interface is applicable to one part of a multi device interface. Refer to Figure IV.2.6.

The PLxx information presented with $BAPR_0$ during IDENT execution flows into the interfaces where it is buffered. As indicated, the device interrupt status line(s) is buffered at $BAPR_0$.

It is worth noting that "something" on BD and $BAPR_0$ is equal to the I/O interfaces both for IOX/IOXT and IDENT execution. Thus, the IOX control logic prepres the bufferred content of BD (in this case PLxx) for device number checking.

The IDENT logic, in parallel, investigates if the interface has any interrupt on the specified level. If there is, the interface blocks generation of BOUTIDENT in case BINIDENT arrives. This is done by the signal STOPIDENT, which is an AND function between interrupt on specified level and the interrupt status at $BAPR_0$. STOPIDENT clamps $BOUTIDENT_0$, which is active low, to a high level. In this case (IDENT execution), BINIDENT arrives. BINIDENT and STOPIDENT gives the signal IDENT internally on the activated interface.

The IDENT signal is used as the least significant address bit to the device number PROM. Most signficant bits are the fixed thumbwheel outputs. That is, a new PROM location, containing the interface's ident code, is accessed and enabled onto the NORD-100 bus BD lines.

After an appropriate "set up time", $BDRY_0$ is generated to signify presence of valid IDENT CODE on the BD lines.

Now refer to Figure IV.2.7. As indicated, the multi device interface is an extended single device interface.

Each device (A, B, ...) has its own device number/ident code PROM. The ident code is returned from the different devices as described for the single device interface.

The reason for some extra logic is that it might be simultaneous interrupts on the specified level within the interface.

One device within a multi device interface is selected by a priority circuitry. The priority circuitry selects one out of a maximum of four devices to be served by the current IDENT execution.

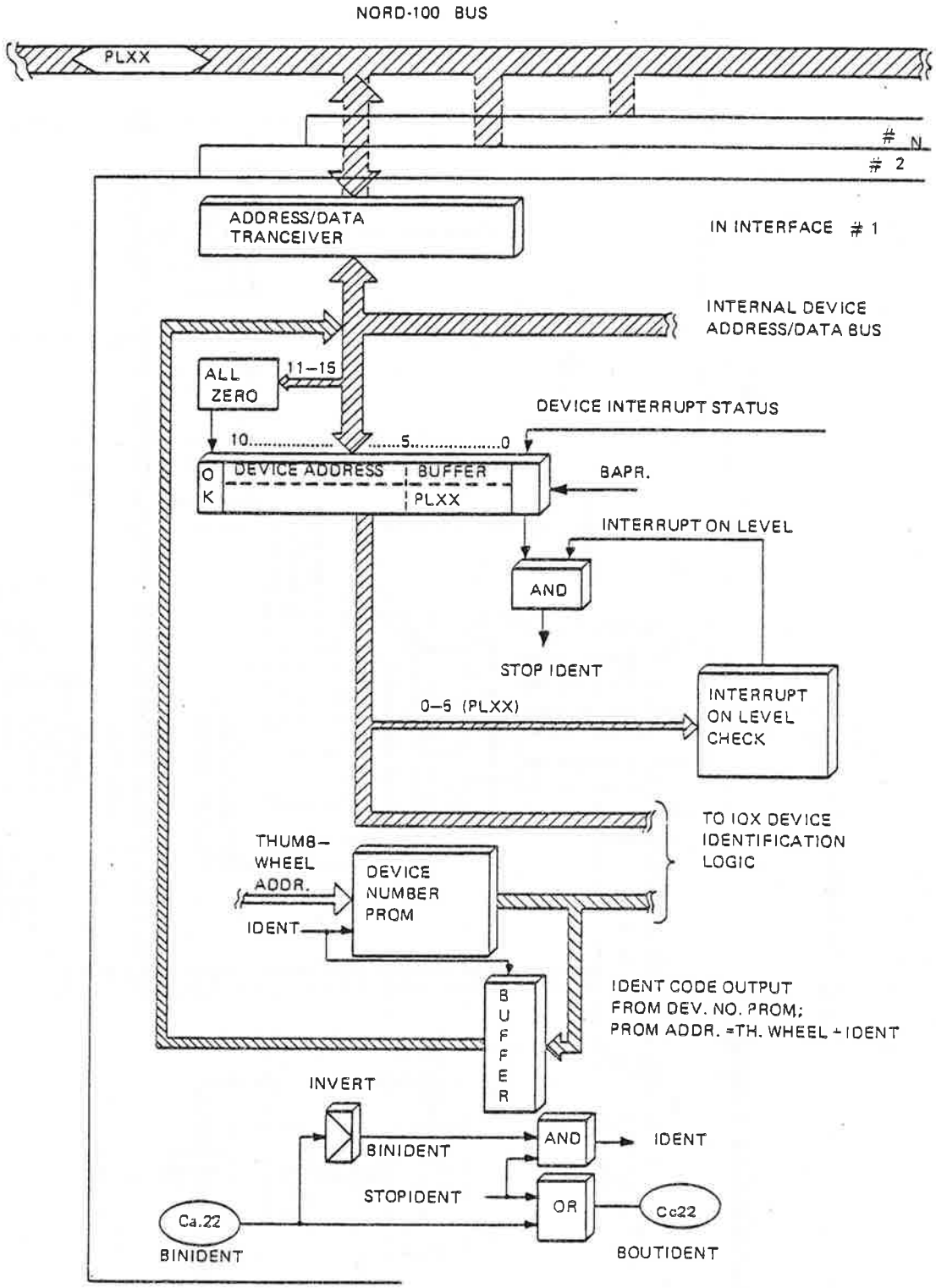


Figure IV.2.6: IDENT Control Logic Implemented on Single Device Controllers (DMA Controllers)

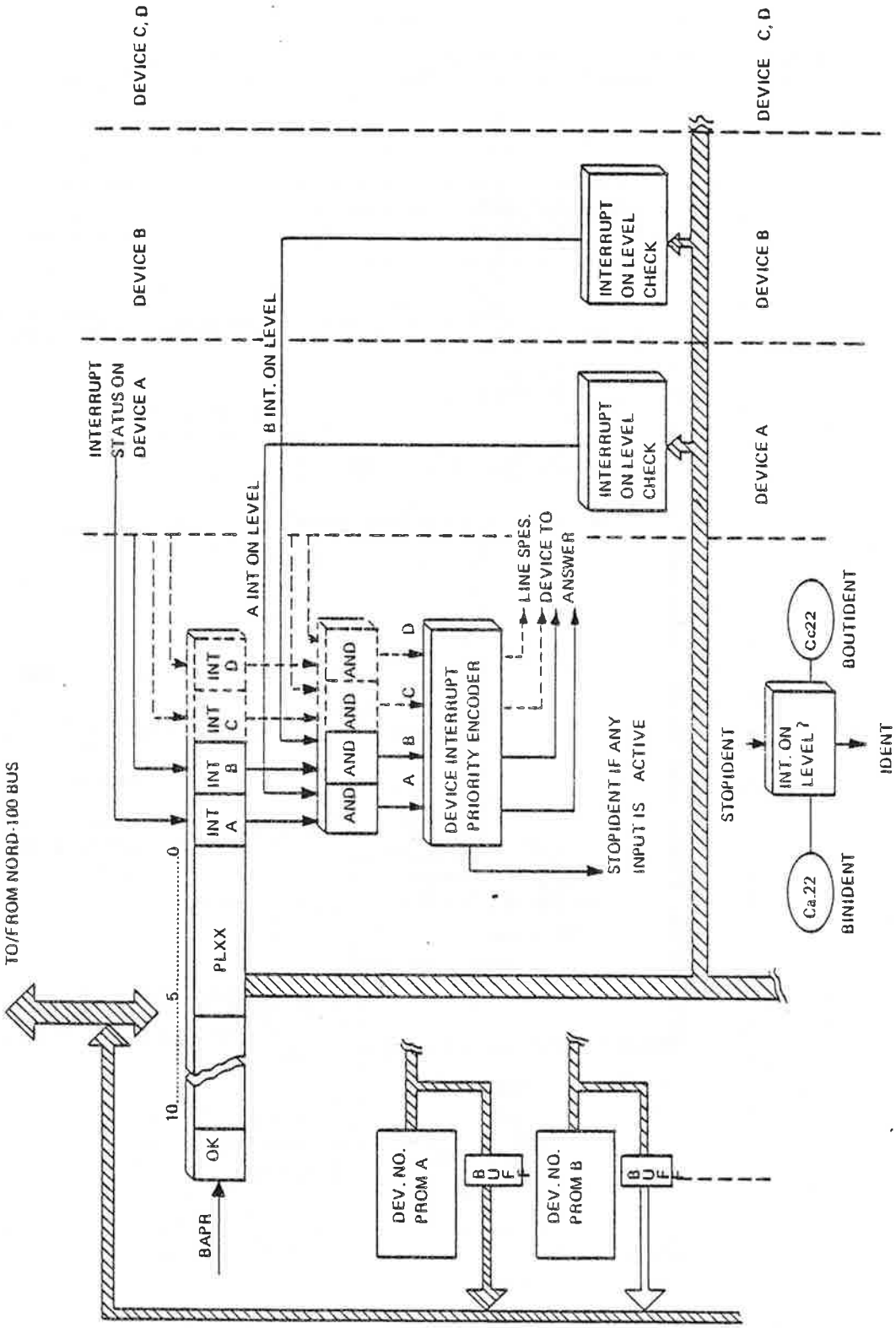


Figure IV.2.7: IDENT Control Logic Implemented on Multi Device Controllers (PIO Interfaces)

V DIRECT MEMORY ACCESS (DMA) INFORMATION EXCHANGE VIA THE NORD-100 BUS

V.1 GENERAL

This section is intended to give a functional description of the information exchange handled directly between an I/O interface and the NORD-100 memory system, referred to as a DMA transfer.

I/O interfaces capable of handling such transfers are, as previously mentioned, generally referred to as DMA controllers.

A DMA controller is initialized by program, i.e., by means of IOX/IOXT instructions. The functions of the program handling a DMA controller varies from one device to another and is not discussed here. (A general description is given in Sections I.3.5 and I.3.7.)

What is to be discussed here is the event of one word of information being exchanged directly between memory and a DMA controller.

Such a transfer is based on a defined handshake mechanism against the Bus Control Unit (BCU) and the memory system. This handshake mechanism is, of course, device independent.

Thus, the following description applies to all I/O interfaces capable of handling direct memory access (DMA) transfers.

V.2 THE DMA REQUEST GENERATION

A DMA controller may need access to memory for one of two reasons:

- more information from memory is needed (for output)
- information ready to be written to memory is available in the controller

That is, either a memory read or write operation.

Common for both cases is that the NORD-100 bus is needed for the exchange. Therefore, if any of the two above conditions occur on a DMA controller, the controller will request allocation of the NORD-100 bus.

The bus allocation request from a DMA controller is referred to as a "DMA request".

The DMA request, named BREQ (Bus REQuest) is a line in the NORD-100 bus input to the Bus Control Unit (BCU) priority arbiter.

BREQ is a shared (wired or) request line among all DMA controllers and may be driven from any slot position in the NORD-100 bus.

V.3 DMA REQUESTS AND THE BUS CONTROL UNIT (BCU) – THE BUS ALLOCATION ACKNOWLEDGE MECHANISM

After having generated the DMA request by activating $BREQ_0$, DMA controllers have to wait for an allocation acknowledge signal to initiate the transfer.

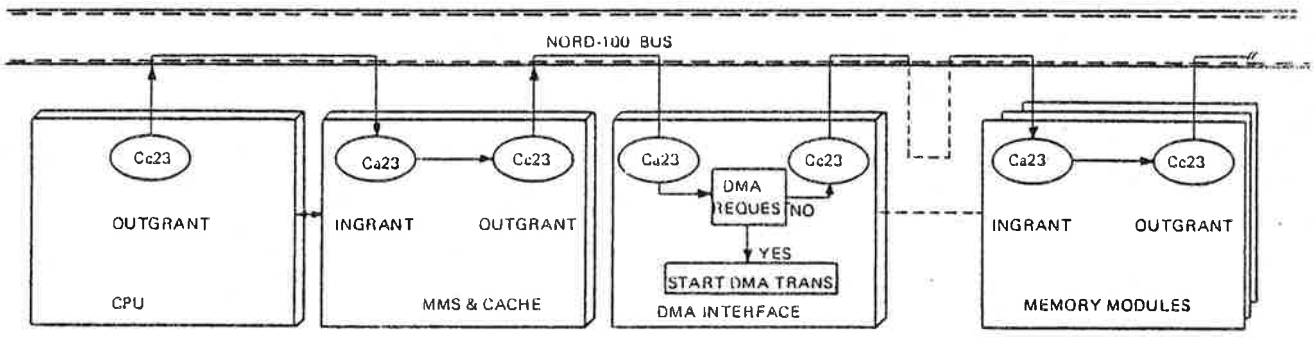
The allocation acknowledge signal, named OUTGRANT, is generated by the BCU when the NORD-100 bus is free for a DMA transfer.

Although several DMA controllers may have generated DMA requests simultaneously, only one should be allowed a transfer at a time.

In order to select only one DMA controller as "granted" at the time the acknowledge signal (OUTGRANT) is daisy chained in the NORD-100 bus. OUTGRANT is passed from one slot position (n) to the next (n + 1), now named INGRANT, via the module placed in position n (refer to the BINIDENT/BOUTIDENT search).

The first module found with DMA request set is granted and is allowed to make a one word exchange. Thus, the INGRANT/OUTGRANT daisy chain establishes a sequential priority scheme between the DMA controllers.

Refer to the illustration given in Figure V.3.1.



NOTE 1: EMPTY SLOT POSITIONS BREAK THE INGRANT/OUTGRANT SEARCH CHAIN

NOTE 2: MODULE PLACED NEAREST THE CPU HAS HIGHEST PRIORITY

NOTE 3: MODULES NOT USING THE DMA SEARCH CHAIN SHOULD STRAP INGRANT TO OUTGRANT

Figure V.3.1: The DMA Allocation Acknowledge Search Chain (INGRANT/OUTGRANT)

V.4 A DMA TRANSFER AND THE NORD-100 BUS

The activity on the NORD-100 bus during a DMA transfer may be divided up into two parts:

- first, the alloation request and allocation acknowledge procedure
- second, an ordinary memory reference cycle

The principles behind the first part which purpose is to establish one DMA controller as granted has already been explained. As indicated, this part is a handshake between the BCU and a requesting DMA controller. The second part, the transfer, is a handshake between the granted DMA controller and memory. BCU is passive during this part, but monitors the time limit of release.

In the following text and illustrations, the control signals relevant in the NORD-100 bus during a DMA transfer are described.

Due to some difference between a memory read (DMA output) and a memory write (DMA input) operation, these cases are covered as Case 1 and Case 2 respectively.

CASE 1: DMA OUTPUT (MEMORY READ)

Refer to the timing diagram in Figure V.4.1.

The sequence is started by a DMA controller issuing $BREQ_0$. As the request is granted the BCU issues the signal BMEM (Bus MEMory reference). $BMEM_0$ serves two important functions. It enables the memory system for further operations and "freezes" the DMA request status on the DMA controllers.

The "freezing" of the DMA request status is done to ensure a stabilized test condition for the INGRANT/OUTGRANT search chain. (It might happen that a DMA controller activates its request simultaneously with the reception of INGRANT.) That is, leading edge of $BMEM_0$ is the last chance for a DMA request to be served by the current INGRANT/OUTGRANT search.

A DMA controller which receives INGRANT active and had DMA request at BMEM is allowed to stop the search and establish itself as granted. The granted DMA controller starts the memory reference by presenting the memory address on the BD lines combined with BAPR.

The direction signal (BINPUT) not active tells the memory system that the started reference is a read operation.

As soon as the DMA controller has removed its memory address (this is done without feedback from memory), the signal $BDAP_0$ is activated. $BDAP_0$ (Bus DAta Present) means in a memory read cycle, that memory may enable data to the BD lines. This is also what the memory system is doing. As the data is valid, memory activates $BDRY_0$ (Bus Data ReadY). The DMA controller uses $BDRY_0$ to strobe the content of the BD lines (the read data) into a data buffer.

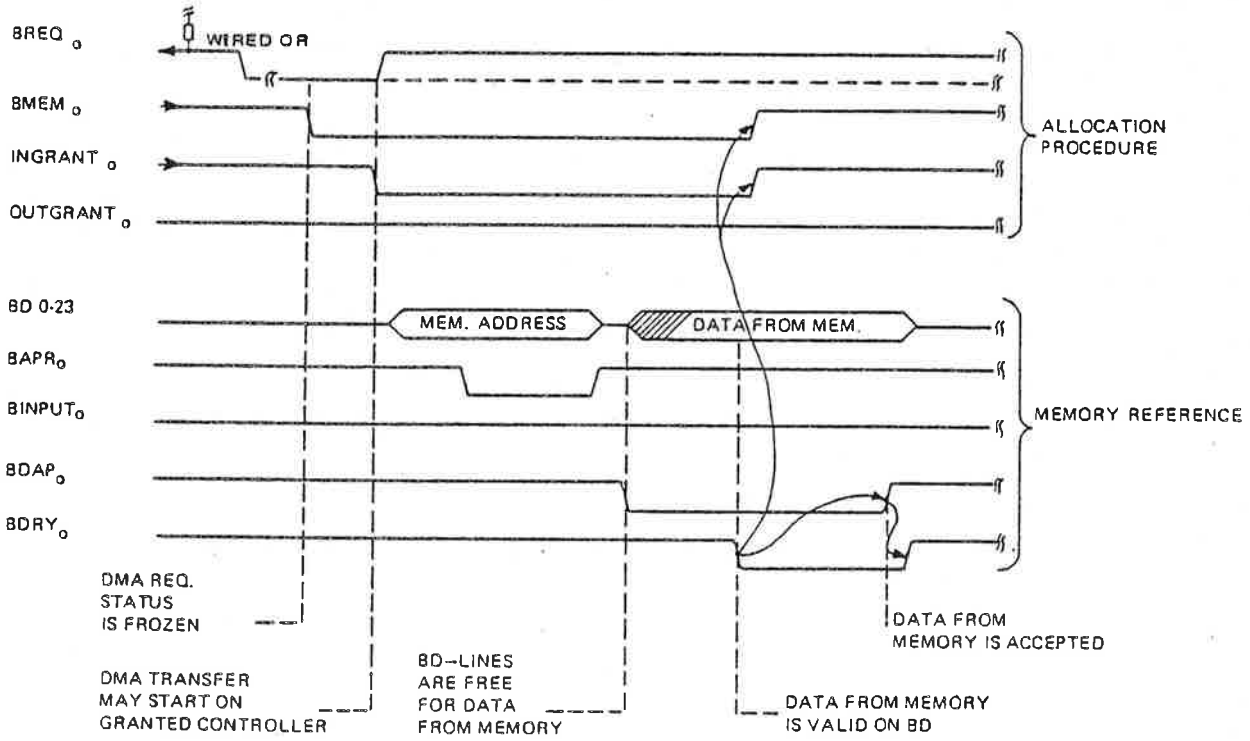


Figure V.4.1: DMA Output (Memory Read) Transfer via the NORD-100 Bus

In the BCU, $BDRY_0$ signifies transfer completed. The NORD-100 bus is released and next bus cycle may be initiated as $BDRY_0$ goes off (trailing edge off $BDRY_0$).

CASE 2: DMA INPUT (MEMORY WRITE)

A timing diagram illustrating a DMA input (memory write) transfer is given in Figure V.4.2.

The only difference from the DMA output transfer is the data cycle in the memory reference. Hence, it has no meaning to repeat the allocation procedure and the address cycle. Refer to Figure V.4.2.

The direction signal (BINPUT) true, informs the memory system that the started memory cycle is a write operation. That is, information is presented the memory system from the granted DMA controller.

The granted DMA controller presents data to the memory system as the address cycle is completed. In order for the memory system to know when the data is valid, the DMA controller combines the data with $BDAP_0$ (data present).

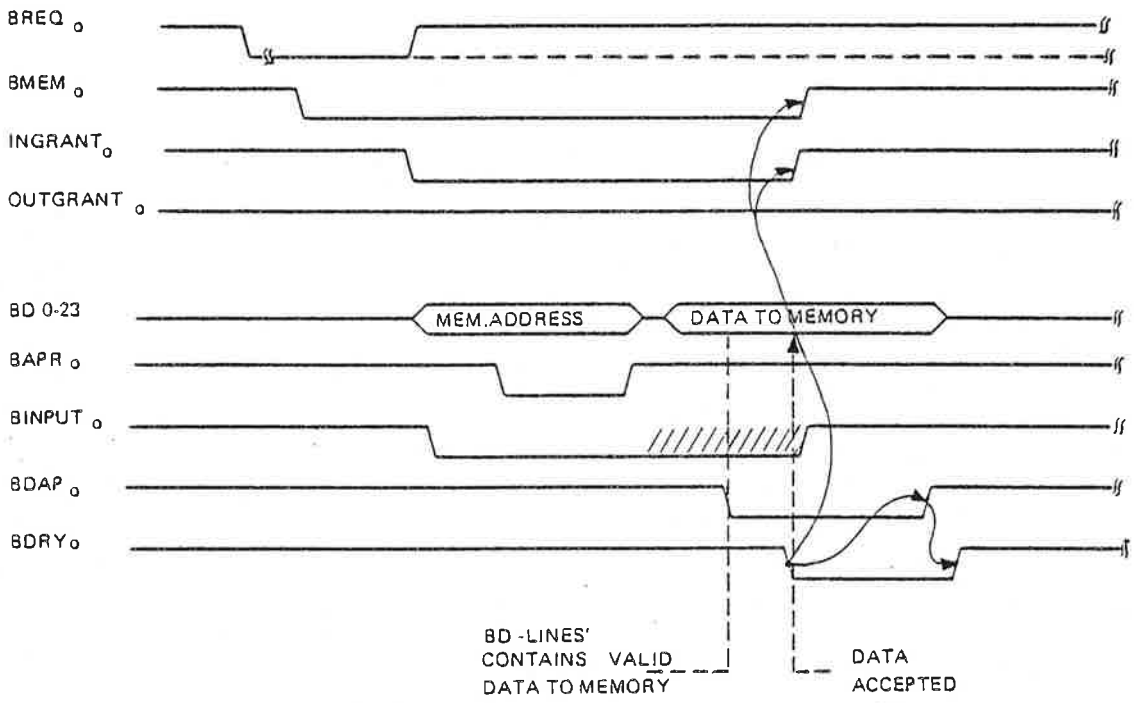


Figure V.4.2: DMA Input (Memory Write) Transfer via the NORD-100 Bus

The memory system uses $BDAP_0$ to strobe the data in a data buffer and activates $BDRY_0$ to signify "data accepted".

The leading edge of $BDRY_0$ terminates the grant mechanism while the trailing edge of $BDRY_0$ terminates the bus cycle.

A DMA controller's handling of direct memory transfers are covered in the next section.

V.5 DMA TRANSFERS AND THE DMA CONTROLLERS

Each DMA controller has their own "request/grant logic". This logic is specially designed to request allocation of the NORD-100 bus and as the request is acknowledged, to handle a memory reference.

The handshake signals in a DMA transfer and the sequence in which they appear at the NORD-100 bus are supposed to be known from the previous section. Based on this knowledge, Figure V.5.1 illustrates why the different signals are generated and how they are used.

Refer to Figure V.5.1 and note that the BCU is not involved in the memory reference. The memory reference is handled by the granted DMA controller. Further, note that the only control signal in response from the memory system is $BDRY_0$.

BDRY is the well known "transfer completed and release bus" signal which prevents time out, i.e., memory out of range (MOR) in this case. If a MOR occurs during a DMA cycle this will be indicated in the PES (Parity Error Status register) by bit 14 (DMA) set and bit 15 (Fetch) not set.

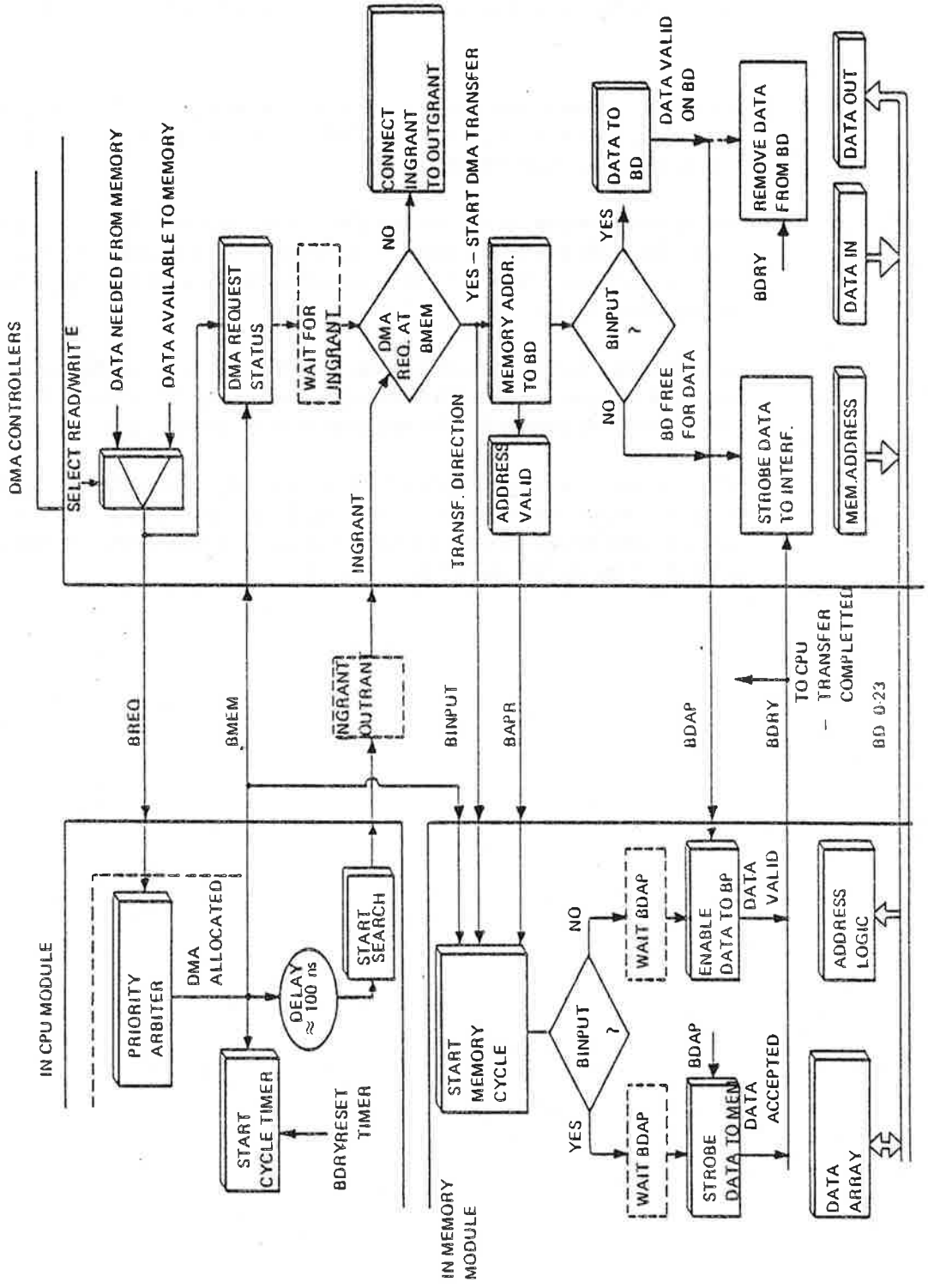


Figure V.5.1: DMA Controller and BCU and MEMORY Interaction

VI NORD-100 BUS EXTENDER (BEX)

VI.1 GENERAL

The number of modules that could be connected to a NORD-100 system is of course limited to the number of available NORD-100 bus slot positions.

Thus, the limitation set by one of the available card crates is 12 or 21 modules depending on whether a 12 or a 21 position card crate is selected.

Although 21, or often less than 12 modules are sufficient for most systems, some configurantions require more space than even the 21 position card crate can offer.

This potential space problem is solved by the NORD-100 Bus EXtender (BEX) system.

The BEX system makes it possible to extend the NORD-100 bus structure by linking together from two to a theoretical maximum of 8 NORD-100 card crates. It should be noted that only one CPU module can be connected to the system.

VI.2 FUNCTIONAL DESCRIPTION

VI.2.1 *DEFINITION OF TERMS*

The implementation of the NORD-100 Bus Extender System is handled by a module, obviously enough, called Bus Extender (BEX). Two crates are physically connected via two cables between one BEX module in each crate. Thus, the BEX modules occupy one slot position in each of the crates that are to be linked together (see Figure VI.2.1).

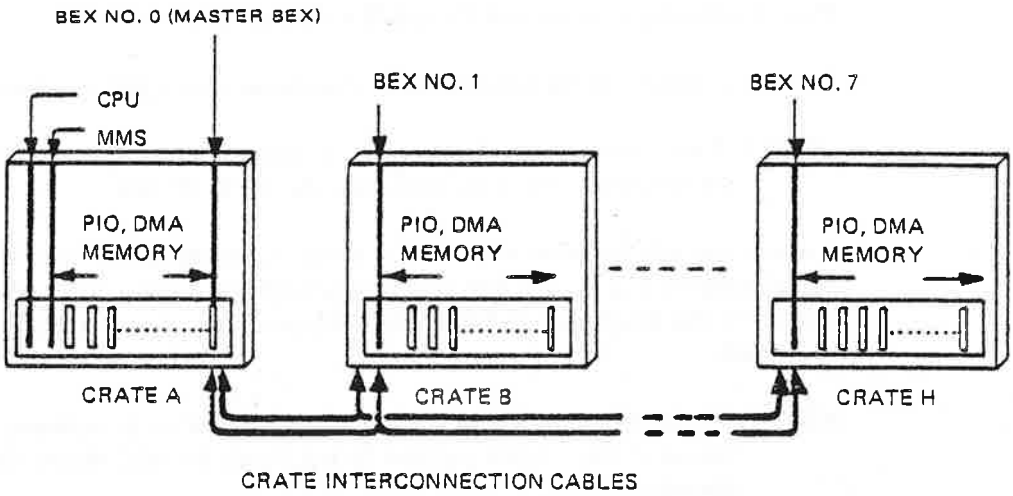


Figure VI.2.1: NORD-100 Bus Extender System

Only one CPU can be connected to the Bus Extender System.

The crate where the CPU is located is referred to as the A crate. The BEX module located in the A crate has to be BEX no. 0 which defines it as the "MASTER BEX".

From the MASTER BEX in crate A, two cables are connected to BEX no. 1, located in the next crate, called crate B.

From BEX no. 1 in crate B, two new cables give a link further to BEX no. 2 in crate C and so on.

The NORD-100 BEX system ensures that each slot position in any crate hase equal properties. Thus, it is possible to mix PIO controllers, DMA controllers and memory modules in all the crates.

VI.2.2 *ORGANIZATION OF MODULES IN A BUS EXTENDED SYSTEM*

The actual placement of modules in a bus extended system follows the same rules as for a single crate system.

The overall subject in the placement rules is to maintain the propagation of the daisy chained backplane signals, i.e., INIDENT/OUTIDENT, INGRANT/OUTGRANT and BINCONTR/BOUTCONTR. (The last chain is for future use.)

From the discussion of these signals earlier in this manual, remember that they all are generated by the CPU module.

Rules for Placement of the MASTER BEX in the A crate :

The search chains are connected between the crates via the BEX modules.

NOTE 1: Thus, there should never be any empty positions between the CPU and the first BEX in the crate chain, i.e., the MASTER BEX.

If one of the search signals are received active by the MASTER BEX, it means the actual search signal *did not* find stop condition in the A crate. Thus, the search is stopped in the A crate at the MASTER BEX's position and passed over to the next card crate.

NOTE 2: If the last I/O device controller (PIO or DMA) or memory module is placed in the A crate position N, the MASTER BEX should be placed in position N + 1.

The normal situation is that note 1 is satisfied by a filled A crate. Consequently, the MASTER BEX is normally placed in the last A crate position.

Rules for Placement of BEX No. 1 to BEX No. 7:

The BEX modules no. 1 to no. 7 are all located in card crates with no CPU. However, they are linked to the CPU by the MASTER BEX. Thus, the BEX modules "represent" or "simulate" the CPU for the modules in the crates B, C, ... H.

NOTE 3: BEX no. 1 to no. 7 should be placed in position 1 in their respective crates (i.e., BEX no. 1 in position 1 in the B crate and so on).

VI.3 CONTROL OF THE BUS EXTENDER (BEX) MODULES

VI.3.1 *INTRODUCTION*

In addition to being a transparent bus extension, the BEX modules provide several features which are controllable by program or switches.

Most of the control functions are aimed against controlled routing of memory addresses when memory is partioned between several card crates. In addition, the response to different error situations may be set individually for each crate.

Some registers may be read for handling of error situations or configuration investigation.

Programmed access to the BEX modules has to be done by means of IOXT instructions (device register address ⩾ 100 000).

VI.3.2 *THE MEMORY ADDRESS ROUTING MECHANISM*

The total memory capacity in a bus extended system may be divided in different crates.

In order to route the memory addresses to the crates where they are represented by physical memory, each BEX module has a Lower Limit (LL) address register and an Upper Limit (UL) address register.

On each BEX connected to a crate with memory, these registers have to be given a value which corresponds to the memory area covered by the crate.

If a BEX is connected to a crate without memory, the values set in the limit registers should be 0, to avoid unnecessary bus activity.

In addition to the Lower Limit and Upper Limit address registers, each BEX contains a base register. The value of the base register is used to give a positive offset to the address presented to a card crate.

The following illustrates how the three address registers (LL, UL, Base) work together.

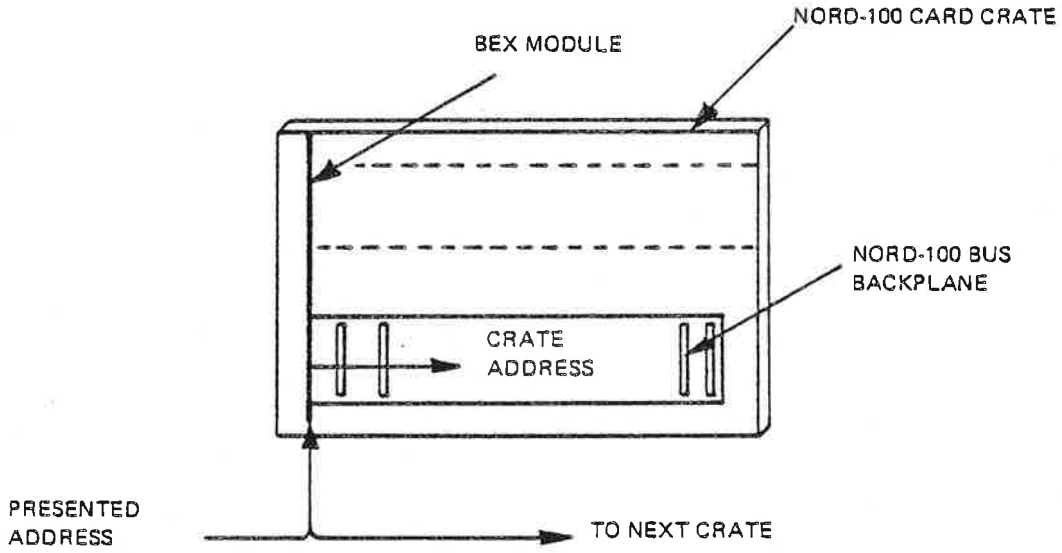


Figure VI.3.1: Memory Address Routing

During a memory reference, either initiated by the CPU or a DMA controller, all crates will be presented the memory address simultaneously. Thus, all BEX modules, in parallel, will "look" at the address to see if it is between the Lower Limit and Upper Limit set for the crate.

Only one BEX should find the presented address between its limits.

$$LL \leqslant \text{PRESENTED ADDRESS} < UL$$

The BEX which finds the address ok starts a memory cycle in its crate, where memory is presented a "crate address" (refer to Figure VI.3.1).

The crate address is calculated by the BEX as given below.

$$\text{CRATE ADDR.} = \text{PRESENTED ADDR.} - LL + \text{BASE}$$

For addresses below 1 M word, Lower Limit (LL) is set equal to the base. That is,

$$\text{CRATE ADDR.} = \text{PRESENTED ADDR.}$$

Only one answer from memory is ensured by the memory modules own lower/ upper limit address test.

Setting of the BEX limit registers could either be done by switches or by program. How this is done, together with an example, is explained in the next sections.

VI.3.3 *HARDWARE SWITCH SETTING*

Most of the parameters that are to be set on the BEX modules, could be set either by program or switches.

If corresponding parameters are set both by program and switches, the programmed value will be used. However, it should be noted that after a "MASTER CLEAR" or a programmed "DEVICE CLEAR" the programmed values will be reset and the hardware settings used as "default values".

The only hardware settings that cannot be set by the program is the BEX number and the VITAL switch (see the description).

Refer to Figure VI.3.2 for physical placement of the switches, the thumbwheel and the Light Emitting Diode (LED) indicator. Refer also to the figure for associated abbreviations.

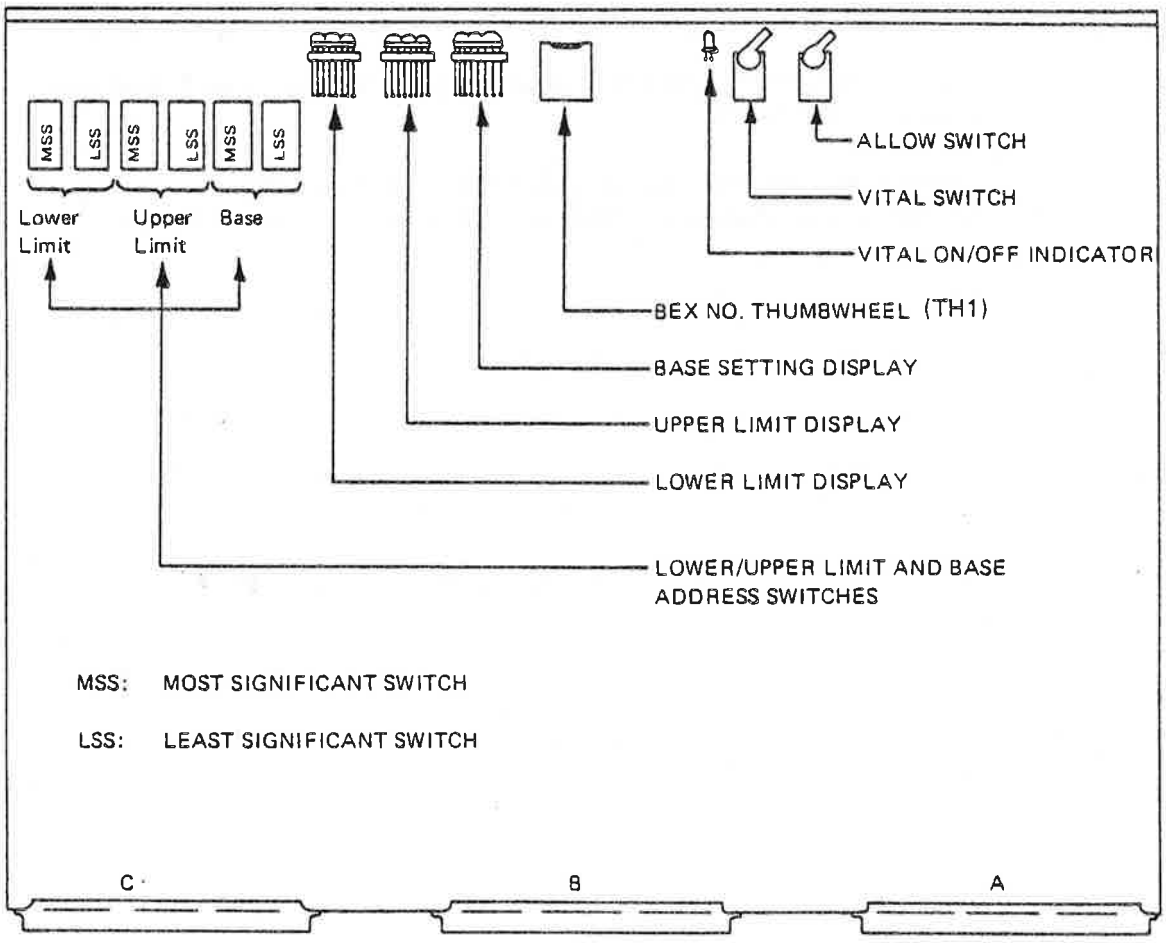


Figure VI.3.2: Physical Placement of Switches and Indicators on a BEX Module

BEX Number Selection — The Device Number (DEV. NO.) Thumbwheel

Eight BEX numbers are defined, thus the device number thumbwheel may take the values from 0, 1, ..., 7.

TH1 pos	Dev. No.	Int. Level	Ident Code
0*	100000	13	10
1	100004	13	11
2	100010	13	12
3	100014	13	13
4	100020	13	14
5	100024	13	15
6	100030	13	16
7	100034	13	17

* BEX No. 0 is defined as MASTER BEX and should be placed in the A crate.

The ALLOW Switch

This function is not applicable and the switch should always be in ON position on all BEX's. The position of this switch could be investigated by program by reading BEX status register (see programming specification section VI.3.4).

The VITAL Switch

This switch is not relevant for the MASTER BEX (BEX no. 0). The VITAL switch controls how a BEX (BEX no. ⩾ 1) will report a Power Fail Interrupt (PFI) in its local crate.

If the switch is ON (yellow LED not lit) and a PFI occurs in a crate, the connected BEX reports a PFI causing level 14 interrupt in the CPU (if enabled). That is, as if the PFI occurred in the A crate.

If the VITAL switch is OFF (yellow LED lit) the BEX sensing PFI reports this to be the CPU by an interrupt on level 13. It must then be decided by software how vital, i.e., how serious the PFI is.

The Lower Limit, Upper Limit and Base Address Register Switches and their Associated Display Indicators

The switches (and displays) for hardware setting of the address limits and the base register is physically located in the upper left corner of the BEX module (refer to Figures VI.3.2 and VI.3.3).

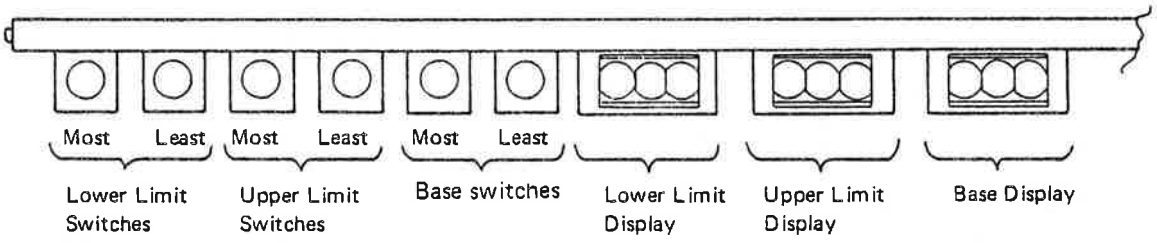


Figure VI.3.3: Upper Left Corner of a BEX Module seen from the Top of the Module

The displays will always show the currently used limit/base setting. Thus, corresponding display and pair of switches will be equal if none of the limit registers has been programmed to a different value than the value set by the switches.

The displays will not follow dynamically changes in the switches. Change in the switches is set into the limit registers and displayed after a "MASTER CLEAR" or programmed "DEVICE CLEAR".

The resolution of the switches is 64 K words per number turn on the least significant limit switch.

In Table VI.3.1 a complete list is given for all the possible switch combinations. The numbers given in the table correpsond to the displayed value by a given switch setting in either the Lower Limit, Upper Limit or Base registers.

To get the corresponding value in K words, a table entry (Table VI.3.1) should be converted to decimal and multiplied by 64 K words.

Refer also to Section VI.4 — Configuration Examples.

Most Sign. AMP Switch \ Least Significant AMP Switch	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	000	001	002	003	004	005	006	007	010	011	012	013	014	015	016	017
1	020	021	022	023	024	025	026	027	030	031	032	033	034	035	036	037
2	040	041	042	043	044	045	046	047	050	051	052	053	054	055	056	057
3	060	061	062	063	064	065	066	067	070	071	072	073	074	075	076	077
4	100	101	102	103	104	105	106	107	110	111	112	113	114	115	116	117
5	120	121	122	123	124	125	126	127	130	131	132	133	134	135	136	137
6	140	141	142	143	144	145	146	147	150	151	152	153	154	155	156	157
7	160	161	162	163	164	165	166	167	170	171	172	173	174	175	176	177
8	200	201	202	203	204	205	206	207	210	211	212	213	214	215	216	217
9	220	221	222	223	224	225	226	227	230	231	232	233	234	235	236	237
A	240	241	242	243	244	245	246	247	250	251	252	253	254	255	256	257
B	260	261	262	263	264	265	266	267	270	271	272	273	274	275	276	277
C	300	301	302	303	304	305	306	307	310	311	312	313	314	315	316	317
D	320	321	322	323	324	325	326	327	330	331	332	333	334	335	336	337
E	340	341	342	343	344	345	346	347	350	351	352	353	354	355	356	357
F	360	361	362	363	364	365	366	367	370	371	372	373	374	375	376	377

Table VI.3.1: Least/Most Switch Settings and Corresponding Display Values

VI.3.4 *BUS EXTENDER (BEX) PROGRAMMING SPECIFICATIONS*

The BEX module is programmable by means of IOXT instructions. Each BEX number (device no.) is assigned 4 IOXT device register addresses.

The registers corresponding to the device register addresses is given below.

Device Register Address	Register
Dev. No. + 0	Read Data
Dev. No. + 1	Write Data
Dev. No. + 2	Read Status Register
Dev. No. + 3	Write Control Register

The value of Dev. No. depends on the device number thumbwheel setting and is given in Section VI.3.3

Register Format and Description

CONTROL REGISTER (DEV. NO. + 3)

Bit No.	Function
0	Not Assigned (NA)
1	Error interrupt enable level (13)
2	Activate, write data to selected register
3	Not Assigned
4	Clear Device
5	Mode (see decoding below)
6	Mode (see decoding below)
7	Mode (see decoding below)
8	Allow
9	External Interrupt Disable
10-15	Not Assigned

Bit Description

Bit 1: If this bit is set, a level 13 interrupt will be generated if a parity error or local power fail interrupt is detected.

Bit 2: Activate.
Only used when writing memory limits into the selected limit register. The selection is done by the mode bits in the same control word as activate.

Bit 4: Clear Device.
Writing the control register with this bit set generates a reset pulse on the affected BEX. The reset pulse will set the BEX to "initial" state.

Programmed content in the limit and base registers will be reset. The switches will be set into the limit/base regsiters and the displays PES and PEA registers are not affected.

Bits 5-7: Modus.
The Modus bits are used to specify which register is to be accessed in a read or write data operation. The decoding of the modus bits is given below.

Bit No.			
7	6	5	Specified register
0	0	0	Read Parity Error Status Register (PES)
0	0	1	Read Parity Error Address Register (PEA)
0	1	0	Read Lower Limit
0	1	1	Read Upper Limit
1	0	0	Write Lower Limit
1	0	1	Write Upper Limit
1	1	0	Write Base

During a write operation the content of the BEX data register is moved to the register specified in the modus bit if the activate bit is set.

In a read operation, the specified regsiters will be enabled onto the external bus during the next Read Data cycle.

Bit 8: Allow.
Should always be one.

Bit 9: External Interrupt Disable.
A possibility in software to disable interrupts from a specific crate.

STATUS REGISTER (DEV. NO. + 2)

Bit No.	Function
0	Not Assigned
1	Interrupt enabled (copy of bit one in CONT. REG.)
2	Not Assigned
3	= 1
4	Error
5-7	Mode (copy of same bits in CONT. REG.)
8	Allow
9	External activity when failing
10	Parity Error
11	Not Assigned
12	Power Fail Interrupt
13-15	Not Assigned

READ DATA (DEV. NO. + 0)
WRITE DATA (DEV. NO. + 1)

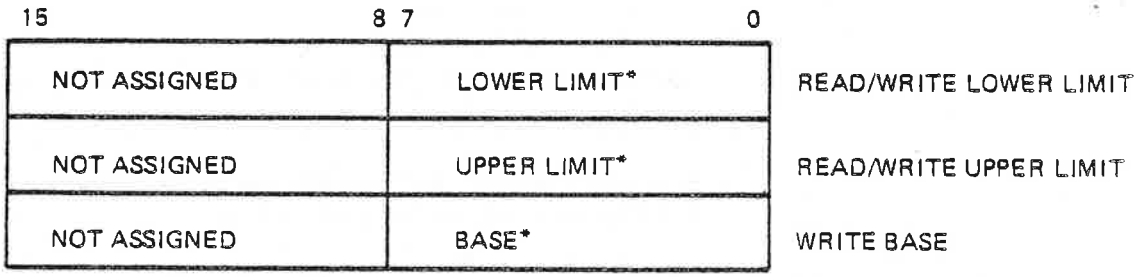
A READ DATA (dev. no. + 0) operation should always be preceded by selection of a register by means of the modus bits in the control register.

A WRITE DATA (dev. no. + 1) operation should always be followed by specification of the actual destination register in the BEX.

The destination register is specified in the modus bits in the control register together with the activate bit (bit 2) set to one.

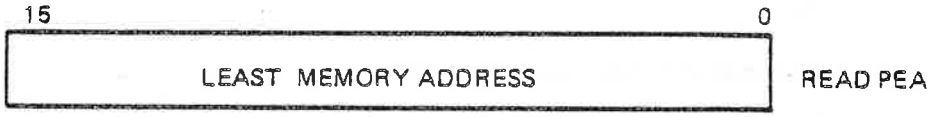
Data Formats

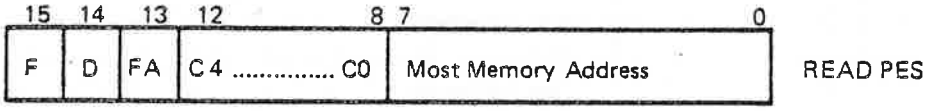
The A register format before write BEX data register when destination is lower/upper limit register or base register, and A regsiter after a read of the same registers.



* Lower/upper limit and base values are in accordance with Table VI.3.1.

Format of the A register after reading PES and PEA registers:





Bits 8-12: Syndrome bits (error code) used to find failing bit if FA = 0.

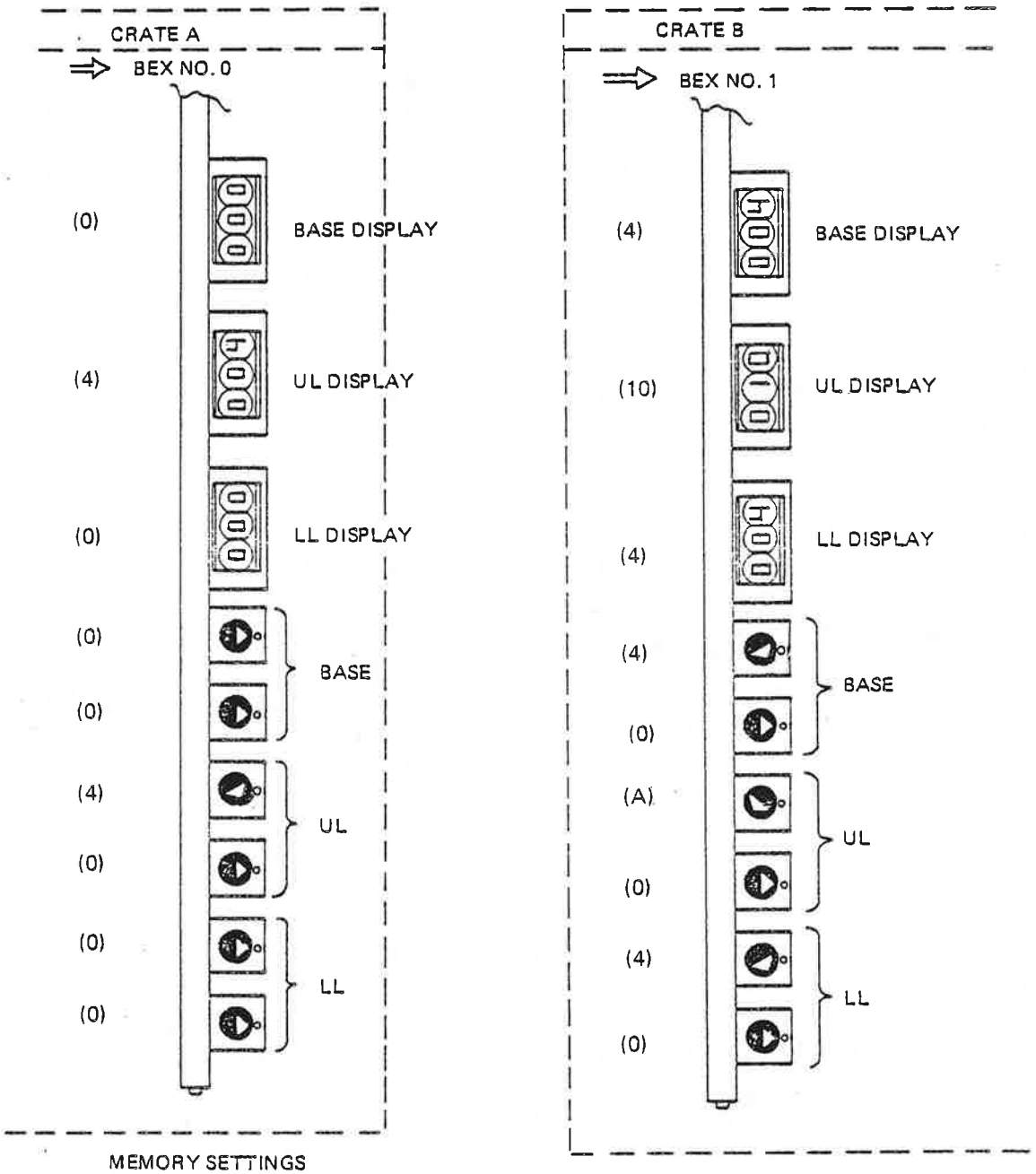
Bit 13: Fatal
"1" = uncorrectable error
"0" = corrected error

Bit 14: DMA — error occurred during DMA cycle in this crate.

Bit 15: Fetch — always read as 0 since fetch is an internal CPU signal.

VI.4 CONFIGURATION EXAMPLES

Assume a system of two NORD-100 card crates and 512 k words physical memory. The memory capacity is divided equally between the crates, i.e., 256 k words in each crate. In switch settings the BEX modules will then be given as below.



MEMORY SETTINGS
CRATE A

00	0-64k
04	64-128k
10	128-192k
14	192-256k

MEMORY SETTINGS
CRATE B

20	256-320k
24	320-384k
30	384-448k
34	448-512k

Figure VI.4.1: BEX Configuration Example

VI.5 BEX INTERCONNECTION — PHYSICAL CABLE ARRANGEMENT

The BEX modules (i.e., the card crates) are physically linked together by two cables. The cables are connected to the BEX ADAPTER (a small print board) mounted in the plug panel in the rear of the cabinet.

The BEX ADAPTER contains four slots for cable connection — two for ingoing cables and two for outgoing cables (to the next crate). The connectors used for ingoing cables are marked AIN, BIN. The connectors used for outgoing cables are marked AOUT, BOUT. Refer to Figure VI.5.1 for illustration.

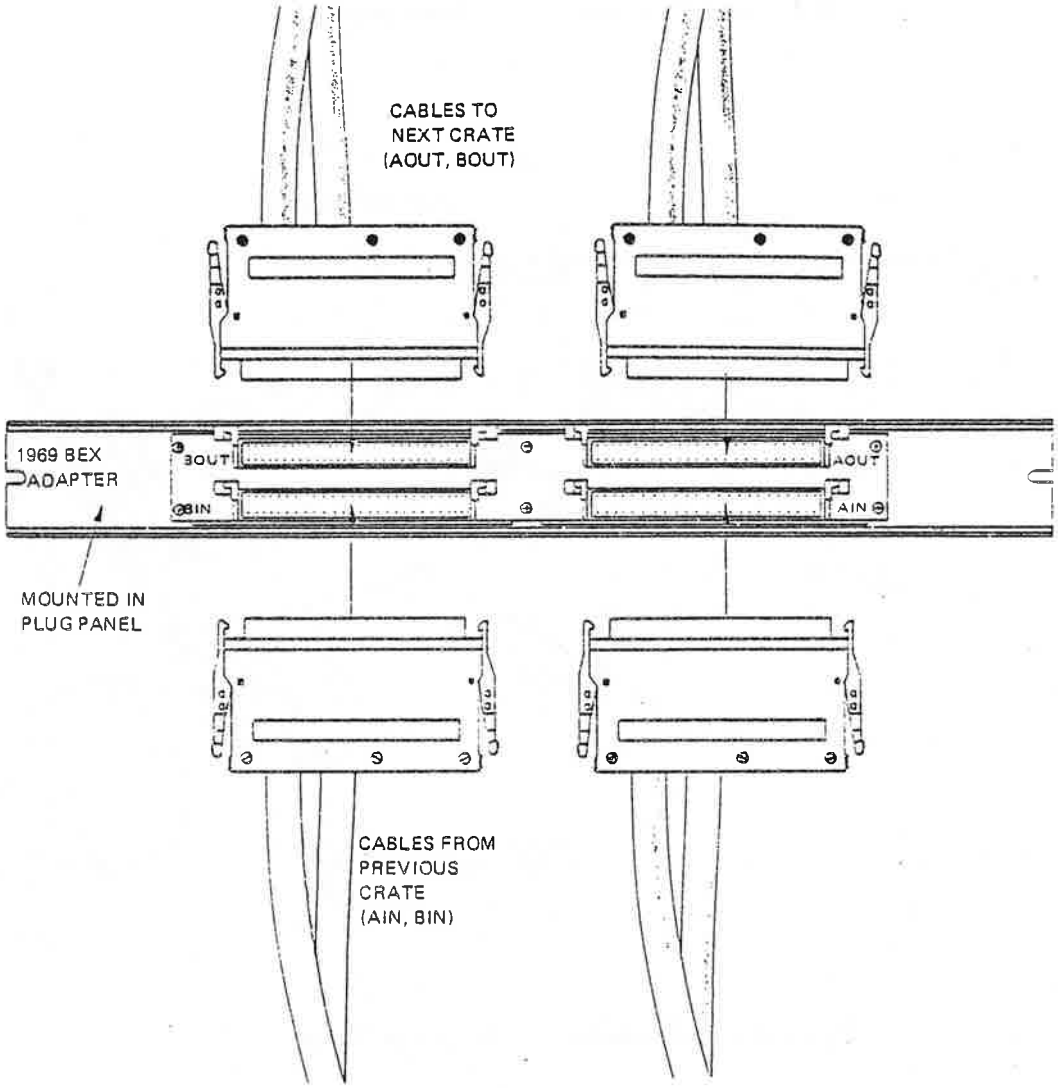


Figure VI.5.1: Illustration — 1969 BEX ADAPTER and BEX Interconnection Cables

The AIN and BIN connectors are never connected (used) in the A crate, while they are always used on other crates.

The AOUT and BOUT connectors are used in all crates except the last crate in the crate chain.

The BEX ADAPTER AIN and BIN connectors are connected to the BEX module by two internal flat cables.

The internal cable is equipped with a termination plug where each cable signal is terminated. The termination plugs are physically connected to the internal cables approximately 10 cm from where they reach the BEX module. Thus, there is no need for termination in the last (and open) AOUT and BOUT ports. Refer to Figure VI.5.2 for an illustration of the internal wiring.

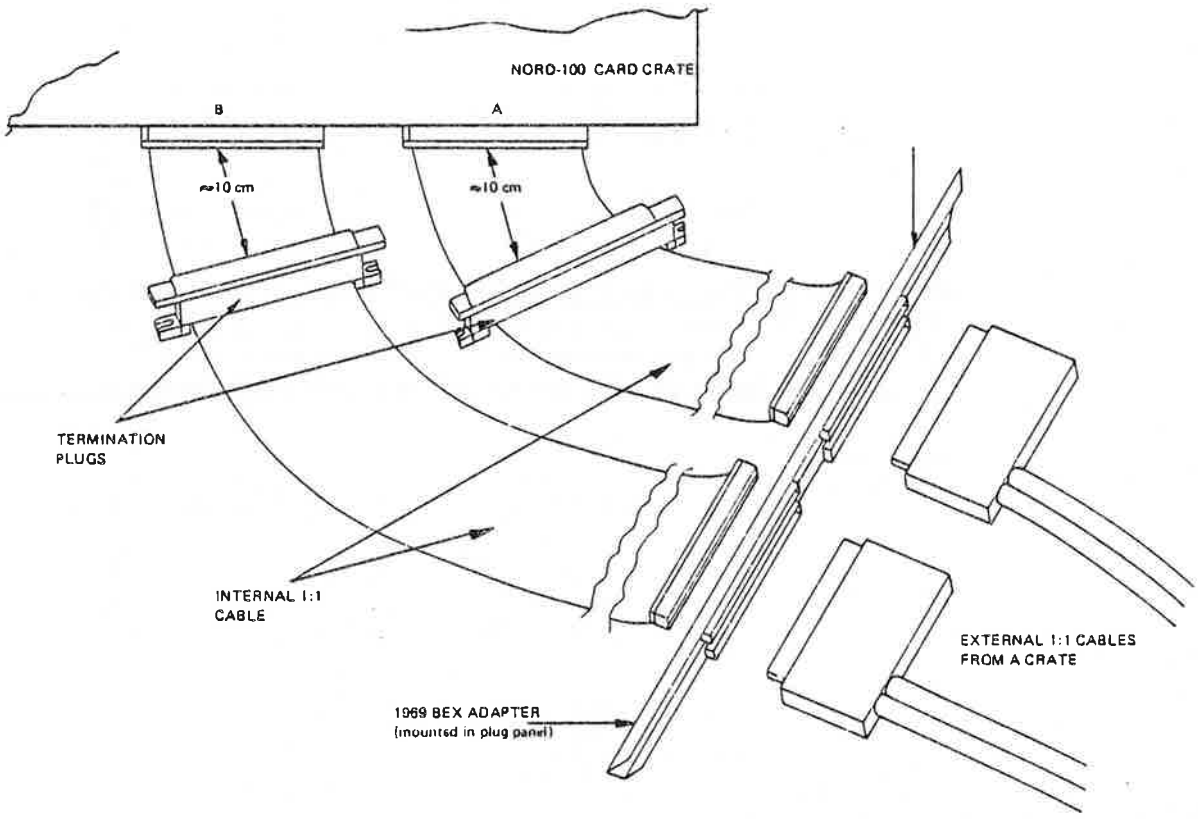


Figure VI.5.2: Illustration — BEX Internal Wiring

APPENDIX A

STANDARD NORD-100 DEVICE REGISTER ADDRESSES AND IDENT CODES

In the following only the most frequently used Device Names are listed.

Two Device Names may use the same Device Register Address range. In these cases only the most common Device Name is listed.

Device Reg. Address Range	*Interrupt Level*	*SINTRAN III Logical Device Numbers* (octal)*	*Ident Code (octal)*	*Device Name*
4- 7	13		4	Memory Parity N-12/N-42
10- 13	13		1	Real Time Clock 1
14- 17	13		2	Real Time Clock 2
20- 23	13		6	Real Time Clock 3
24- 27	13		7	External Interrupt
30- 33	12		16	NORD-50/1
34- 37	10		16	ACM 5
40- 43	10		15	ACM 1
44- 47	10		25	ACM 2
50- 53	10		40	ACM 3
54- 57	10		41	ACM 4
60- 77				NORD-50/1 Regs.
100-107	10-12	6	4	Sync. Modem 1
110-117	10-12	16	14	Sync. Modem 2
120-127	10-12	30	20	Sync. Modem 3
130-137	10-12	31	24	Sync. Modem 4
140-147	10-12	26	30	Sync. Modem 5
150-157	10-12	27	34	Sync. Modem 6
160-167	10-12		40	Sync. Modem 7
170-177	10-12		10	Sync. Modem 8
200-207	10-12	7	60	Terminal 17
210-217	10-12	17	61	Terminal 18
220-227	10-12	52	62	Terminal 19
230-237	10-12	53	63	Terminal 20
240-247	10-12	54	64	Terminal 21
250-257	10-12	55	65	Terminal 22
260-267	10-12	56	66	Terminal 23
270-277	10-12	57	67	Terminal 24
300-307**	10-12	1	1(120)***	Terminal 1
310-317**	10-12	11	5(121)***	Terminal 2/TET 15
320-327**	10-12	42	6(122)***	Terminal 3/TET 14
330-337**	10-12	43	7(123)***	Terminal 4/TET 13
340-347	10-12	44	44	Terminal 5/TET 12
350-357	10-12	45	45	Terminal 6/TET 11
360-367	10-12	46	46	Terminal 7/TET 10
370-377	10-12	47	47	Terminal 8/TET 9

* A complete list of SINTRAN III Logical Device Numbers is found in SINTRAN III Reference Manual (ND-60.125).

** Terminal no. 1 is implemented on the CPU module. Terminals with device numbers 310-317, 320-327 and 330-337 are normally not used.

*** Number in parenthesis is valid for 4 current loop modules.

Device Reg. Address Range	*Interrupt Level*	*SINTRAN III Logical Device Numbers* (octal)*	*Ident Code (octal)*	*Device Name*
400- 403	12	2	2	Paper Tape Reader 1
404- 407	12	12	22	Paper Tape Reader 2
410- 413	10	3	2	Paper Tape Punch 1
414- 417	10	13	22	Paper Tape Punch 2
420- 423	12	4	3	Card Reader 1
424- 427	12	14	23	Card Reader 2
430-433	10	5	3	Line Printer 1
434- 437	10	15	23	Line Printer 2
440- 443	10	10	11	Calcomp Plotter 1
444- 447	10	50	12	Card Punch 1
450- 453	10	35	21	Card Punch 3/Calc. 2
454- 457	10	51	13	Card Punch 2
460- 467	10-12		31	E & Pict. Syst. I/O
470- 477	12			Graphical Pen
500- 507	11		1	Disk System 1
510- 517	11		5	Disk System 2
520- 527	11		3	Mag.Tape 1
530- 537	11		7	Mag. Tape 2
540- 547	11		2	Drum 1
550- 557	11		6	Drum 2
560- 577	12-13	1006	156	HDLC HASP 1
600- 607	11	22	4	Versatec 1
610- 617	11		11	Core -to-Core 1
620- 637	11	36	10	CDC I/O Link
640- 647	10-12	1040	124	Terminal 33
650- 657	10-12	1041	125	Terminal 34
660- 667	10-12	1042	126	Terminal 35
670- 677	10-12	1043	127	Terminal 36
700- 707	12	20	11	CATSY 1
710- 717	12	21	21	CATSY 2
720- 727	11		23	E & S Pict. Syst. DMA
730- 737	10		10	D/A- Converter
750- 753	13		5	BIG MPM LOG Module
754- 757	12		13	Process Input 5
760- 767	10-11-12-13		100	Test Card
770- 773	12		17	Dig. Reg. 1 Input
774- 777	10		17	Dig. Reg. 1 Output
1000-1003	12		26	Dig. Reg. 2 Input
1004-1007	10		26	Dig. Reg. 2 Output
1010-1013	12		27	Dig. Reg. 3 Input
1014-1017	10		27	Dig. Reg. 3 Output
1020-1023	12		43	Dig. Reg. 4 Input
1024-1027	10		43	Dig. Reg. 4 Output
1030-1033	12		116	NORD 50/2
1034				Watch Dog
1035				Process Output 1
1036				Process Output 2

Device Reg. Address Range	*Interrupt Level*	*SINTRAN III Logical Device Numbers* (octal)*	*Ident Code (octal)*	*Device Name*
1037				Process Output 3
1040-1043	12		15	Process Input 1
1044-1047	12		25	Process Input 2
1050-1053	12		40	Process Input 3
1054-1057	12		12	Process Input 4
1060-1077				NORD-50/2 Reg.
1100-1107	10-12	1044	130	Terminal 37
1110-1117	10-12	1045	131	Terminal 38
1120-1127	10-12	1046	132	Terminal 39
1130-1137	10-12	1047	133	Terminal 40
1140-1147	10-12	1050	134	Terminal 41
1150-1157	10-12	1051	135	Terminal 42
1160-1167	10-12	1052	136	Terminal 43
1170-1177	10-12	1053	137	Terminal 44
1200-1207	10-12	70	70	Terminal 25
1210-1217	10-12	71	71	Terminal 26
1220-1227	10-12	72	72	Terminal 27
1230-1237	10-12	73	73	Terminal 28
1240-1247	10-12	74	74	Terminal 29/PHOTOS. 1
1250-1257	10-12	75	75	Terminal 30/PHOTOS.2
1260-1267	10-12	76	76	Terminal 31/PHOTOS.3
1270-1277	10-12	77	77	Terminal 32/PHOTOS. 4
1300-1307	10-12	60	50	Terminal 9
1310-1317	10-12	61	51	Terminal 10
1320-1327	10-12	62	52	Terminal 11
1330-1337	10-12	63	53	Terminal 12
1340-1347	10-12	64	54	Terminal 13
1350-1357	10-12	65	55	Terminal 14
1360-1367	10-12	66	56	Terminal 15
1370-1377	10-12	67	57	Terminal 16
1400-1407	10-12	1054	140	Terminal 45
1410-1417	10-12	1055	141	Terminal 46
1420-1427	10-12	1056	142	Terminal 47
1430-1437	10-12	1057	143	Terminal 48
1440-1443	12		101	A/D Converter 1
1444-1447	12		102	A/D Converter 2
1450-1453	12		103	A/D Converter 3
1454-1457	12		104	A/D Converter 4
1460-1463	12		105	A/D Converter 5
1464-1467	12		106	A/D Converter 6
1470-1473	12		107	A/D Converter 7
1474-1477	12		110	A/D Converter 8
1500-1507	10-12	1060	144	Terminal 49

Device Reg. Address range	*Interrupt Level*	*SINTRAN III Logical Device Numbers (octal)*	*Ident Code (octal)*	*Device Name*
1510-1517	10-12	1061	145	Terminal 50
1520-1527	10-12	1062	146	Terminal 51
1530-1537	10-12	1063	147	Terminal 52
1540-1547	11		17	Big Disk System 1
1550-1557	11		20	Big Disk System 2
1560-1567	11	1000- 1002	21	Floppy Disk 1 (Unit 0, 1, 2)
1570-1577	11	1003- 1005	22	Floppy Disk 2 (Unit 0, 1, 2)
1600-1603	11		14	Versatec 2
1604-1607				HDLC Remote Load 1
1610-1613				HDLC Remote Load 2
1614-1617				HDLC Remote Load 3
1620-1623				HDLC Remote Load 4
1624-1627				HDLC Remote Load 5
1630-1633				HDLC Remote Load 6
1634-1637				HDLC Remote Load 7
1640-1657	12-13		150	HDLC NORD-NET 1
1660-1677	12-13		151	HDLC NORD-NET 2
1700-1717	12-13		152	HDLC NORD-NET 3
1720-1737	12-13		153	HDLC NORD-NET 4
1740-1757	12-13		154	HDLC NORD-NET 5
1760-1777	12-13		155	HDLC NORD-NET 6

APPENDIX B

INTERNAL REGISTERS

The following internal registers are implemented for internal control and status of the CPU, memory management system and memory. These registers are only accessed by privileged instructions, and could not be accessed by an ordinary customer's program. Internal registers can be accessed when the computer is in STOP or OPCOM mode.

For a detailed description refer to the manual "NORD-100 Functional Description".

Register Name:	*No.:*	*Description:*
PANS	0	Panel status register. Gives information to the microprogram about the display status. Also used by microprogram.
PANC	0	Panel Control. Controls the state of the display from the microprogram. Also used by microprogram.
STS	1	Status Register. Bits 0-7 are level dependent and accessible from user programs while bits 8-15 are system dependent and only accessible by system (TRA/TRR).
OPR	2	Operator's register. Implemented in firmware.
LMP	2	Display register. Implemented in firmware.
PSR	3	Paging status register.
PCR	3	Paging control register.
PVL	4	Previous level. The content of the register is: IRR <previous level * 10_8> DP.
IIC	5	Internal interrupt code.
IIE	5	Internal interrupt enable.
PID	6	Priority interrupt detect.
PIE	7	Priority interrupt enable.
CSR	10	Cache status.
CCLR	10	Clear cache
LCILR	11	Lower cache inhibit limit register
ACTL	11	Active level
ALD	12	Automatic load descriptor
UCILR	12	Upper cache inhibit limit register
PES	13	Parity error status
PCR	14	Paging control register read on specified level
PEA	15	Parity error address

B.1 *PROGRAMMING SPECIFICATIONS FOR TERMINAL NO. 1*

The current loop interface, located on the CPU board, has device numbers 300-307.

IOX 300:

Read input data (according to input control word setting). The last inputted character is transferred to the A register. The data available signal is reset if MOPC is not active.

IOX 301:

No operation.

IOX 302:

Read input status.

Bit 0 = 1; data available will give interrupt when it occurs.

Bit 3 = 1; data is available (ready for transfer). Is never given if MOPC is active.

4 = 1; inclusive or of error bits 5-7.

Bit 5 = 1; framing error.

Bit 6 = 1: parity error.

Bit 7 = 1; overrun.

Bits 1-2 and 8-15 are always zero.

IOX 303:

Set input control.

Bit 0 = 1; enable interrupt if data available (ready for transfer) occurs.

Bit 11 and Bit 12:

Bit 11 = 1 and Bit 12 = 1 signifies 5 bits code.
Bit 11 = 0 and Bit 12 = 1 signifies 6 bits code.
Bit 11 = 1 and Bit 12 = 0 signifies 7 bits code.
Bit 11 = 0 and Bit 12 = 0 signifies 8 bits code.

Bit 13 = 1 signifies 1 stop bit.
Bit 13 = 0 signifies 2 (1.5 for 5 bits) stop bits.

Bit 14 = 1; a parity bit is added to the number of bits mentioned above.
Bit 14 = 0; no extra bit is added to the bits mentioned above.

IOX 304:

Returns 0 in the A register and has no other effect.

IOX 305:

Write data (according to input control word setting).

IOX 306:

Read output status.

Bit 0 = 1; ready for transfer will give interrupt when it occurs.
Bit 3 = 1; ready for transfer.
Bits 1-2 and 4-15 are always zero.

IOX 307:

Set output control.

Bit 0 = 1; enable interrupt if ready for transfer occurs.

B.2 *NORD-100 4 OR 8 ASYNCHRONOUS SERIAL INTERFACE PROGRAMMING SPECIFICATIONS*

GENERAL

Four/eight independent asynchronous serial interface lines. Each line has independent split speed (input/output speed) possible range from 50 to 9600 baud.

Line connection is switch selectable for each line (see Appendix C.2).

Connection could be in accordance to:

- 20mA current loop for local terminal connection
- CCITT V.24/X.21 bis (EIA RS-232c) for full duplex asychronous, "dialed up" modem lines.

B.2.1 *NORD-100 4 or 8 Asynchronous Current Loop Programming Specifications*

TERMINAL ADDRESS CODES

The IOX <dev. no.> relevant for different terminals is found in Appendix A. The device number (<dev. no.>) is selected by a thumbwheel. Device number and corresponding thumbwheel settings are given in Appendix D.2.2.

INPUT CHANNEL (INTERRUPT LEVEL 12)

Read Data Register

IOX <dev. no.> + 0

The number of data bits read into the A register is specified by bits 11 and 12 in the interrupt channel control register. The received character is right justified (from bit 0 and upwards).

Read Status Register

IOX <dev. no.> + 2

See paragraph for Status Registers.

Write Control Register

IOX <dev. no.> + 3

See paragraph for Control Register.

OUTPUT CHANNEL (INPUT LEVEL 10)

Write Data Register

IOX <dev. no.> + 5

The number of bits specified by bits 11 and 12 in the *input* channel control register is written to the output data register, starting with bit 0 and counting upwards.

Read Status Register

IOX <dev. no.> + 6

See paragraph describing Status registers.

Write Control Register

IOX <dev. no.> + 7

See paramter describing control registers.

DATA RATE SELECTION

IOX <dev. no.> + 1

IDENT CODE

The ident code for the input channel and the output channel will be the same, with the input channel responding to level 12 and the output channel responding to level 10. The selection of different ident codes are given in the paragraph on Ident does and Interrupt mechanism.

INPUT CHANNEL

Status Register

Bit	
0	Ready for transfer interrupt enabled
1	Not used
2	Not used
3	Device ready for transfer
4	Inclusive OR of errors
5	Framing error
6	Parity error
7	Overrun
8	Not used
9	
10	
11	
12	
13	
14	
15	

Note: Additional explanation to status bits.

Bit 5: Framing error means that the stop bit is missing.

Bit 6: Parity error means that a parity error has occurred while working in parity generating/checking mode.

Bit 7: Overrun means that at least one character is overwritten while input is active.

Control Register

Bit

0	Enable interrupt on device ready for transfer
1	Not used
2	Not used
3	Test mode
4	Not used
5	
6	
7	
8	
9	
10	
11-12	Character length
13	Number of stop bits
14	Parity generation/checking
15	Not used

Note: There is no need for separate activation. The received data will always be clocked into the receiver data buffer.

Bit 3: Test mode will loop transmitted data back to received data and if the other terminal is connected to the line, transmitted data will also be transferred to this terminal. If test mode is selected for one of the four interfaces all four will be set in test mode.

Bit 11-12: The content of these bits gives the following character lengths, both for the input channel and the output channel.

Bit 12	Bit 11	
0	0	8 bits
0	1	7 bits
1	0	6 bits
1	1	5 bits

If bit 14 is a 1, a parity bit is *added* to the number given in this table.

Bit 13: This bit = 0 will select 1.5 stop bits for 5 bits character and 2 stop bits otherwise. This bit = 1 will select 1 stop bit.

Bit 14: If this control bit is 0, no parity bit will be added to the character on the output channel and the received character will not be checked for parity. A 1 in this control bit will add an even parity bit to the character on the output channel, and give an error indication if the received character has an odd parity.

OUTPUT CHANNEL

Status Register

Bit

0 Ready for transfer interrupt enabled
1 Not used
2 Not used
3 Device ready for transfer
4-15 Not used

Bit 3: This bit indicates that the output data buffer is ready to receive a new character.

Control Register

Bit

0 Enable interrupt on device ready for transfer.
1-15 Not used

Note: The device is activated when a character is loaded into the output character register. There is no need for separate activation.

CONTROL AND STATUS WORDS

Input

Bit	Status	Control
0	RFT	Enable RFT
1		
2		
3	Dev. RFT	Test
4	ERR OR	
5	Framing	
6	Parity	
7	Overrun	
8		
9		
10		
11		Character length
12		Character length
13	Stop bits	
14		Parity generation/check
15		

Output

Bit	Status	Control
0	RFT enable	Enable RFT
1		
2		
3	Device RFT	
4	ERR OR	
5	Framing	
6	Parity	
7	Overrun	
8		
9		
10		
11		
12		
13		
14		
15		

IDENT CODES AND INTERRUPT MECHANISM

Ident Codes

The ident codes are binary coded by the switches (bits 2-7) in position 6E, with 0 corresponding to ON and 1 corresponding to OFF.

All ident codes from 0 to 377_8 can be selected.

Interrupt Mechanism

What is needed for a device to give an interrupt?

First of all, the device must be ready for a transfer, i.e., status bit 3 must be on. For input this means that a whole character is received by the input buffer and is ready to be read into the A register. For output it means that it is possible to place at least one more character in the output buffer. Secondly, interrupt on ready for transfer must be control register bit 0 (which also is status register bit 0). The AND function of ready for transfer and ready for transfer interrupt enabled is gated to put it connected to interrupt level 12 (terminal 35) and output is connected to interrupt level 10 (terminal 27). When an interrupt is detected (dependant on the status in CPU and the program), the CPU for the interrupting level, which gives the ident code in A register. The ident code is identical for input and output channel.

B.2.2 *NORD-100 ASYNCHRONOUS V24 (MODEM) PROGRAMMING SPECIFICATIONS*

TERMINAL ADDRESS CODES

The IOX <dev. no.> relevant for different terminals is found in Appendix A. The device number (<dev. no.>) is selected by a thumbwheel. Device number and corresponding thumbwheel settings are given in Appendix D.2.2.

INPUT CHANNEL (INTERRUPT LEVEL 12)

Read Data Register

IOX <dev. no.> + 0

The number of data bits read into the A register is specified by bits 11 and 12 in the input channel control register. The received character is right justified (from bit 0 and upwards).

Read Status Register

IOX <dev. no.> + 2

Write Control Register

IOX <dev. no.> + 3

OUTPUT CHANNEL (INTERRUPT LEVEL 10)

Write Data Register

IOX <dev. no.> + 5

The number of bits specified by bits 11 and 12 in the *input* channel control register is written to the output data register, starting with bit 0 and counting upwards.

Read Status Register

IOX <dev. no.> + 6

Write Control Register

IOX <dev. no.> + 7

DATA RATE SELECTION

IOX <dev. no.> + 1

IDENT CODE

The ident code for the input channel and the output channel will be the same, with the input channel responding to level 12 and the output channel responding to level 10.

INPUT CHANNEL

Status Register

Bit

0	Ready for transfer interrupt enabled
1	Not used
2	Not used
3	Device ready for transfer
4	Inclusive OR of errors
5	Framing error
6	Parity error
7	Overrun
8, 9, 10	Not used
11	Carrier missing
13, 14, 15	Not used

Note: Additional explanation to status bits.

Bit 5: Framing error means taht the stop bit is missing.

Bit 6: Parity error means that a parity error has occurred while working in parity generation/checking mode.

Bit 7: Overrun means that at least one character is overwritten while input is active.

Bit 11: Carrier missing gives the status of received line signal detector or carrier on the line.

0 indicates carrier present
1 indicates carrier missing

Control Register

Bit

0 Enable interrupt on device ready for transfer
1 Not used
2 Not used
3 Test mode
4 Start time-out for breaking connection
5
6
7 } Not used
8
9
10
11-12 Character length
13 Number of stop bits
14 Parity generation/checking
15 Not used

Note: There is not need for separate activation. The received data will always be clocked into the receiver data buffer.

Bit 3: Test mode will loop transmitted data back to received data, and if the other terminals are connected to the line transmitted data will also be transferred to this terminal. If test mode is selected for one of the two interfaces both will be st in test mode.

Bit 4: If this bit is activated, the DATA TERMINAL READY signal will drop after approximately 20 seconds if no characters are received.

Bits 11-12: The content of these bits give the following character lengths, both for the input channel and the output channel.

Bit 12	Bit 11	
0	0	8 bits
0	1	7 bits
1	0	6 bits
1	1	5 bits

If bit 14 is a 1, a parity bit is *added* to the number given in this table.

Bit 13: This bit = 0 will select 1.5 stop bits for 5 bits character and 2 stop bits otherwise. This bit = 1 will select 1 stop bit.

Bit 14: If this control bit is 0, not parity bit will be added to the character on the output channel, and the receiver character will not be checked for parity. A 1 in this control bit will add an even parity bit to the character on the output channel and give an error indication if the received character has an odd parity.

OUTPUT CHANNEL

Status Register

Bit

0 Ready for transfer interrupt enabled
1 Not used
2 Not used
3 Device ready for transfer
4-10 Not used
11 Carrier missing
12-15 Not used

Bit 3: This bit indicates that the output data buffer is ready to receive a new character.

Bit 11: As for input channel.

Control Register

Bit

0 Enable interrupt on device ready for transfer
1-15 Not used

Note: The device is activated when a character is loaded into the output character register. There is not need for separate activation.

CONTROL AND STATUS WORDS

Input

Bit	Status	Control
0	RFT enable	Enable RFT
1		
2		
3	Dev. RFT	Test
4	ERR OR	Start time-out
5	Framing	
6	Parity	
7	Overrun	
8		
9		
10		
11	Carrier missing	Character length
12		Character length
13		Stop bits
14		Parity generation/check
15		

Output

Bit	Status	Control
0	RFT enable	Enable RFT
1		
2		
3	Dev. RFT	
4	ERR OR	
5	Framing	
6	Parity	
7	Overrun	
8		
9		
10		
11	Carrier missing	
12		
13		
14		
15		

INTERFACE SIGNALS

Connection Pin	Signal Name	Description
A	Transmitted data	Data from NORD
D	Request to send (RTS)	Will be high when DSR is high.
E	Data terminal ready (DTR)	Will be high when DSR is low. DTR is used to disconnect the modem connection. If signal detector and DSR are high, the first input character has to be received within 20 seconds. If not, DIR will go low for approximately 6 seconds, and reamin oscillating with 20 seconds high and 6 seconds low, until an input character is received.
F	Ready for sending (RFS)	Must be high when NORD transmitted data.
H	Data set ready (DSR)	High will give high on RTS. Low will give low on RTS and a steady high on DTR.
J	Signal detector (CARRIER)	Must be high when NORD receive data.
	Received data	Data to NORD.
M	Signal ground	Ground reference for all interface signals.

All signal levels are in accordance with EIA Standard RS-232C.

B.3 *SPECIFICATION OF LINE PRINTER INTERFACE FOR CDC 9380 FOR NORD-10/100*

Standard device no.:	0430 (0430-0433) octal
No. of device no.:	4
Standard int. level:	10 des.
Standard ident no.:	3

Write Control Word IOX DEV NO. +3

Bit 0:	Enable interupt on ready for transfer
Bit 1:	Enable interrupt on error
Bit 2:	Activate device (print character now in buffer)
Bit 3:	Test
Bit 4:	Device and interface clear
Bits 5-15:	Not used

Read Status Word IOX DEV NO. +2

Bit 0:	Interrupt enabled on ready
Bit 1:	Interrupt enabled on error
Bit 2:	Not used
Bit 3:	Ready for transfer
Bit 4:	Error, bit 5 or 6 set
Bit 5:	Line printer not ready
Bit 6:	Out of paper
Bit 7:	Compressed pitch
Bit 8:	LP9 is on, to indicate to the controller that data on the lines is format information and is interpreted as control code
Bit 9:	Inhibit, illegal character in buffer
Bit 10:	Not used
Bits 11-12:	Band detect

Bit 11	Bit 12	Type of band
0	0	128 characters
1	0	96 characters
0	1	64 characters
1	1	48 characters

Note: This interface is only handling 64, 96 character printers.

Bits 13-15: Not used

Write Data Word IOX DEV NO. + 1

Writes a character in the buffer register.

All character codes $0\text{-}37_8$ are illegal and ignored by the interface, except the following control codes:

11_8	HT	(gives space in CDC controller)
12_8	LF	
14_8	FF	
15_8	CR	
$20_8\text{-}33_8$		VFU channels give LP9 and disable LP5
20_8		VFU channel 1 (FF)
21_8		
.		
.		
33_8		VFU channel 12

Read Data Word IOX DEV. NO. + 0

It is possible to read back the data written in the buffer when running in test mode (bit 3 set in control word).

B.4 *NORD-100 DISK PROGRAMMING SPECIFICATIONS*

Disk Device Register Address:

The codes below are relevant for disk system I. Each disk system may consist of 4 disk units. For disk system II add 10_8 to the specified codes.

Read Memory Address*

The 24 bits memory address has to be read by two consecutive IOX 500s. The least 16 bits are read by the first IOX 500. The most significant 8 bits are read by next IOX 500.

IOX 500

Load Memory Address*

The 24 bits memory address has to be loaded by two consecutive IOX 501. The most significant 8 bits are loaded by the first IOX 501. The least significant 16 bits are loaded with the next IOX 501.

IOX 501

Read Sector Counter

IOX 502

Load Block Address

IOX 503

Read Status Register

IOX 504

Load Control Word

IOX 505

Read Block Address

IOX 506

Load Word Counter Register

IOX 507

The minimum number of words to be transferred is one sector, i.e., 200_8 words, the maximum number of words is one track, i.e., 25 sectors.

* The sequence is initialized by a "Master Clear", programmed device clear or a read status. During transfers it is only possible to read the 16 least significant address bits.

Read Block Address

This instruction is implemented for maintenance purposes only. By first loading a control word with bit 3 (test mode), the instruction

IOX 506

will return the previously loaded block address to the A register.

Control Word

Bit 0:	Enable interrupt on device ready for transfer
Bit 1:	Enable interrupt on errors
Bit 2:	Activate device
Bit 3:	Test mode
Bit 4:	Device clear
Bit 5:	Not assigned
Bit 6:	Not assigned
Bit 7:	Marginal recovery
Bit 8:	Not assigned
Bits 9-10:	Unit select
Bits 11-12:	Device operation
Bit 13:	Not assigned
Bit 14:	Not assigned
Bit 15:	Write format

Unit Select Code:

Bit 10	9	
Bit 0	0	Unit 0
Bit 0	1	Unit 1
Bit 1	0	Unit 2
Bit 1	1	Unit 3

Device Operation Code:

Bit 12	11	
Bit 0	0	Read transfer
Bit 0	1	Write transfer
Bit 1	0	Read parity
Bit 1	0	Compare

Status Word

Bit 0:	Ready for transfer, interrupt enabled
Bit 1:	Error interrupt enabled
Bit 2:	Device active
Bit 3:	Device ready for transfer
Bit 4:	Inclusive OR of errors (status bits 5-11)
Bit 5:	Write protect violate
Bit 6:	Time out
Bit 7:	Missing clock/Disk fault/Seek error
Bit 8:	Address mismatch
Bit 9:	Parity error
Bit 10:	Compare error
Bit 11:	DMA error
Bit 12:	Transfer complete
Bit 13:	Transfer on
Bit 14:	On cylinder
Bit 15:	Bit 15 loaded by previous control word

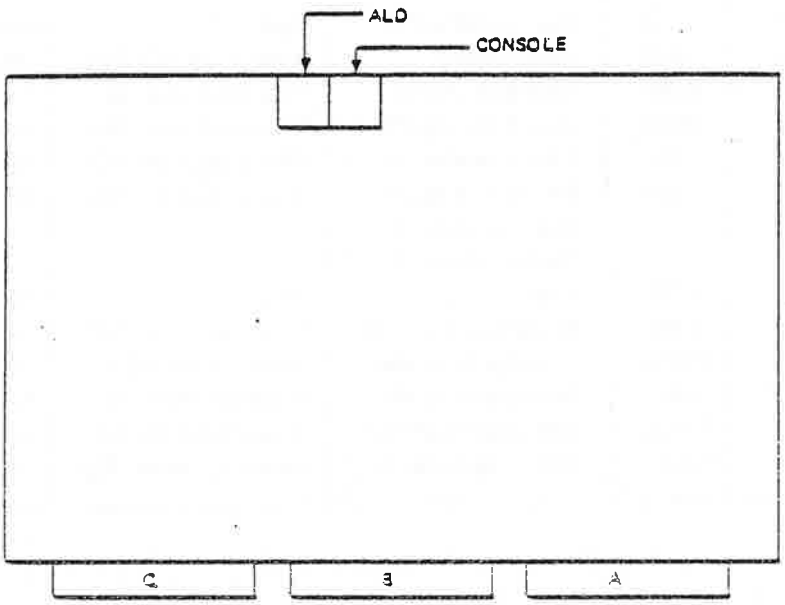
Interrupt

The disk interrupt level is 11 and the ident number for the first disk system is 1.

APPENDIX C

SWITCH SETTINGS FOR THE DIFFERENT NORD-100 MODULES

C.1 *SWITCHES ON THE CPU MODULE (3002)*



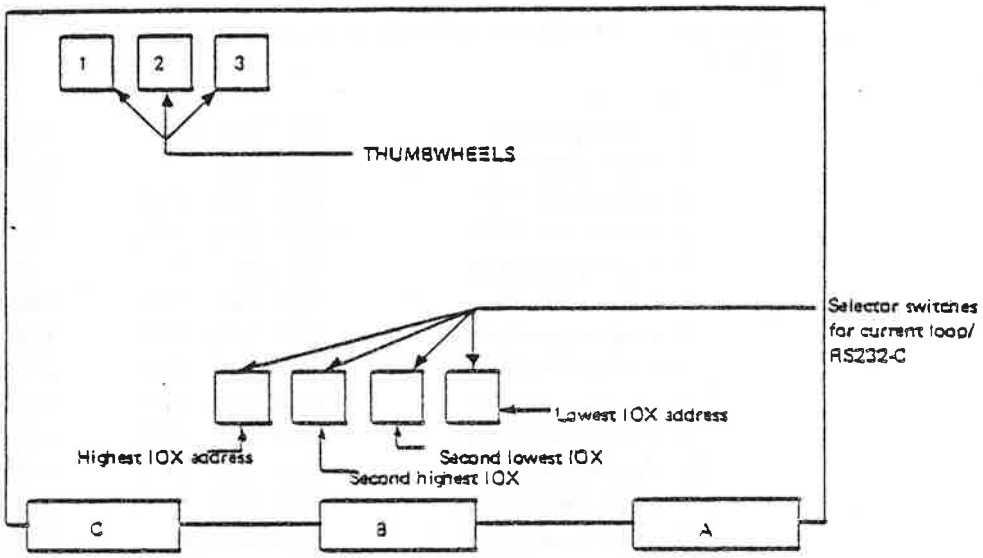
C.1.1 *ALD — Automatic Load Descriptor*

ALD	I12	LOCK and Standby Power OK	LOCK and Standby Power not OK	UNLOCK and Load
15	0	Start in address 20	Stop	Nothing
14	1560	Start in address 20	Binary load from 1560	Binary load from 1560
13	20500	Start in address 20	Mass load from 500	Mass load from 500
12	21540	Start in address 20	Mass load from 1540	Mass load from 1540
11	400	Start in address 20	Binary load from 400	Binary load from 400
10	1600	Start in address 20	Binary load from 1600	Binary load from 1600
9		Start in address 20		
8		Start in address 20		
7	100000	Stop	Stop	Nothing
6	101560	Binary load from 1560	Binary load from 1560	Binary load from 1560
5	120500	Mass load from 500	Mass load from 500	Mass load from 500
4	121540	Mass load from 1540	Mass load from 1540	Mass load from 1540
3	100400	Binary load from 400	Binary load from 400	Binary load from 400
2	101600	Binary load from 1600	Binary load from 1600	Binary load from 1600

C.1.2 *Console: Speed setting for console terminal.*

0	110 baud
1	150 baud
2	300 baud
3	2400 baud
4	1200 baud
5	1800 baud
6	4800 baud
7	9600 baud
8	2400 baud
9	600 baud
10	200 baud
11	134.5 baud
12	75 baud
13	50 baud
14	—
15	—

C.2 *SWITCHES ON FLOPPY AND 4 TERMINALS MODULE (3010)*



1 = Floppy disk system
2 = terminal group
3 = initial band rate for terminals

C.2.1 *1 Floppy Disk System*

0 = floppy system no. 1 (10X 1560 - 1567, IDENT = 21)
1 = floppy system no. 2 (10X 1570 - 1577, IDENT = 22)

2-15 are unused, will answer on 1OX 0-7.

C.2.2 *2 Terminal Group:*

Each group consists of 4 terminals with consecutive I0X addresses and ident codes.

0 = terminals 1-4	(IOX 300 - 337,	IDENT 120-123)
1 = terminals 5-8	(IOX 340 - 377,	IDENT 44-47)
2 = terminals 9-12	(IOX 1300 - 1337,	IDENT 50-53)
3 = terminals 13-16	(IOX 1340 - 1377,	IDENT 54-57)
4 = terminals 33-36	(IOX 640 - 677,	IDENT 124-127)
5 = terminals 37-40	(IOX 1100 - 1137,	IDENT 130-133)
6 = terminals 41-44	(IOX 1140 - 1177,	IDENT 134-137)
7 = terminals 45-48	(IOX 1400 - 1437,	IDENT 140-143)
8 = terminals 49-52	(IOX 1500 - 1537,	IDENT 144-147)
9 = terminals 53-56	(IOX 1640 - 1677,	IDENT 150-153)
10 = terminals 57-60	(IOX 1700 - 1737,	IDENT 154-157)
11 = terminals 61-64	(IOX 1740 - 1777,	IDENT 160-163)
12 = terminals 17-20	(IOX 200 - 237,	IDENT 60-63)
13 = terminals 21-24	(IOX 240 - 277,	IDENT 64-67)
14 = terminals 25-28	(IOX 1200 - 1237,	IDENT 70-73)
15 = terminals 29-32	(IOX 1240 - 1277,	IDENT 74-77)

C.2.3 *3 Initial Baud Rate for Terminals*

0 = 110 baud
1 = 150 baud
2 = 300 baud
3 = 2400 baud
4 = 1200 baud
5 = 1800 baud
6 = 4800 baud
7 = 9600 baud
8 = 2400 baud
9 = 600 baud
10 = 200 baud
11 = 134.5 baud
12 = 75 baud
13 = 50 baud
14 = unused
15 = unused

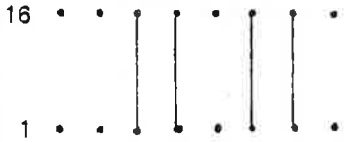
Selector switches for current loop/RS232-C.
Switch set to 0 selects current loop.
Switch set to 1 selects RS232-C.
Switch settings under Terminal Group and Initial baud rate for terminals are also valid for the 8 terminal modules (3013).

This description is correct for the 2 position switch.

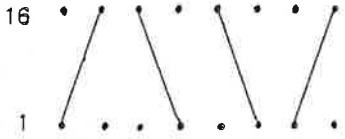
If a hexadecimal switch is used:

0 = current loop
F = RS232-C

If component houses are used:

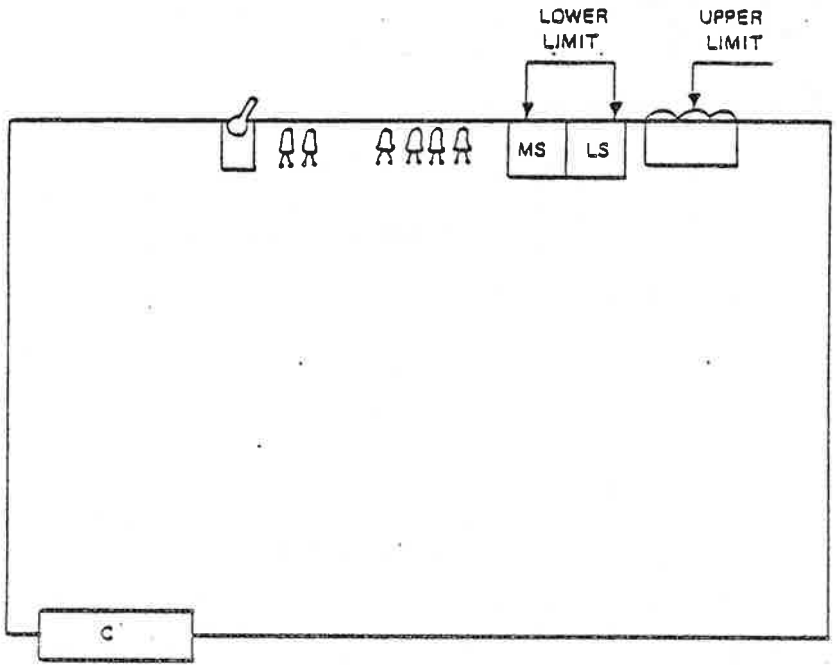


Selects current loop



Selects RS-232-C

C.3 *SWITCHES ON MEMORY MODULES (3005)*



Lower limit is a 2 digit octal number, determining lower memory address.

Upper limit is automatically displayed according to actual memory size.

The limit address increments are 16K units, such that:

Octal Address = 40,000 x limit.

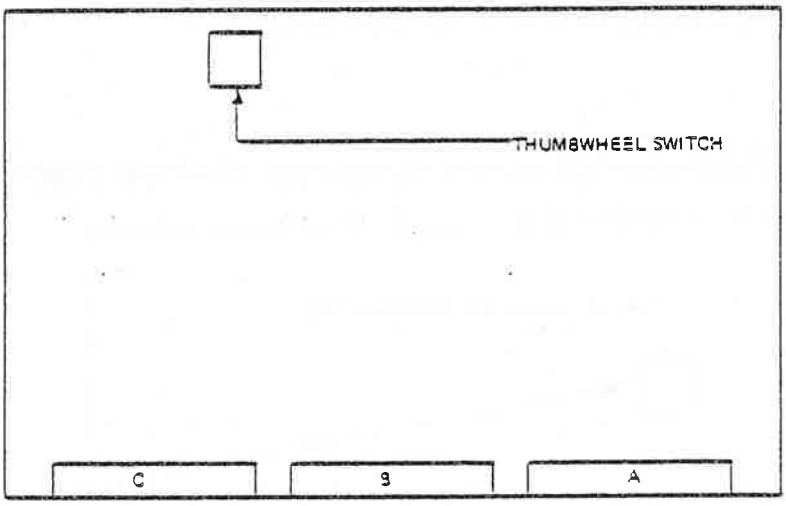
Lower limit	Size	Upper limit	Address range
0 0	16K	0 1	0 - 16K
0 0	32K	0 2	0 - 32K
0 0	64K	0 4	0 - 64K
0 3	32K	0 5	48 - 80K
0 3	64K	0 7	48 - 112K
3 4	64K	4 0	448 - 512K

The switches can be set to numbers 8 - 15, but only 0-7 normally have a meaning. It is, however, possible to allow the slot position code to determine the address range by putting lower limit to 88. (This is only meaningful for 64K). Then the address ranges will be:

Slot	Lower limit	Size	Upper limit	Address Range
12	88	64K	04	0-64K
11	88	64K	10	64-128K
10	88	64K	14	. .
9	88	64K	20	. .
8	88	64K	24	. .
7	88	64K	30	. .
6	88	64K	34	384-448K
5	88	64K	40	448-512K

etc.

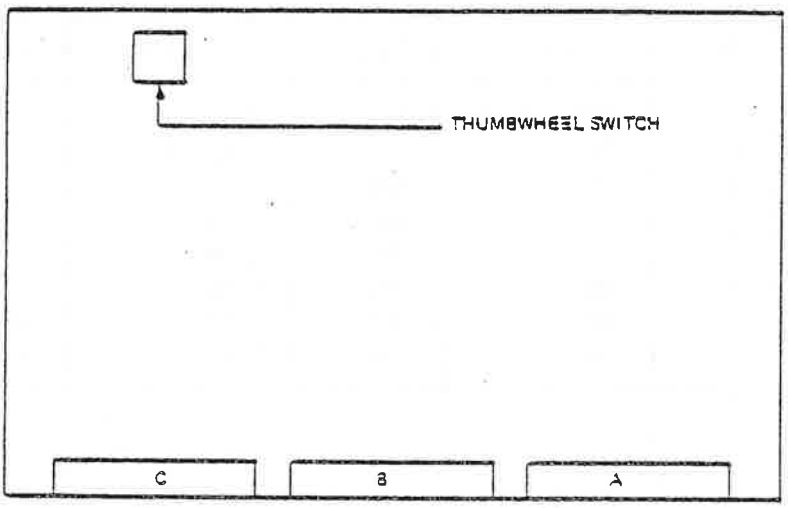
C.4 *SWITCHES ON THE 10MB DISK MODULE (3004)*



Switch pos 0: device no. 500-507, ident no. 1 (disk system 1)
Switch pos 1: device no. 510-517, ident no. 5 (disk system 2)
Switch pos 2-15: not used

IOX address bit 15 active will inhibit this card.

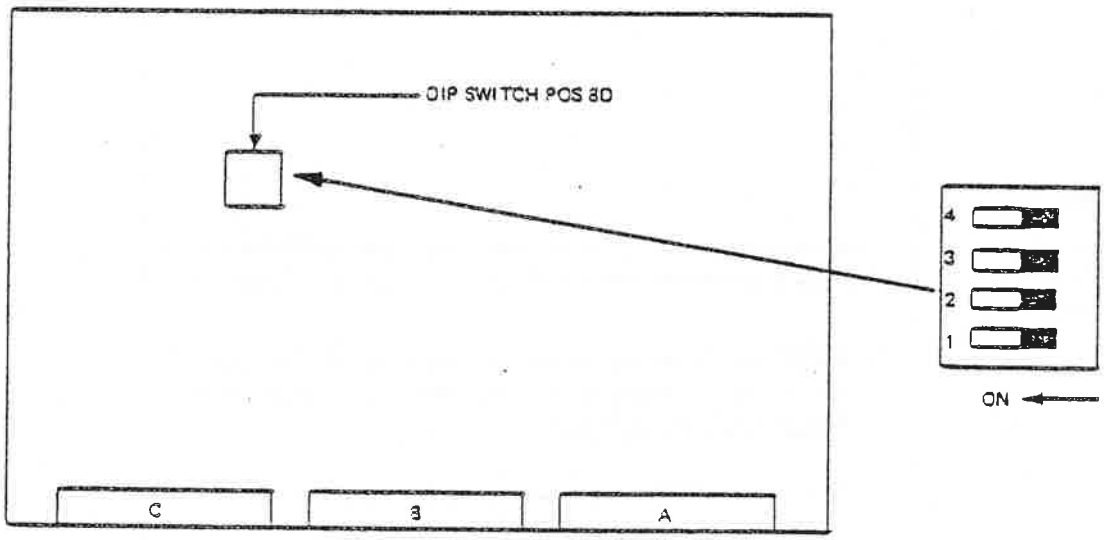
C.5 *SWITCH SETTING ON THE PERTEC MAGNETIC TAPE MODULE (3006)*



Switch pos 2: device no. 520-527, ident no. 3 (mag. tape 1)
Switch pos 3: device no. 530-537, ident no. 7 (mag. tape 2)
Switch pos 2-15: not used.

IOX address bit 15 active will inhibit this card.

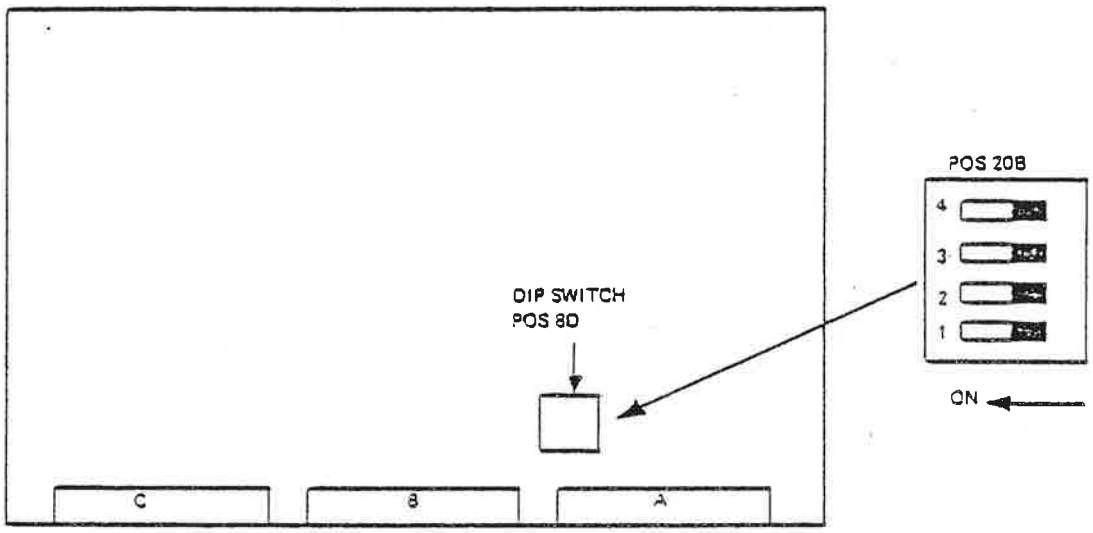
C.6 *SWITCH SETTING ON NORD-100 BUS ADAPTER (3008)*



Switch 1 = off: 0 ⩽device no. ⩽ 3777, 0 ⩽ ident no. ⩽ 377
Switch 1 = on: 2000 ⩽ device no. ⩽ 3777, 400 ⩽ ident no. ⩽ 777
Switch 2 = off: normal
Switch 2 = on: block all interfaces on this bus
Switches 3 and 4: not used.

IOX address bit 15 active will inhibit this card.

C.7 *SWITCH SETTING ON LOCAL I/O BUS (3009)*

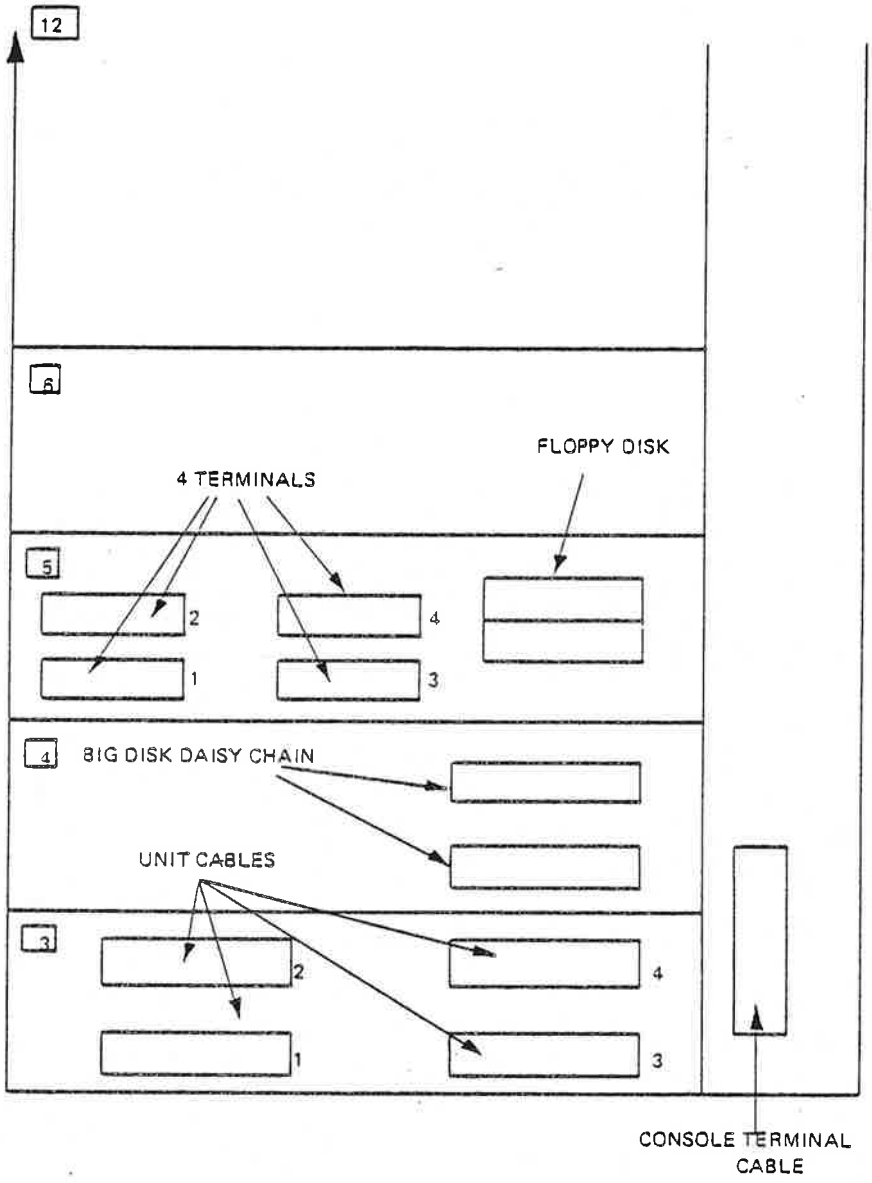


Switch 1 = off: 0 ⩽device no. ⩽ 1777, 0 ⩽ ident no. ⩽ 377
Switch 1 = on: 2000 ⩽ device no. ⩽ 3777, 400 ⩽ ident no. ⩽ 777
Switch 2 = off: normal
Switch 2 = on: block all interfaces on this bus
Switches 3 and 4: not used.

IOX address bit 15 active will block this card.

APPENDIX D

NORD-100 PLUG PANEL FOR EXTERNAL DEVICE CONNECTION



APPENDIX E

ORGANIZATION OF NORD-100 MODULES

E.1 *LOCATION OF INTEGRATED CIRCUITS (CARD COORDINATES)*

To make it easy to find integrated circuits on the NORD-100 modules, the modules are equipped with a coordinate system. The coordinate system is built up like coordinates in a map with numbers and letters.

The integrated circuits are placed in rows and columns. As seen in Figure E.1, the rows are marked with letters from A through H. The colums are marked with numbers from 1 through 28. When you have the card in front of you, with the component side up and the plugs away from you (see the figure below), the rows are marked at the right short edge. The row letters start with A at the plug edge and end with H at the edge towards you. The column numbers are marked at the long edge towards you, starting with 1 at the right corner and ending up with 28 in the left corner.

The row letters and column numbers are printed on the module. Sometimes, however, the letters and numbers are not printed. This is because of space problem on the module. You may also find letters and numbers that are printed upside-down, but the coordinate system remains the same.

NOTE: The module is seen from the component side.

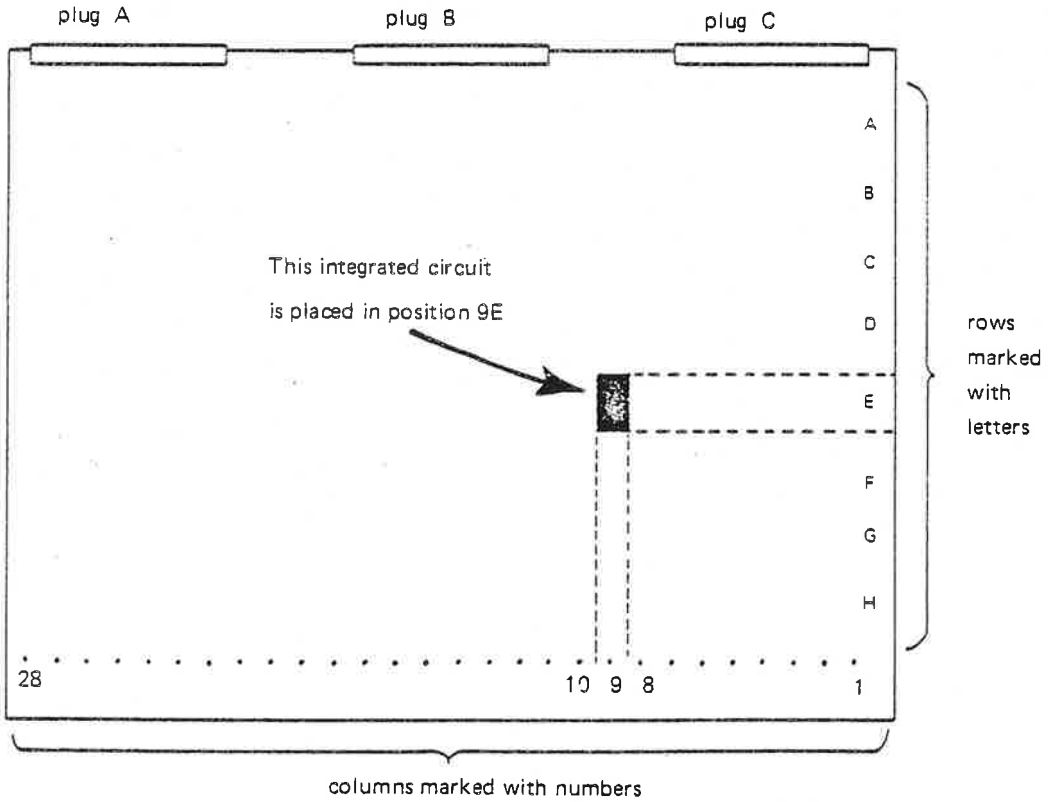


Figure E.1: Coordinate System on NORD-100 Modules.

On the schematics in Appendix G all integrated circuits are marked with a number and a letter (e.g., 9E). This is a reference to the coordinate system on the modules. To find the integrated circuit's physical position you simply use the coordinate system as shown in the figure below.

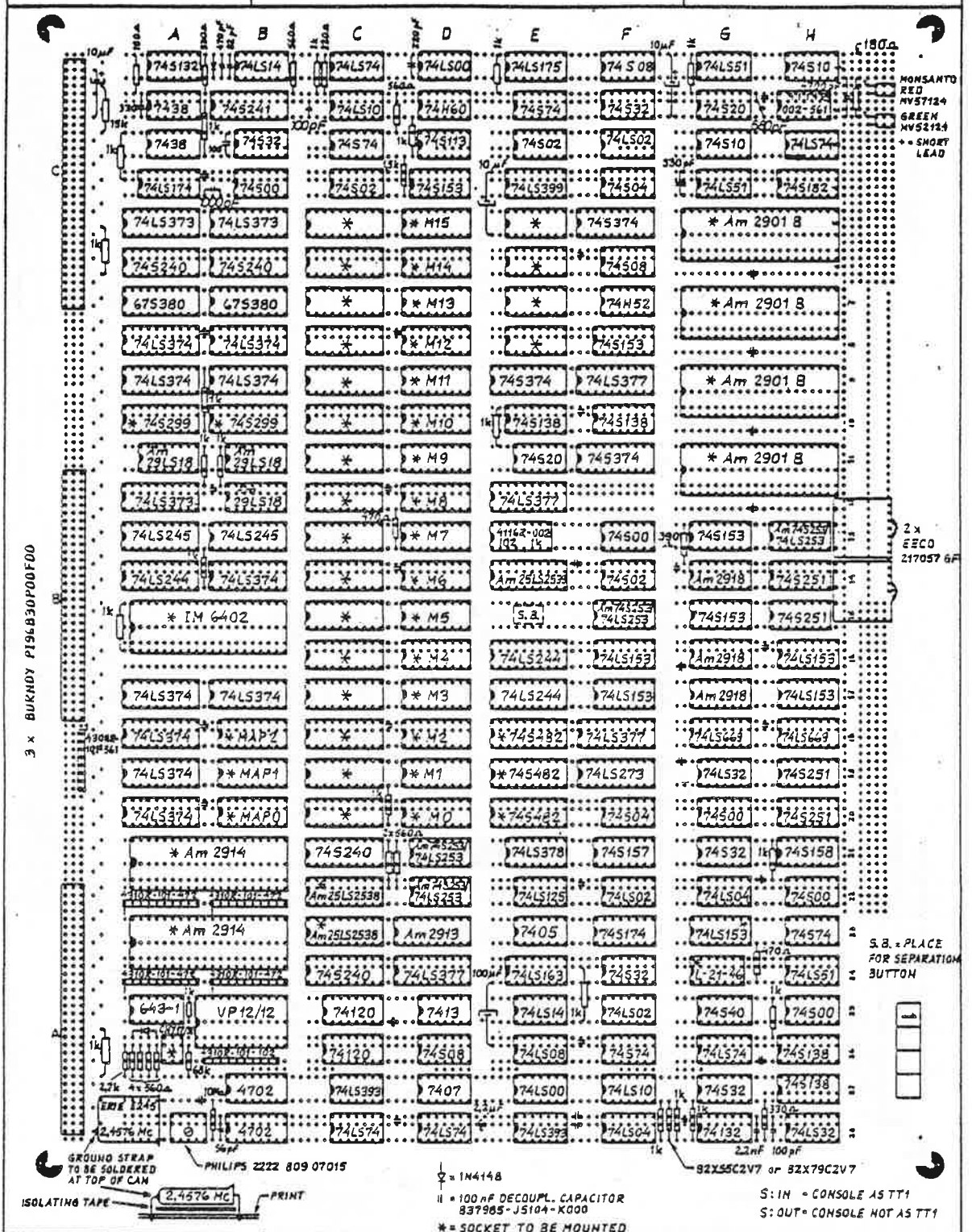
Each NORD-100 module has an arrangement drawing. Examples of arrangement drawings for NORD-100 modules are shown in Section E.2. The arrangement drawing for a NORD-100 module identifies the type of integrated circuit occupying each "coordinate position" on the module. The type of integrated circuit is written on the circuit at each location. The arrangement drawings have the same coordinate system as the modules. They are drawn from the component side with the plug edge to the left. Make sure you are looking at the arrangement drawing corresponding to the schematic where you found the integrated circuit's position, 9E.

Example:

For the NORD-100 CPU module, the integrated circuit in position 9E is labelled 74S374.

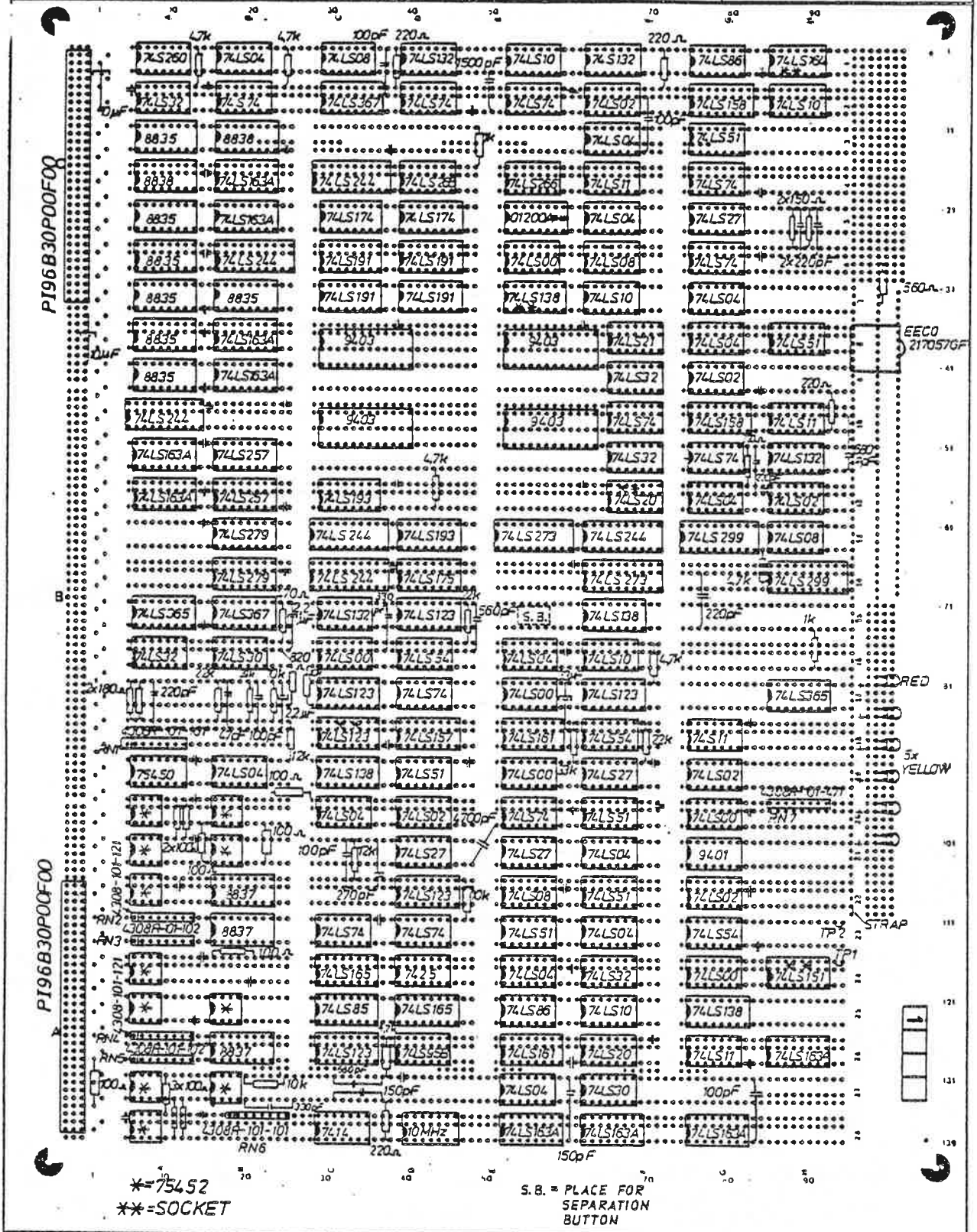
E.2 ARRANGEMENT DRAWINGS EXAMPLES

NORSK DATA A.S	TITLE: NORD-100 CPU	ID. NO. 322602	CARD PRINT N E



CORRECTIONS ACCORDING TO ECO			CORRECTIONS WITHOUT ECO.		
NO.	DATE / SIGN.	VERSION	NO.	DESCRIPTION	DATE / SIGN.
100-4	13. 2. 79 / Eml		79-01	Equiv. added on IC in 13H, 15F, 21D, 22D	06.11.79 / Eml
100-14	28.05.79 / Eml	EE → FE	79-05	74LS669A changed to 74LS669. 74S207 and 74LS208 removed.	23.11.79 / Eml
100-48 Rev.B	08.12.80 / Eml	FE → GE	80-07	74LS153 in 23G placed on socket	28/04/80 / BW
100-49	22.10.79 / Eml	GE → HE			
100-65	13.12.79 / Eml	HE → JE			
DESIGNED BY	I. B.	DRAWN BY	Eml	DATE 13. 2. 79	PRINT NO. 3002

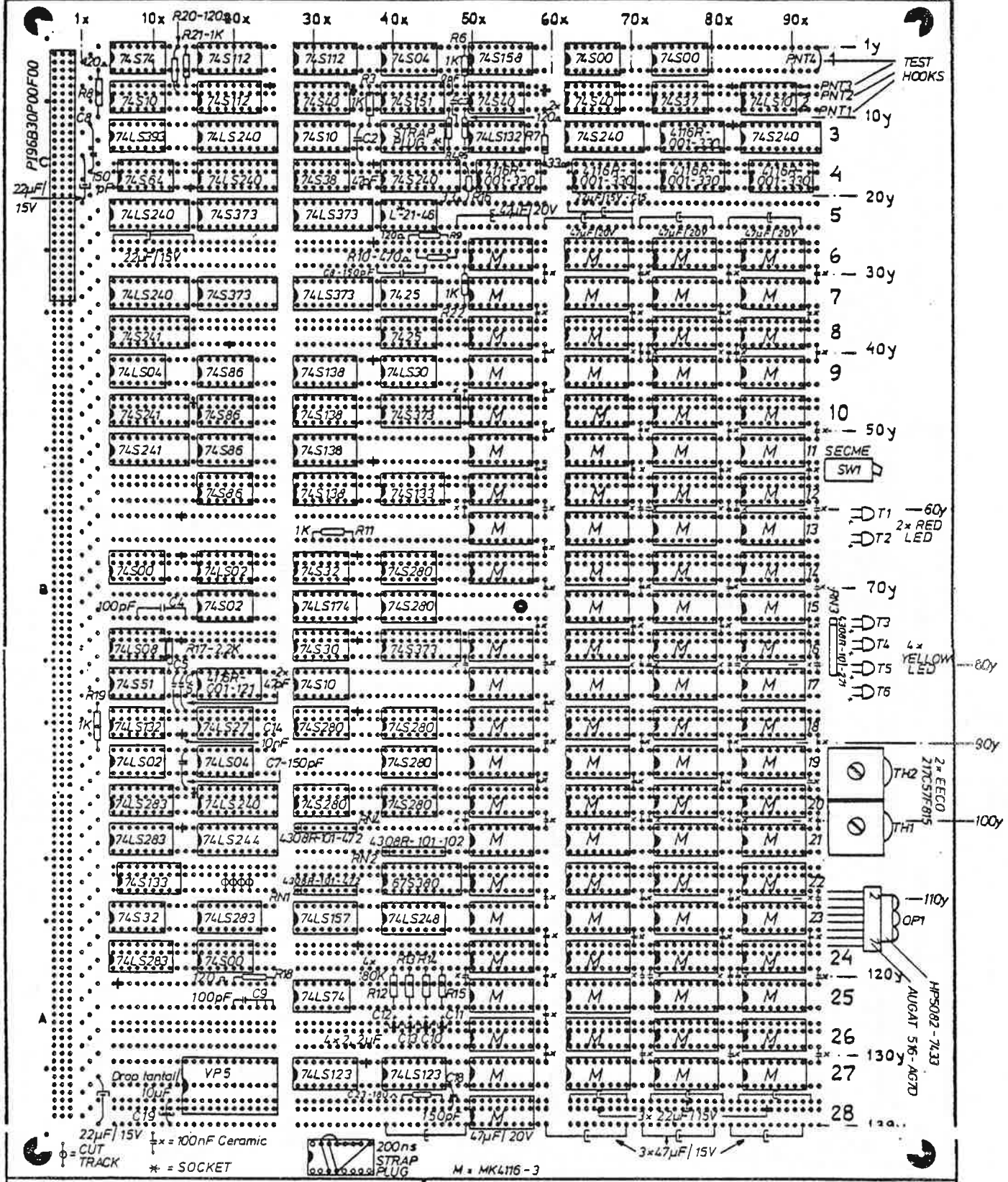
NORSK DATA A.S	TITLE: 10M BYTE DISK CONTROLLER	ID.NO. 322604	CARD PRINT N M



CORRECTIONS ACCORDING TO ECO			CORRECTIONS WITHOUT ECO		
NO.	DATE / SIGN.	VERSION	NO.	DESCRIPTION	DATE / SIGN.
100-108	01/07/80/AL	MM→NM			

DESIGNED BY AF/HS	DRAWN BY AL.	DATE 21/04/80	PRINT NO. 3004

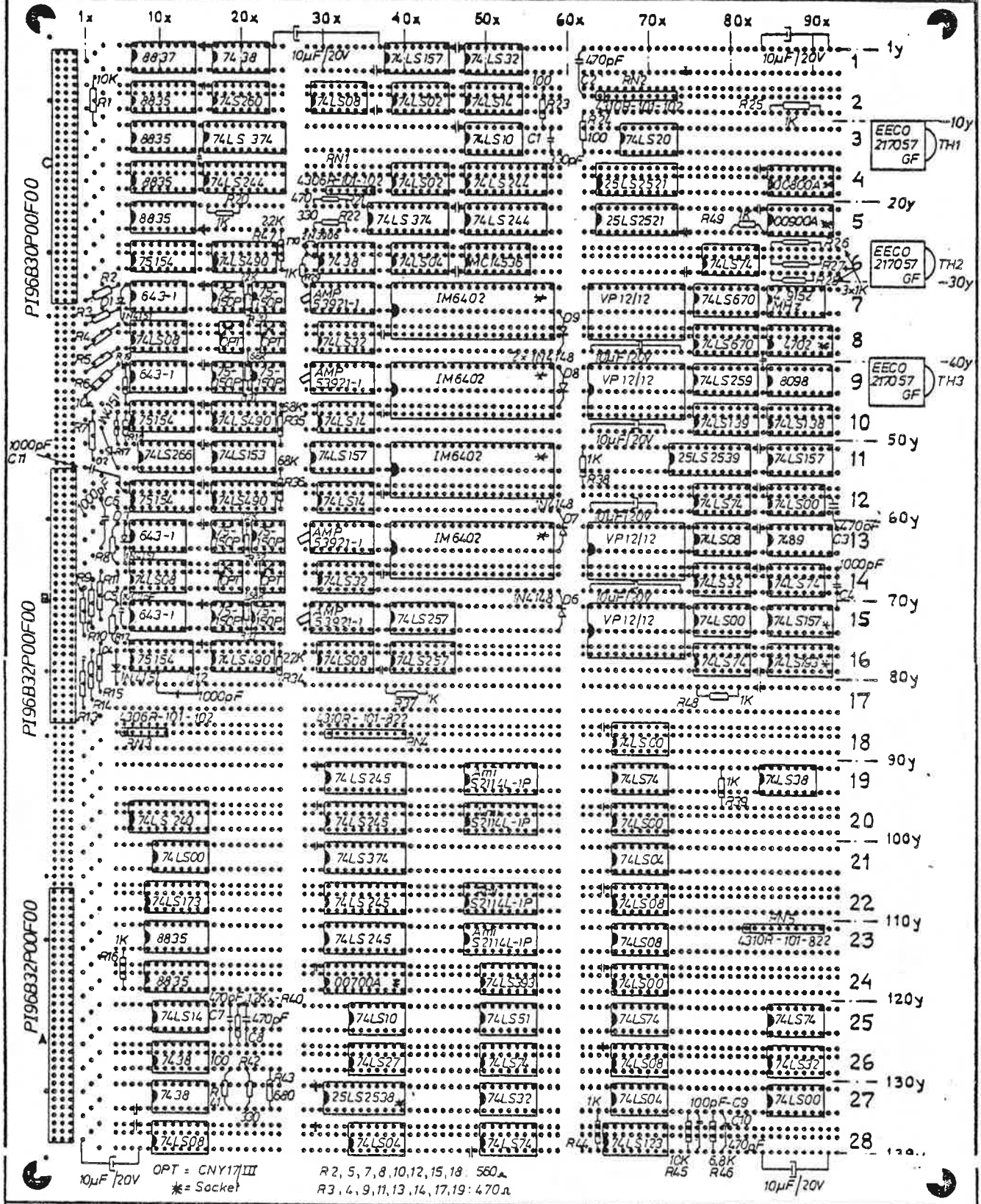
NORSK DATA A.S	TITLE: 64K DYNAMIC RAM	ID.NO.: 322605	CARD: H	PRINT: H.



CORRECTIONS ACCORDING TO E C O			CORRECTIONS WITHOUT E C O		
NO.	DATE/SIGN	VERSION	NO.	DESCRIPTION	DATE/SIGN

DESIGNED BY: PJJ	DRAWN BY: BW	DATE: 02/12/80	PRINT NO.: 3005

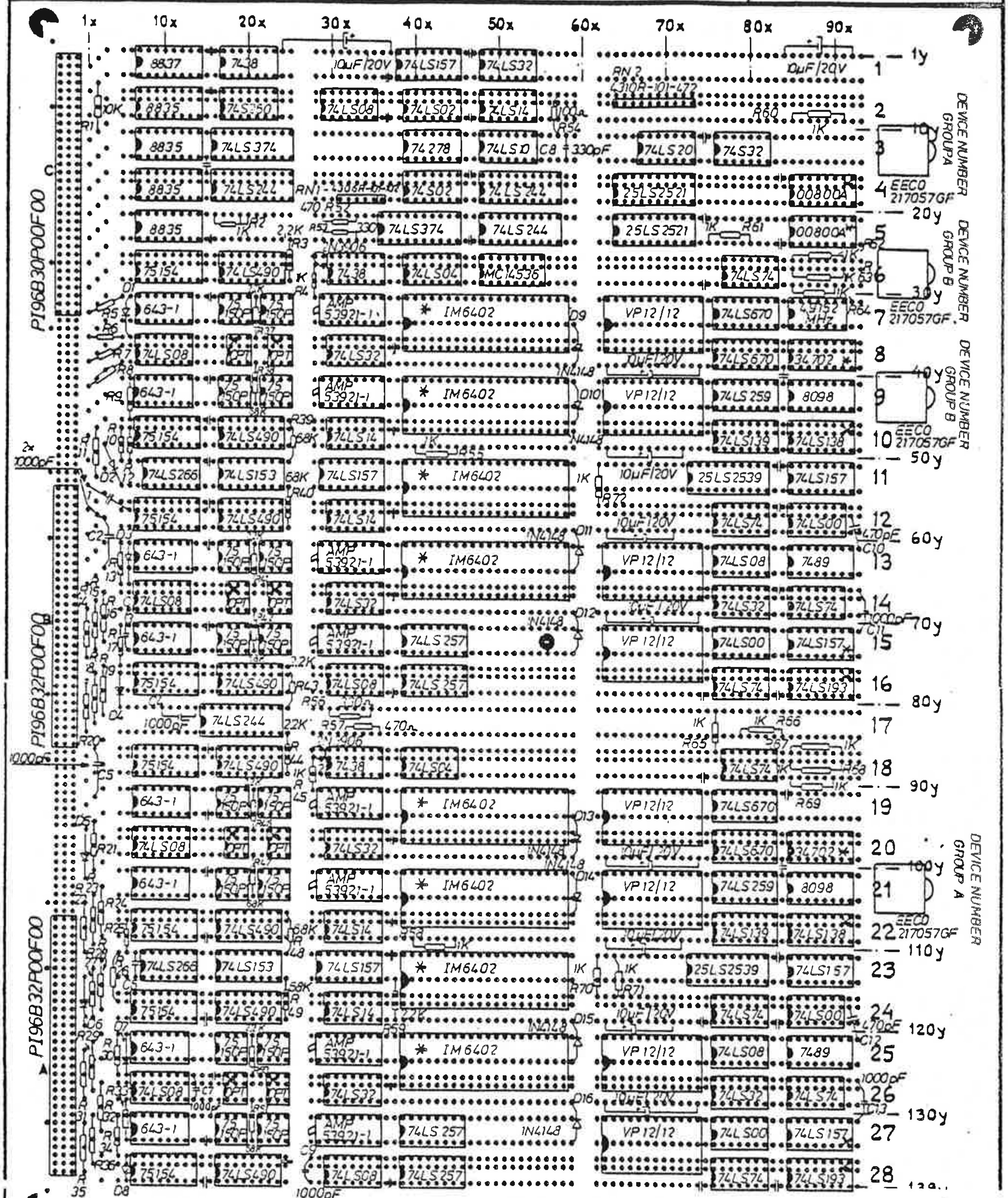
NORSK DATA A.S	TITLE FLOPPY AND 4 TERM. INTERFACE	ID NO. 322610	CARD PRINT N N



CORRECTIONS ACCORDING TO E C O			CORRECTIONS WITHOUT E C O		
NO.	DATE/SIGN	VERSION	NO.	DESCRIPTION	DATE/SIGN
			80-28	NUMBER OF SOCKETS CHANGED	14/11/80 /BW

DESIGNED BY PHE	DRAWN BY AL	DATE 04/11/80	PRINT NO 3010

NORSK DATA A.S	TITLE	ID. NO.	CARD	PRINT
	8-TERMINAL INTERFACE	322613	K	H



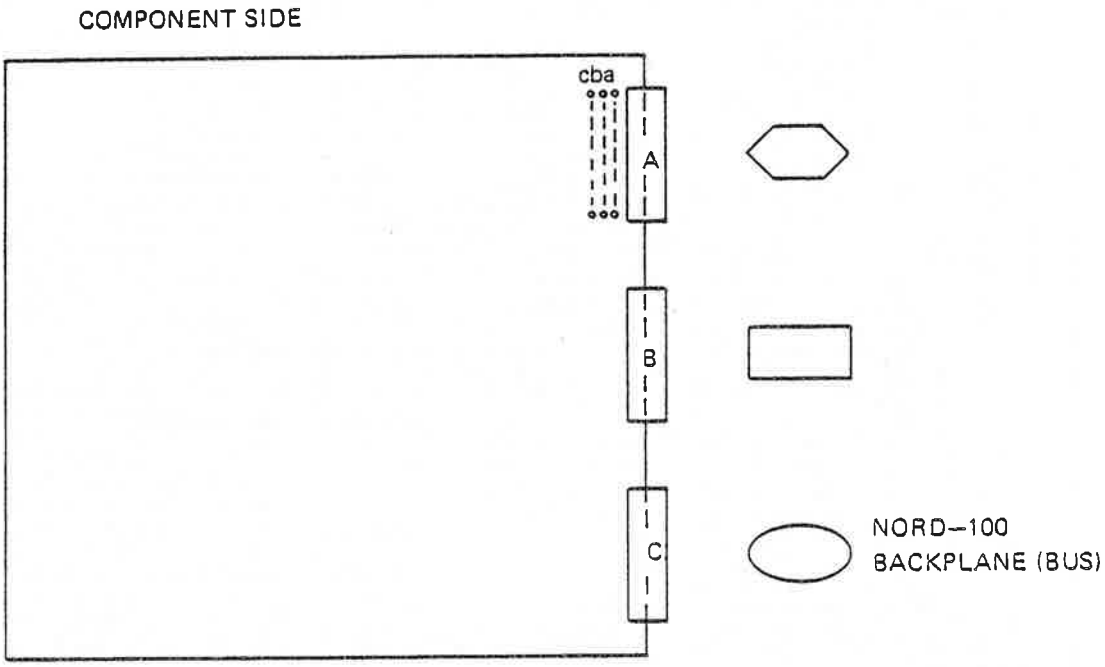
C3,C6 = 1000pF D1-D8 = 1N4151 OPT = CNY17/III * = SOCKET

R6, 7, 9, 10, 14, 16, 18, 20, 24, 26, 27, 28, 29, 31, 35, 36 : 470Ω
R5, 8, 11, 12, 13, 15, 17, 19, 21, 22, 23, 25, 30, 32, 33, 34 : 560Ω

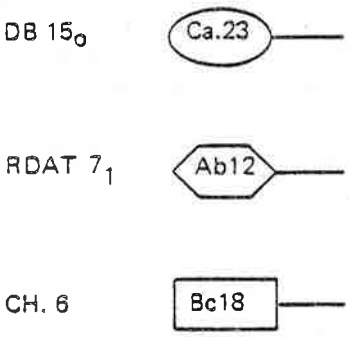
CORRECTIONS ACCORDING TO E C O			CORRECTIONS WITHOUT E C O		
NO.	DATE/SIGN	VERSION	NO.	DESCRIPTION	DATE/SIGN
100-118	02/09/80/ BW	HH→JH	80-30	Value of C7 added.	21.11.80/ [signature]
100-137	10/12/80/ AL	JH→KH			

DESIGNED BY	DRAWN BY	DATE	PRINT NO.
PHE	AL	02/09/80	3013

E.3 *NOTATION OF SIGNALS TO AND FROM NORD-100 MODULES*



Example:



E.4 *CABLE LIST FOR EXTERNAL DEVICE CONNECTION*

Example: 4 or 8 terminals/asynchronous modem connection.

NORSK DATA A.S	Title: 4 TERMINAL INTERFACES FOR N-100 TANDBERG TDV 2115	Drawing No.: 3-9322

WIRE NO.	SIGNAL	POLARITY	EUROPLUG PIN NO.	PLUG PANEL FEMALE EXT.CAB. MALE COVER HARTING 25 PIN PLUG PIN NO.	DEVICE PLUG EXT.CAB.FEMALE COVER MODEM 25S PLUG PIN NO.		
1	CHASSIS		c 1	1			
2			a 1	14			
3	TXD		c 2	2			
4	I1+		a 2	15	23		
5	RXD		c 3	3			
6	I2-		a 3	16	* 24-25		
7	RTS		c 4	4		TERMINAL	1
8	O1+		a 4	17	12		
9	RFS		c 5	5			
10	O2-		a 5	18	13		
11	DSR		c 6	6			
12			a 6	19			
13	GND		c 7	7			
14	DTR		a 7	20			
15	SIGN. DET.		c 8	8			
16			a 8	21			
17	CHASSIS		c 9	1			
18			a 9	14			
19	TXD		c 10	2			
20	I1+		a 10	15			
21	RXD		c 11	3			
22	I2-		a 11	16			
23	RTS		c 12	4		TERMINAL	2
24	O1+		a 12	17			
25	RFS		c 13	5			
26	O2-		a 13	18			
27	DSR		c 14	6			
28			a 14	19			
29	GND		c 15	7			
30	DTR		a 15	20			
31	SIGN. DET.		c 16	8			
32			a 16	21			
33	CHASSIS		c 17	1			
34			a 17	14			
35	TXD		c 18	2			
36	I1+		a 18	15			
37	RXD		c 19	3			
38	I2-		a 19	16			
39	RTS		c 20	4		TERMINAL	3
40	O1+		a 20	17			
41	RFS		c 21	5			
42	O2-		a 21	18			
43	DSR		c 22	6			
44			a 22	19			
45	GND		c 23	7			
46	DTR		a 23	20			
47	SIGN. DET.		c 24	8			
48			a 24	21			
49	CHASSIS		c 25	1			
50			a 25	14			
51	TXD		c 26	2			
52	I1+		a 26	15			
53	RXD		c 27	3			
54	I2-		a 27	16		TERMINAL	4
55	RTS		c 28	4			
56	O1+		a 28	17			
57	RFS		c 29	5			
58	O2-		a 29	18			
59	DSR		c 30	6			
60			a 30	19			
61	GND		c 31	7			
62	DTR		a 31	20			
63	SIGN. DET.		c 32	8			
64			a 32	21			

INTERNAL CABLE TYPE : 64 wire flatcable
EXTERNAL CABLE TYPE 1: 4 pair shielded round cable
EXTERNAL CABLE TYPE 2:
EXTERNAL CABLE TYPE 3:
EXTERNAL CABLE TYPE 4:

Pin 1 in the device plug is connected to cable. shield.
In the computer end of the external cable the shield is connected to an earth strap with faston.
Male plug in computer end of cable

* Strap in external cable.

Drawn by: HO/kha	Remarks: 8-connector on 3010 8 & A-connector on 3013	Replacement for	Date
Approved		Replaced by	Date
Date: 26.11.79			

NORSK DATA A.S

Title: 4 TERMINAL INTERFACE
ASYNCRON MODEM FOR N-100

Drawing No.: 3-9317

WIRE NO.	SIGNAL	POLARITY	EUROPLUG PIN NO.	PLUG PANEL female. Ext.Cab. male Cover Hartings 25 pin plug PIN NO.	DEVICE PLUG Plug 25 P Cover DB 51-226-1 PIN NO.	
1	CHASSIS		c 1	1		
2			a 1	¥ 14 -19		
3	TXD		c 2	2	2	
4	I1+		a 2	15		
5	RXD		c 3	3	3	
6	I2-		a 3	16		
7	RTS		c 4	4	4	TERMINAL 1
8	O1+		a 4	17		
9	RFS		c 5	5	5	
10	O2-		a 5	18		
11	DSR		c 6	6	6	
12			a 6	¥ 19 -14		
13	GND		c 7	7	7	
14	DTR		a 7	20	20	
15	SIGN. DET.		c 8	8	8	
16			a 8	21		
17	CHASSIS		c 9	1		
18			a 9	14		
19	TXD		c 10	2		
20	I1+		a 10	15		
21	RXD		c 11	3		
22	I2-		a 11	16		
23	RTS		c 12	4		TERMINAL 2
24	O1+		a 12	17		
25	RFS		c 13	5		
26	O2-		a 13	18		
27	DSR		c 14	6		
28			a 14	19		
29	GND		c 15	7		
30	DTR		a 15	20		
31	SIGN. DET.		c 16	8		
32			a 16	21		
33	CHASSIS		c 17	1		
34			a 17	14		
35	TXD		c 18	2		
36	I1+		a 18	15		
37	RXD		c 19	3		
38	I2-		a 19	16		
39	RTS		c 30	4		TERMINAL 3
40	O1+		a 20	17		
41	RFS		c 21	5		
42	O2-		a 21	18		
43	DSR		c 22	6		
44			a 22	19		
45	GND		c 23	7		
46	DTR		a 23	20		
47	SIGN. DET.		c 24	8		
48			a 24	21		
49	CHASSIS		c 25	1		
50			a 25	14		
51	TXD		c 26	2		
52	I1+		a 26	15		
53	RXD		c 27	3		
54	I2-		a 27	16		
55	RTS		c 28	4		TERMINAL 4
56	O1+		a 28	17		
57	RFS		c 29	5		
58	O2-		a 29	18		
59	DSR		c 30	6		
60			a 30	19		
61	GND		c 31	7		
62	DTR		a 31	20		
63	SIGN. DET.		c 32	8		
64			a 32	21		

INTERNAL CABLE TYPE : 64 wire flatcable
EXTERNAL CABLE TYPE 1: 4 pair round cable with screen
EXTERNAL CABLE TYPE 2:
EXTERNAL CABLE TYPE 3:
EXTERNAL CABLE TYPE 4:

Pin 1 in the device plug is connected to cable shield.
In the computer end of the external cable the shield is connected to an earth strap (faston).

¥ Strap in external cable

Drawn by	TF	Remarks	Replacement for	Date
Approved		B-connector on 3010 B & A-connector on 3013		
Date	20.11.79		Replaced by	Date

APPENDIX F

NORD-100 BUS BACKPLANE SIGNALS

A complete list of signal names and pin assignments is given in the NORD-100 Backplane table on the following page. All logic signals are in the active low polarity, i.e.,:

Logical "0" — 2.4 to 5.0 volts
Logical "1" — 0 to 0.5 volts

The normal TTL noise margins are assumed, but it is recommended to use receivers with hysteresis on sensitive inputs, especially on signals that are at "1" while the data bus switches.

Maximum load = 0.8 mA for each slot

	a	SOURCE	USED	b	SOURCE	USED	c	SOURCE	USED
1	GND	P	CMI	GND	P	CMI	GND		
2	+5V	P	CMI	+5V	P	CMI	+5V		
3	BD 1	CMI	CMI	BD 16	CMI	CMI	BD 0	CMI	CMI
4	BD 3	CMI	CMI	BD 17	CMI	CMI	BD 2	CMI	CMI
5	BD 5	CMI	CMI	BD 18	CMI	CMI	BD 4	CMI	CMI
6	BD 7	CMI	CMI	BD 19	CMI	CMI	BD 6	CMI	CMI
7	BD 9	CMI	CMI	BD 20	CMI	CMI	BD 8	CMI	CMI
8	BD 11	CMI	CMI	BD 21	CMI	CMI	BD 10	CMI	CMI
9	BD 13	CMI	CMI	BD 22	CMI	CMI	BD 12	CMI	CMI
10	BD 15	CMI	CMI	BD 23	CMI	CMI	BD 14	CMI	CMI
11	GND	P	CMI	GND	P	CMI	GND	P	CMI
12	BREF	C	M	LOAD	I	C	BREQ	I	C
13	PA 1		I	RESTART	I	C	PA 0		I
14	PA 3		I	RUN	C	I	PA 2		I
15	BINT 10	I	C	CONTINUE	I	C	BINT 11	I	C
16	BINT 12	I	C	STOP	I	C	BINT 13	I	C
17	PANREQ	I	C	BLANK	I	I	BINT 15	I	C
18	BINPUT	CI	CM	BPERR	M	CX	BDAP	CI	MI
19	BDRY	MI	CI	BINACK	C	I	BIOXE	C	I
20	BAPR	CI	MI	BMCL	CI	CMI	BMEM	C	MI
21	INCONTR	E	E	BERROR	E	E	OUTCONTR	E	E
22	INIDENT	CI	I	BCRQ	E	E	OUTIDENT	CI	I
23	INGRANT	CI	I	BMINH	P	CM	OUTGRANT	CI	I
24	GND	P	CMI	GND	P	CMI	GND	P	CMI
25	+15V	P	CMI	+15V	P	CMI	+15V	P	CMI
26	An. Return	P	I	An. Return	P	I	An. Return	P	I
27	-15V	P	CMI	-15V	P	CMI	-15V	P	CMI
28	+12V	P	CI	+12V	P	CI	+12V	P	CI
29	POW. SENSE	P	CI	POW. SENSE	P	CI	POW. SENSE	P	CI
30	5V St.by	P	CM	5V St.by	P	CM	5V St.by	P	CM
31	+5V	P	CMI	+5V	P	CMI	+5V	P	CMI
32	GND	P	CMI	GND	P	CMI	GND	P	CMI

NORD-100 Backplane Table

Codes for SOURCE and USED are:

C = CPU or Bus Receiver
M = Memory
I = I/O interface
P = Power supply unit
E = Future extensions only
X = Bus expander

All logic signals have active low TTL levels.

Logic signals, alphabetically:

An Return

Separate ground line for analogue circuits. Connected to logic ground return (GND) in power supply end. (± 15V Power Supply).

BAPR

Bus Address Present on multiplexed data and address bus. Wired OR line.

BCRQ

Bus Control Request from source wanting full control over bus (for future extensions). Wired OR line.

BDRY

Bus Data Ready signals that data are ready or have been accepted, given by answering device. Wired OR line.

BD0-23

Multiplexed data and address bus. Bit 0 is least significant.

BERROR

Bus Error signals that an error was detected during a bus cycle, e.g., fatal memory error. Wired OR line.

BINACK

Bus Input Acknowledge signals that an interface requesting an input operation may enable data. Generated by controlling unit.

BINPUT

Bus Input signalled by a unit which will transmit data. I/O interfaces must wait for BINACK before enabling data and BDRY. Wired OR line.

BINT10-13, 15

Interrupt lines. BINT10 has lowest priority. Wired OR lines.

BDAP

Bus Data Present signals that data is present during DMA or memory cycles.

BIOXE

Input/Output Enable. A strobe to enable data transfer to or from an I/O interface. Generated by controlling unit.

BLANK

Output Blanking signal for process interface. Wired OR signal generated by monitoring device.

BMCL

Bus Master Clear for logic initialization at power up and when Master Clear button is pushed. Wired OR line.

BMEM

Bus Memory Cycle signals that a bus cycle accesses memory. Generated by controlling unit.

BMINH

Bus Memory Inhibit used to inhibit memory accesses during power down and power up sequence in systems which have battery backup for memory only. Generated by controlling unit.

BPERR

Bus Parity error. Fatal or correctable error from memory, according to the ECC register.

BREQ

Request for a DMA cycle. Wired OR line.

CONTINUE

May be used to start a CPU that is in STOP mode. Wired OR line. (Only in CPU crate.)

GND

Logical ground return.

INCONTR

Response to BCRQ indicating that a control over the bus is available. A unit which does not want to control the bus must issue OUTCONTR is response to INCONTR. INCONTR is generated as OUTCONTR by nearest unit in a less significant board position.

INGRANT

Response to BREQ, indicating that the bus is available for a DMA cycle. An interface which issued BREQ prior to the last leading edge of BMEM may use the bus for a single memory read or write cycle. Otherwise, INGRANT is passed onto OUTGRANT which is connected to INGRANT of the next lower priority card position (further removed from controlling unit). INGRANT originates as OUTGRANT from controlling unit.

INIDENT

Response to BINT10-13, together with address bits 0-5 which specify BINT number. An interface which issued BINT on the specified level prior to the last leading edge of BAPR shall respond by enabling its IDENT CODE onto the BD bus. Otherwise, INIDENT is passed on to OUTIDENT which is connected to INIDENT of the next lower priority card position (further removed from controlling unit). INIDENT originates in the OUTIDENT from controlling unit.

PA0-3

Card position code and defines device numbers of analogue and digital process interfaces.

LOAD

Activates the load microprogram if the CPU is in STOP mode. Wired OR line (only in CPU crate).

OUTCONTR

See INCONTR.

OUTGRANT

See INGRANT.

OUTIDENT

See INIDENT.

RESTART

Starts program execution in location 20_8 if the CPU is in STOP mode. Wired OR line (only in CPU crate).

RUN

Indicates that the CPU is active executing a program, i.e., *not* in STOP mode. Generated by CPU (only in CPU crate).

SPARE

Not assigned.

STOP

Forces the CPU to enter STOP mode after completion of the current instruction. Wired OR line (only in CPU crate).

+5V

Main logic supply voltage.

+5V Standby

Logic supply voltage for memory retention during power fail.

+15V

Supply voltage for analogue interface circuits. For customer use.

–15V

Supply voltage for analogue interface circuits. For customer use.

+12V Standby

Supply voltage for memory. Requires battery backup.

F.1 *REPRESENTATION OF SIGNALS IN TIMING DIAGRAMS*

Bus Organization

The NORD-100 bus is arranged as printed backplane carrying a total of 96 lines (power and ground lines included).

Each line in the backplane is defined in accordance to the manual "NORD-100 Backplane Manual" (available on special request).

The bus may be divided up into two logical parts:

- 24 bit parallel multiplexed address and data bus
- control lines

Naming of Bus Lines

The Data and Address Lines

The lines in the 24 bits multiplexed address and data bus is named BD0-23.

BD0 denotes the least significant bit while BD23 denotes the most significant bit.

The Control Lines

Each control line has their own unique name and function (see Appoendix F.2).

Generally, a signal name is an abbreviation of its function preceeded by a B for Bus (example, BREQ – Bus REQuest).

Signal Polarity

As stated in Section F.2, all logical signals are active in low polarity, i.e.:

Logical "0" – 2.4 to 5.0 volts
Logical "1" – 0.0 to 0.5 volts

To define whether a signal is active in low or high polarity, the signal name is associated with a suffix 0 (active low) or 1 (active high).

Example:

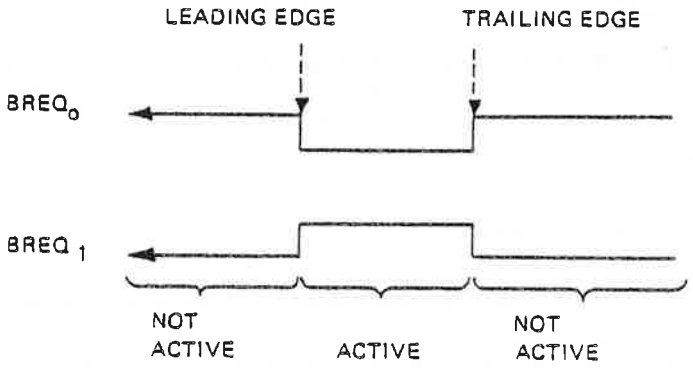
$BREQ_0$ — Bus REQuest is active (true) in low polarity.

Convention

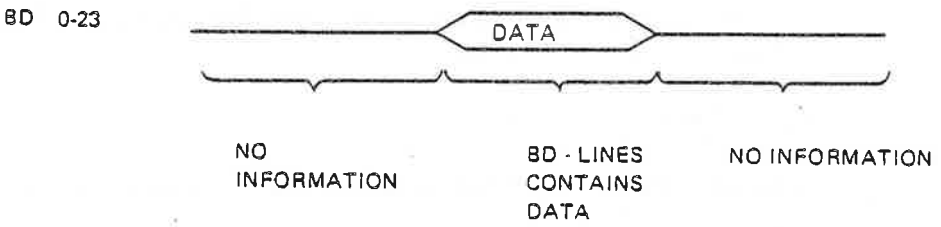
$BREQ_0 = \overline{BREQ_1}$

Representation of Signals in Timing Diagrams

The representation of control signals in a timing diagram is as shown below.



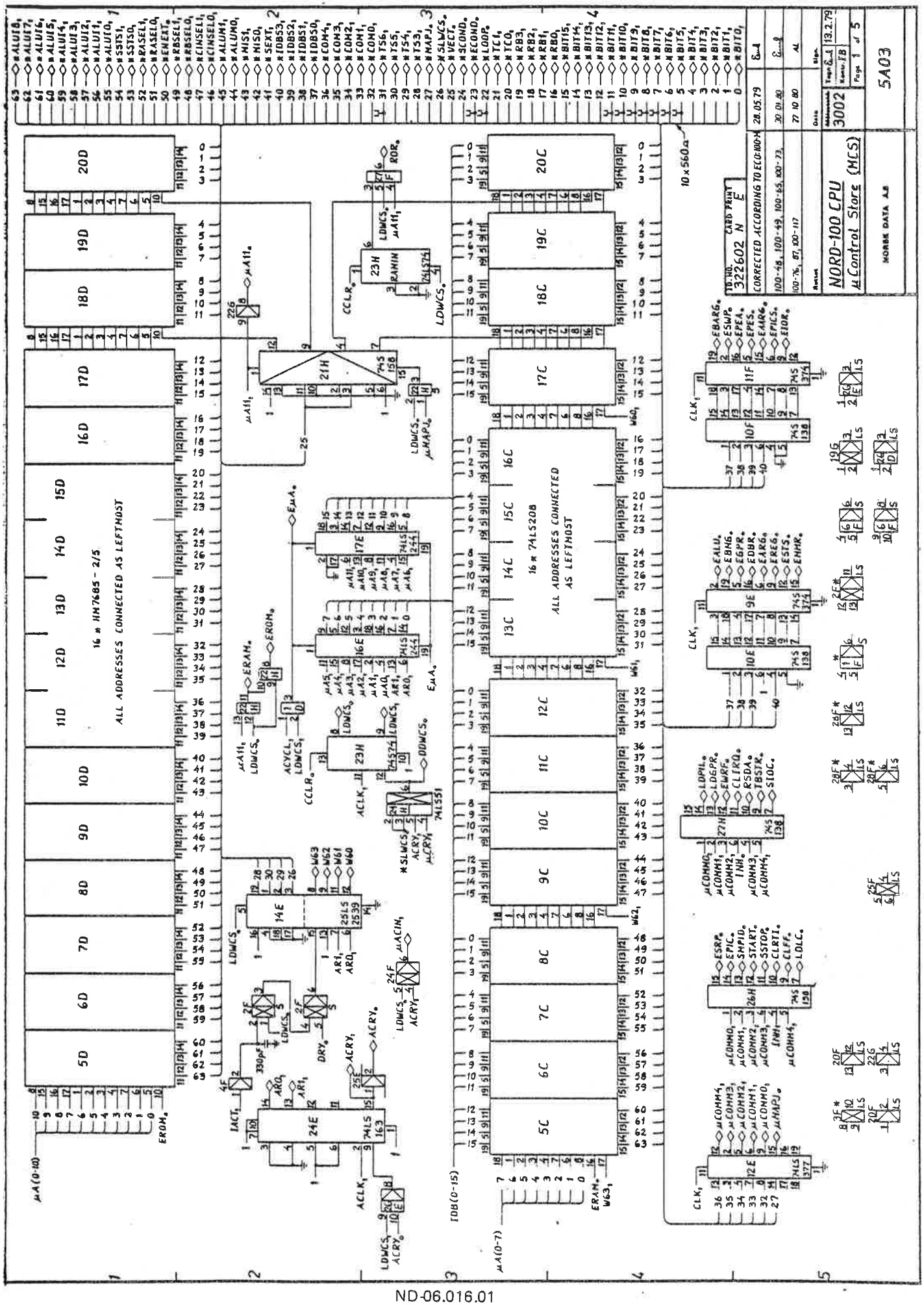
The BD0-23 lines are represented as shown below.

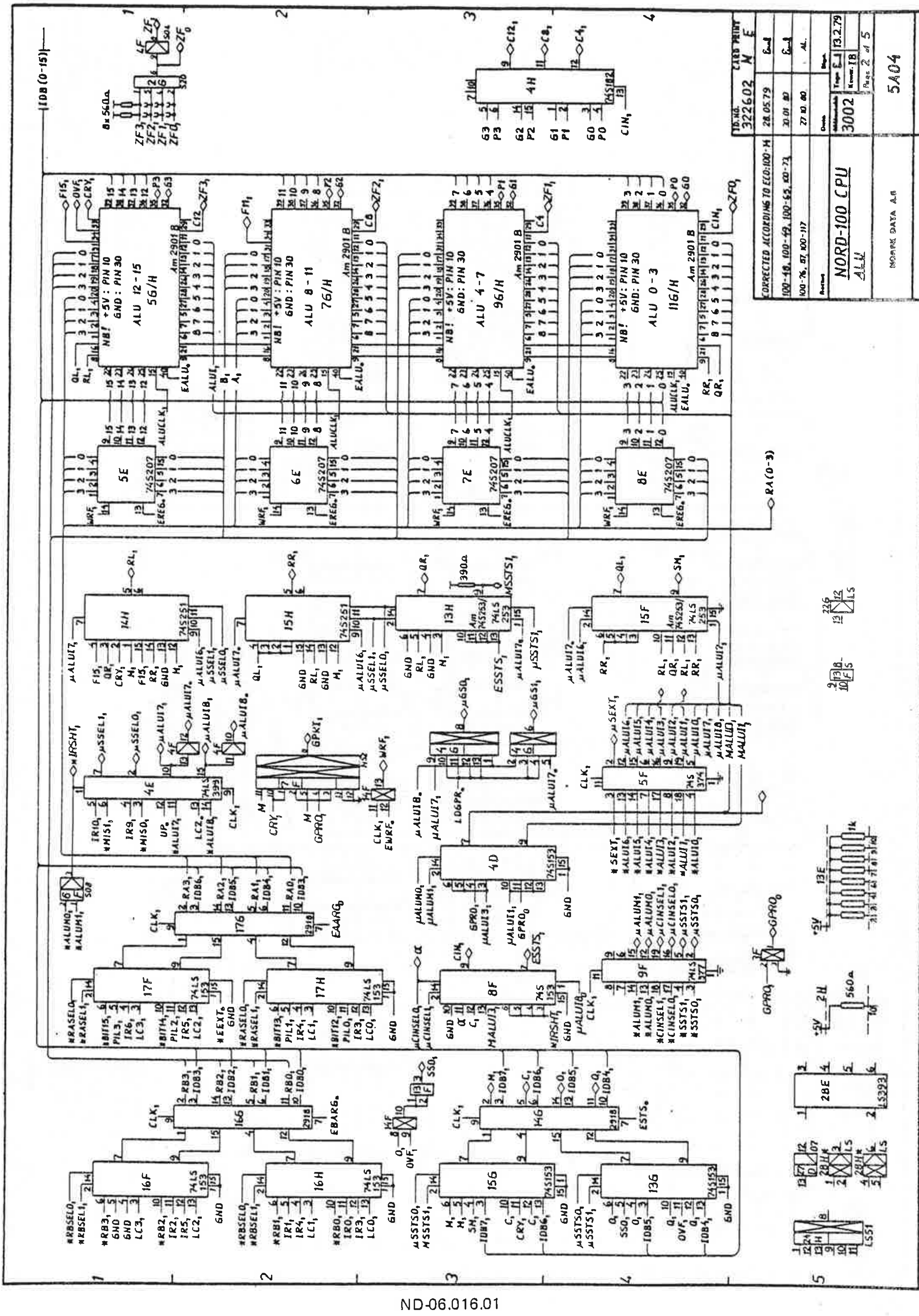


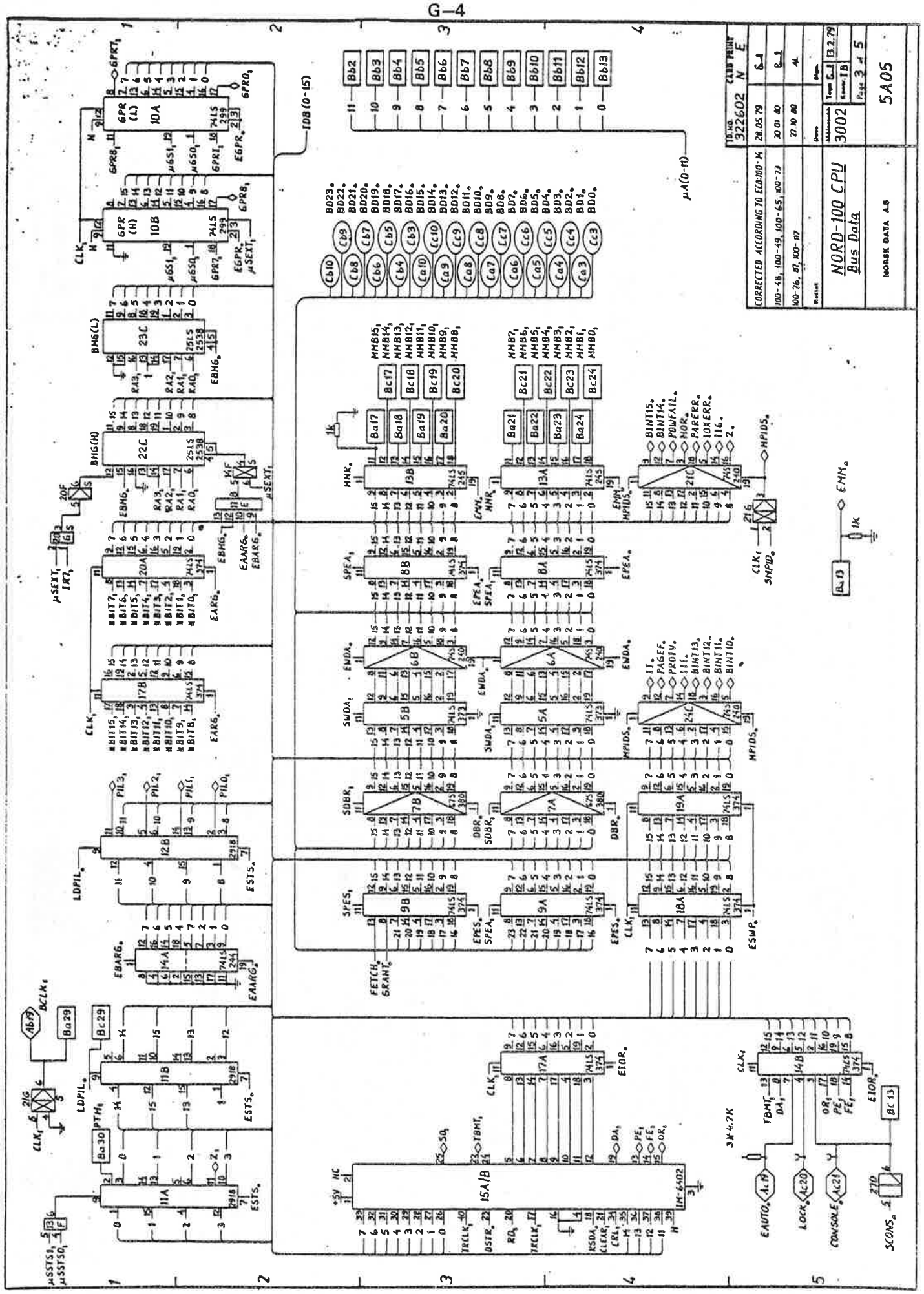
APPENDIX G

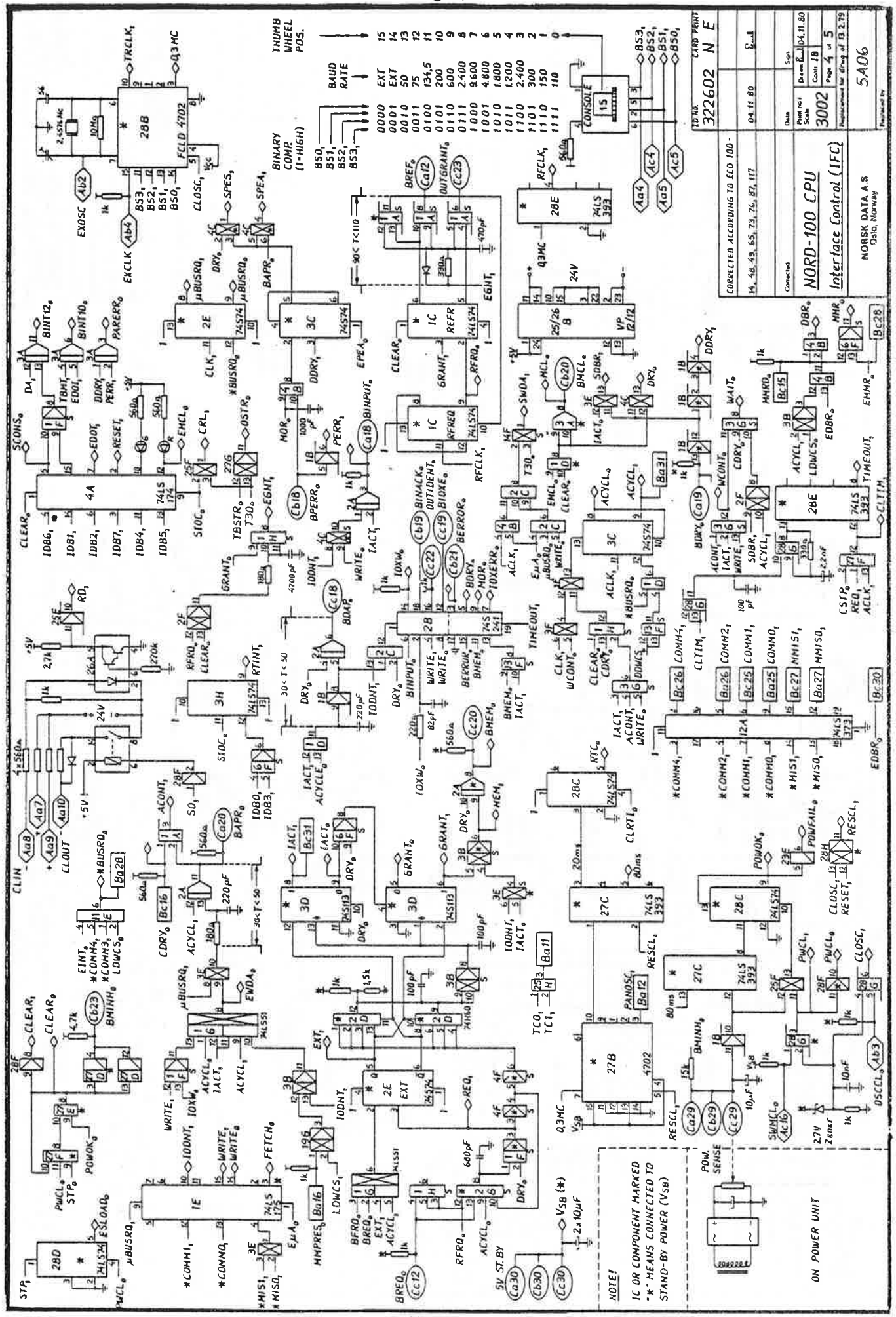
SCHEMATICS

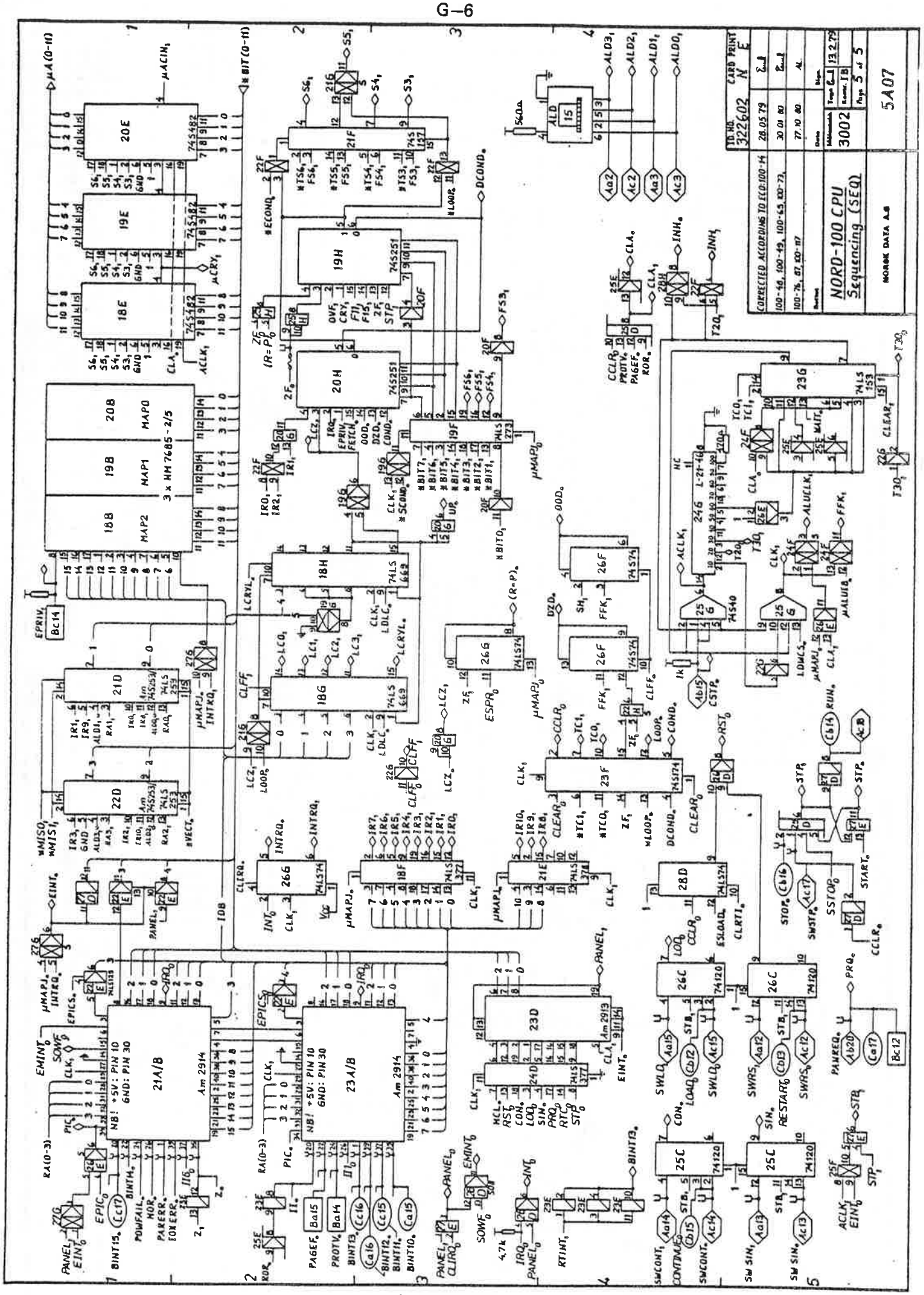
G.1 NORD-100 CPU



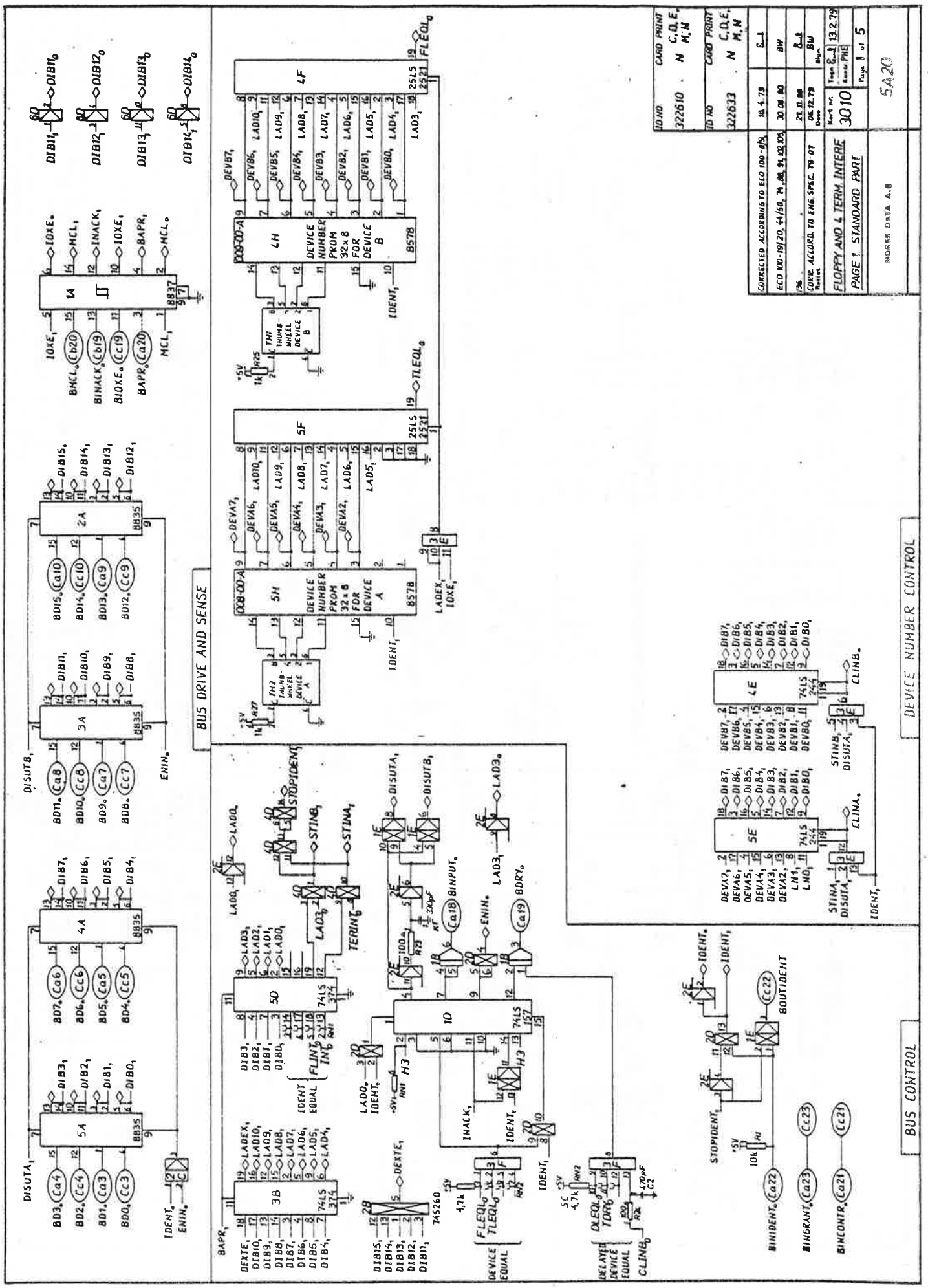

NORD-100 CPU
ALU
5A04
322602
3002
CORRECTED ACCORDING TO ECO:100-14
ALU 12-15
ALU 8-11
ALU 4-7
ALU 0-3
Am 2901 B

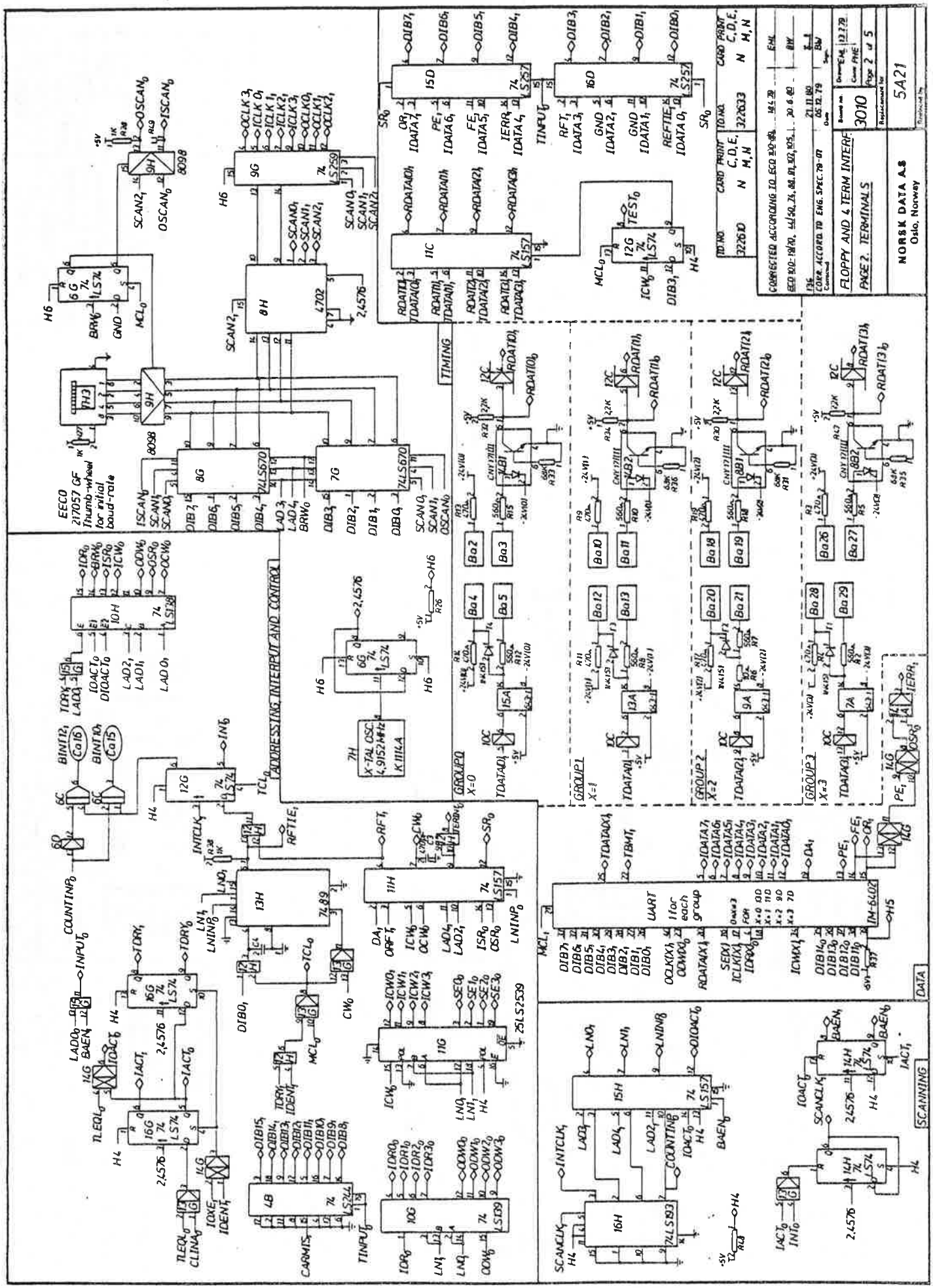

NORD-100 CPU
Bus Data
3002
5A05
NORSK DATA A.S
CORRECTED ACCORDING TO ECO 100-M
100-48, 100-49, 100-65, 100-73
100-76, 87, 100-87
28.05.79
20.01.80
27.10.80
Page 3 of 5
IDB (0-15)
μA (0-11)
BD0. BD1. BD2. BD3. BD4. BD5. BD6. BD7. BD8. BD9. BD10. BD11. BD12. BD13. BD14. BD15. BD16. BD17. BD18. BD19. BD20. BD21. BD22. BD23.
MM0. MM1. MM2. MM3. MM4. MM5. MM6. MM7. MM8. MM9. MM10. MM11. MM12. MM13. MM14. MM15.
BINT15. BINT14. POWFAIL. MOR. PARERR. IOXERR. II6. Z.
II. PAGEF. PROTV. III. BINT13. BINT12. BINT11. BINT10.
HPIDS.
EMM0.
1K
3K 4.7K
EAUTO. LOCK. CONSOLE. SCONS.
270
ESWP.
FETCH. GRANT.
EIOR.
EARG.
EBARG.
EWDA.
SWDA.
SDBR.
DBR.
SPEA.
EPEA.
SPES.
EPES.
LDPIL.
ESTS.
EBMG.
EMMR.
μSEXT.
GPR (L) 10A
GPR (H) 10B
23C
22C
20A
17B
12B
11B
11A
9B
9A
8B
8A
7B
7A
6B
6A
5B
5A
14A
20F
21C
24C
19A
18A
17A
15A/B
13B
13A
IM-6402

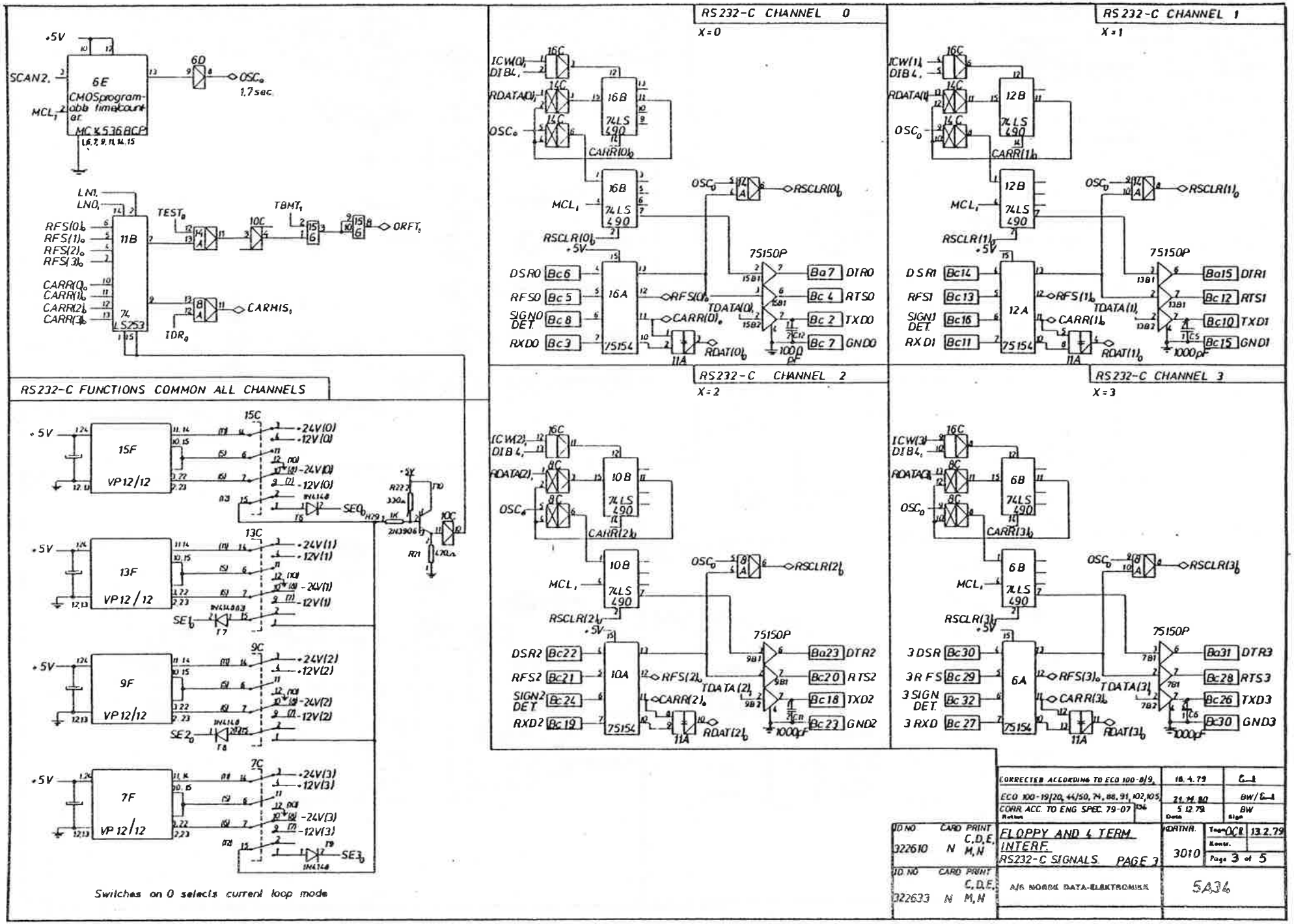

NORD-100 CPU
Interface Control (IFC)
3002
5406
322602 N E
NORSK DATA A.S
Oslo, Norway
NOTE!
IC OR COMPONENT MARKED
"*" MEANS CONNECTED TO
STAND-BY POWER (VSB)
ON POWER UNIT
POW. SENSE
THUMB WHEEL POS.
BAUD RATE
BINARY COMP. (1=HIGH)
CLIN
CLOUT
CLEAR
EXCLK
EXOSC
TRCLK
Q3MC
RFCLK
SCONS
EHCL
CRL
RESET
EDOT
EBINT
PAREER
BINT12
BINT10
SPES
SPEA
BAPR
BPERR
PERR
BINPUT
BREF
OUTGRANT
EGNT
RFREQ
RFCLK
BINACK
OUTIDENT
BIOXE
BERROR
BERREQ
BDRY
MOR
IOXERR
IOXW
IODNT
IACT
WRITE
GRANT
BREQ
RREQ
RFRQ
BUSRQ
EINT
BHINH
EWDA
FETCH
POWOK
POWFAIL
PANOSC
RESCL
CLRT
TIMEOUT
CLTIM
WCONT
SDBR
ACYCL
DRY
BMEM
HEM
BEMEM
IACYCLE
CONSOLE
MMR
MMRD
EDBR
DBE
WAIT
SWDA
SWMCL
VSB (*)
2x10µF
10nF
27V Zener
10µF
15k
1k
1,5k
2,7k
4,7k
220k
100pF
220pF
680pF
4700pF
2,2nF
82pF
1000pF
470pF
330Ω
180Ω
560Ω
24V
5V
Ba11
Ba12
Ba16
Ba28
Ba31
Bc15
Bc16
Bc26
Bc27
Bc30
Ca10
Ca12
Ca18
Ca19
Ca20
Ca23
Ca29
Ca30
Cb18
Cb19
Cb20
Cb21
Cb23
Cb29
Cb30
Cc12
Cc18
Cc19
Cc20
Cc22
Cc23
Cc29
Cc30
Aa4
Aa5
Aa7
Aa8
Aa9
Aa10
Ab2
Ab3
Ab4
Ac4
Ac5
Ac16
Ac30
Bc28
Bc29
Bc31
Bc25
Ba25
Ba26
Ba27
Ba29
Ba30
Ba31
Ba32
Ba33
Ba34
Ba35
Ba36
Ba37
Ba38
Ba39
Ba40
Ba41
Ba42
Ba43
Ba44
Ba45
Ba46
Ba47
Ba48
Ba49
Ba50
Ba51
Ba52
Ba53
Ba54
Ba55
Ba56
Ba57
Ba58
Ba59
Ba60
Ba61
Ba62
Ba63
Ba64
Ba65
Ba66
Ba67
Ba68
Ba69
Ba70
Ba71
Ba72
Ba73
Ba74
Ba75
Ba76
Ba77
Ba78
Ba79
Ba80
Ba81
Ba82
Ba83
Ba84
Ba85
Ba86
Ba87
Ba88
Ba89
Ba90
Ba91
Ba92
Ba93
Ba94
Ba95
Ba96
Ba97
Ba98
Ba99
Ba100
Ba101
Ba102
Ba103
Ba104
Ba105
Ba106
Ba107
Ba108
Ba109
Ba110
Ba111
Ba112
Ba113
Ba114
Ba115
Ba116
Ba117
Ba118
Ba119
Ba120
Ba121
Ba122
Ba123
Ba124
Ba125
Ba126
Ba127
Ba128
Ba129
Ba130
Ba131
Ba132
Ba133
Ba134
Ba135
Ba136
Ba137
Ba138
Ba139
Ba140
Ba141
Ba142
Ba143
Ba144
Ba145
Ba146
Ba147
Ba148
Ba149
Ba150
Ba151
Ba152
Ba153
Ba154
Ba155
Ba156
Ba157
Ba158
Ba159
Ba160
Ba161
Ba162
Ba163
Ba164
Ba165
Ba166
Ba167
Ba168
Ba169
Ba170
Ba171
Ba172
Ba173
Ba174
Ba175
Ba176
Ba177
Ba178
Ba179
Ba180
Ba181
Ba182
Ba183
Ba184
Ba185
Ba186
Ba187
Ba188
Ba189
Ba190
Ba191
Ba192
Ba193
Ba194
Ba195
Ba196
Ba197
Ba198
Ba199
Ba200
Ba201
Ba202
Ba203
Ba204
Ba205
Ba206
Ba207
Ba208
Ba209
Ba210
Ba211
Ba212
Ba213
Ba214
Ba215
Ba216
Ba217
Ba218
Ba219
Ba220
Ba221
Ba222
Ba223
Ba224
Ba225
Ba226
Ba227
Ba228
Ba229
Ba230
Ba231
Ba232
Ba233
Ba234
Ba235
Ba236
Ba237
Ba238
Ba239
Ba240
Ba241
Ba242
Ba243
Ba244
Ba245
Ba246
Ba247
Ba248
Ba249
Ba250
Ba251
Ba252
Ba253
Ba254
Ba255
Ba256
Ba257
Ba258
Ba259
Ba260
Ba261
Ba262
Ba263
Ba264
Ba265
Ba266
Ba267
Ba268
Ba269
Ba270
Ba271
Ba272
Ba273
Ba274
Ba275
Ba276
Ba277
Ba278
Ba279
Ba280
Ba281
Ba282
Ba283
Ba284
Ba285
Ba286
Ba287
Ba288
Ba289
Ba290
Ba291
Ba292
Ba293
Ba294
Ba295
Ba296
Ba297
Ba298
Ba299
Ba300
Ba301
Ba302
Ba303
Ba304
Ba305
Ba306
Ba307
Ba308
Ba309
Ba310
Ba311
Ba312
Ba313
Ba314
Ba315
Ba316
Ba317
Ba318
Ba319
Ba320
Ba321
Ba322
Ba323
Ba324
Ba325
Ba326
Ba327
Ba328
Ba329
Ba330
Ba331
Ba332
Ba333
Ba334
Ba335
Ba336
Ba337
Ba338
Ba339
Ba340
Ba341
Ba342
Ba343
Ba344
Ba345
Ba346
Ba347
Ba348
Ba349
Ba350
Ba351
Ba352
Ba353
Ba354
Ba355
Ba356
Ba357
Ba358
Ba359
Ba360
Ba361
Ba362
Ba363
Ba364
Ba365
Ba366
Ba367
Ba368
Ba369
Ba370
Ba371
Ba372
Ba373
Ba374
Ba375
Ba376
Ba377
Ba378
Ba379
Ba380
Ba381
Ba382
Ba383
Ba384
Ba385
Ba386
Ba387
Ba388
Ba389
Ba390
Ba391
Ba392
Ba393
Ba394
Ba395
Ba396
Ba397
Ba398
Ba399
Ba400
Ba401
Ba402
Ba403
Ba404
Ba405
Ba406
Ba407
Ba408
Ba409
Ba410
Ba411
Ba412
Ba413
Ba414
Ba415
Ba416
Ba417
Ba418
Ba419
Ba420
Ba421
Ba422
Ba423
Ba424
Ba425
Ba426
Ba427
Ba428
Ba429
Ba430
Ba431
Ba432
Ba433
Ba434
Ba435
Ba436
Ba437
Ba438
Ba439
Ba440
Ba441
Ba442
Ba443
Ba444
Ba445
Ba446
Ba447
Ba448
Ba449
Ba450
Ba451
Ba452
Ba453
Ba454
Ba455
Ba456
Ba457
Ba458
Ba459
Ba460
Ba461
Ba462
Ba463
Ba464
Ba465
Ba466
Ba467
Ba468
Ba469
Ba470
Ba471
Ba472
Ba473
Ba474
Ba475
Ba476
Ba477
Ba478
Ba479
Ba480
Ba481
Ba482
Ba483
Ba484
Ba485
Ba486
Ba487
Ba488
Ba489
Ba490
Ba491
Ba492
Ba493
Ba494
Ba495
Ba496
Ba497
Ba498
Ba499
Ba500
Ba501
Ba502
Ba503
Ba504
Ba505
Ba506
Ba507
Ba508
Ba509
Ba510
Ba511
Ba512
Ba513
Ba514
Ba515
Ba516
Ba517
Ba518
Ba519
Ba520
Ba521
Ba522
Ba523
Ba524
Ba525
Ba526
Ba527
Ba528
Ba529
Ba530
Ba531
Ba532
Ba533
Ba534
Ba535
Ba536
Ba537
Ba538
Ba539
Ba540
Ba541
Ba542
Ba543
Ba544
Ba545
Ba546
Ba547
Ba548
Ba549
Ba550
Ba551
Ba552
Ba553
Ba554
Ba555
Ba556
Ba557
Ba558
Ba559
Ba560
Ba561
Ba562
Ba563
Ba564
Ba565
Ba566
Ba567
Ba568
Ba569
Ba570
Ba571
Ba572
Ba573
Ba574
Ba575
Ba576
Ba577
Ba578
Ba579
Ba580
Ba581
Ba582
Ba583
Ba584
Ba585
Ba586
Ba587
Ba588
Ba589
Ba590
Ba591
Ba592
Ba593
Ba594
Ba595
Ba596
Ba597
Ba598
Ba599
Ba600
Ba601
Ba602
Ba603
Ba604
Ba605
Ba606
Ba607
Ba608
Ba609
Ba610
Ba611
Ba612
Ba613
Ba614
Ba615
Ba616
Ba617
Ba618
Ba619
Ba620
Ba621
Ba622
Ba623
Ba624
Ba625
Ba626
Ba627
Ba628
Ba629
Ba630
Ba631
Ba632
Ba633
Ba634
Ba635
Ba636
Ba637
Ba638
Ba639
Ba640
Ba641
Ba642
Ba643
Ba644
Ba645
Ba646
Ba647
Ba648
Ba649
Ba650
Ba651
Ba652
Ba653
Ba654
Ba655
Ba656
Ba657
Ba658
Ba659
Ba660
Ba661
Ba662
Ba663
Ba664
Ba665
Ba666
Ba667
Ba668
Ba669
Ba670
Ba671
Ba672
Ba673
Ba674
Ba675
Ba676
Ba677
Ba678
Ba679
Ba680
Ba681
Ba682
Ba683
Ba684
Ba685
Ba686
Ba687
Ba688
Ba689
Ba690
Ba691
Ba692
Ba693
Ba694
Ba695
Ba696
Ba697
Ba698
Ba699
Ba700
Ba701
Ba702
Ba703
Ba704
Ba705
Ba706
Ba707
Ba708
Ba709
Ba710
Ba711
Ba712
Ba713
Ba714
Ba715
Ba716
Ba717
Ba718
Ba719
Ba720
Ba721
Ba722
Ba723
Ba724
Ba725
Ba726
Ba727
Ba728
Ba729
Ba730
Ba731
Ba732
Ba733
Ba734
Ba735
Ba736
Ba737
Ba738
Ba739
Ba740
Ba741
Ba742
Ba743
Ba744
Ba745
Ba746
Ba747
Ba748
Ba749
Ba750
Ba751
Ba752
Ba753
Ba754
Ba755
Ba756
Ba757
Ba758
Ba759
Ba760
Ba761
Ba762
Ba763
Ba764
Ba765
Ba766
Ba767
Ba768
Ba769
Ba770
Ba771
Ba772
Ba773
Ba774
Ba775
Ba776
Ba777
Ba778
Ba779
Ba780
Ba781
Ba782
Ba783
Ba784
Ba785
Ba786
Ba787
Ba788
Ba789
Ba790
Ba791
Ba792
Ba793
Ba794
Ba795
Ba796
Ba797
Ba798
Ba799
Ba800
Ba801
Ba802
Ba803
Ba804
Ba805
Ba806
Ba807
Ba808
Ba809
Ba810
Ba811
Ba812
Ba813
Ba814
Ba815
Ba816
Ba817
Ba818
Ba819
Ba820
Ba821
Ba822
Ba823
Ba824
Ba825
Ba826
Ba827
Ba828
Ba829
Ba830
Ba831
Ba832
Ba833
Ba834
Ba835
Ba836
Ba837
Ba838
Ba839
Ba840
Ba841
Ba842
Ba843
Ba844
Ba845
Ba846
Ba847
Ba848
Ba849
Ba850
Ba851
Ba852
Ba853
Ba854
Ba855
Ba856
Ba857
Ba858
Ba859
Ba860
Ba861
Ba862
Ba863
Ba864
Ba865
Ba866
Ba867
Ba868
Ba869
Ba870
Ba871
Ba872
Ba873
Ba874
Ba875
Ba876
Ba877
Ba878
Ba879
Ba880
Ba881
Ba882
Ba883
Ba884
Ba885
Ba886
Ba887
Ba888
Ba889
Ba890
Ba891
Ba892
Ba893
Ba894
Ba895
Ba896
Ba897
Ba898
Ba899
Ba900
Ba901
Ba902
Ba903
Ba904
Ba905
Ba906
Ba907
Ba908
Ba909
Ba910
Ba911
Ba912
Ba913
Ba914
Ba915
Ba916
Ba917
Ba918
Ba919
Ba920
Ba921
Ba922
Ba923
Ba924
Ba925
Ba926
Ba927
Ba928
Ba929
Ba930
Ba931
Ba932
Ba933
Ba934
Ba935
Ba936
Ba937
Ba938
Ba939
Ba940
Ba941
Ba942
Ba943
Ba944
Ba945
Ba946
Ba947
Ba948
Ba949
Ba950
Ba951
Ba952
Ba953
Ba954
Ba955
Ba956
Ba957
Ba958
Ba959
Ba960
Ba961
Ba962
Ba963
Ba964
Ba965
Ba966
Ba967
Ba968
Ba969
Ba970
Ba971
Ba972
Ba973
Ba974
Ba975
Ba976
Ba977
Ba978
Ba979
Ba980
Ba981
Ba982
Ba983
Ba984
Ba985
Ba986
Ba987
Ba988
Ba989
Ba990
Ba991
Ba992
Ba993
Ba994
Ba995
Ba996
Ba997
Ba998
Ba999
Ba1000


NORD-100 CPU
Sequencing (SEQ)
3002
5A07
NORSK DATA A.S
CORRECTED ACCORDING TO ECO:100-14
100-16, 100-19, 100-65, 100-73,
100-76, 87, 100-87
28.05 79
30 01 80
27.10 80
TD NO. 322602
CARD PRINT N E
Page 5 of 5
21A/8
23A/8
Am 2914
3 x MM 7685 - 2/5
MAP0
MAP1
MAP2
μMAP
μMAPJ
μMAPL
INTRQ
EPRIV
EMINT
SOWF
PANEL
EINT
BINT15
BINT14
BINT13
BINT12
BINT11
BINT10
POWFAIL
MOR
PAKERR
IOXERR
PAGEF
PROTV
RESTART
CONTINUE
SW SIN
SWCONT
SWLD
SWRS
STRT
STP
CLEAR
RUN
ACLK
ALUCLK
ALUOE
ALD0
ALD1
ALD2
ALD3
INH
CLA
DCOND
ECOND
LOOP
CLFF
LCZ
LDLC
LCRYL
ESPR
IOB
CLIRQ
INT
VIC
PIC
EPIC
EPICS
RA(0-3)
μACIN
μACRY
μBIT(0-11)
μMAPJ
μMAPL
μMAPJ
INTRQ
ND-06.016.01

G.2 4 TERMINALS AND FLOPPY DISK

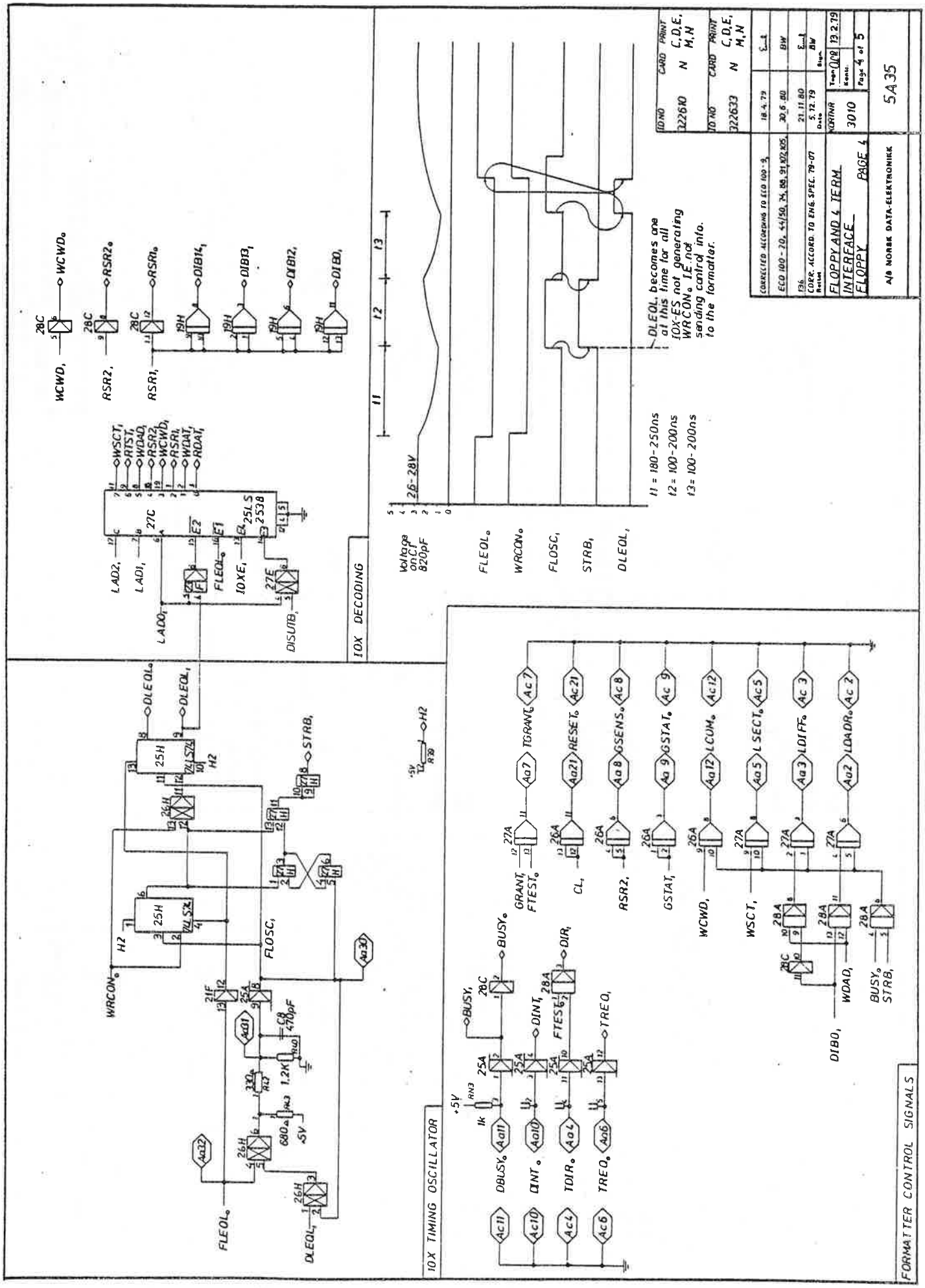


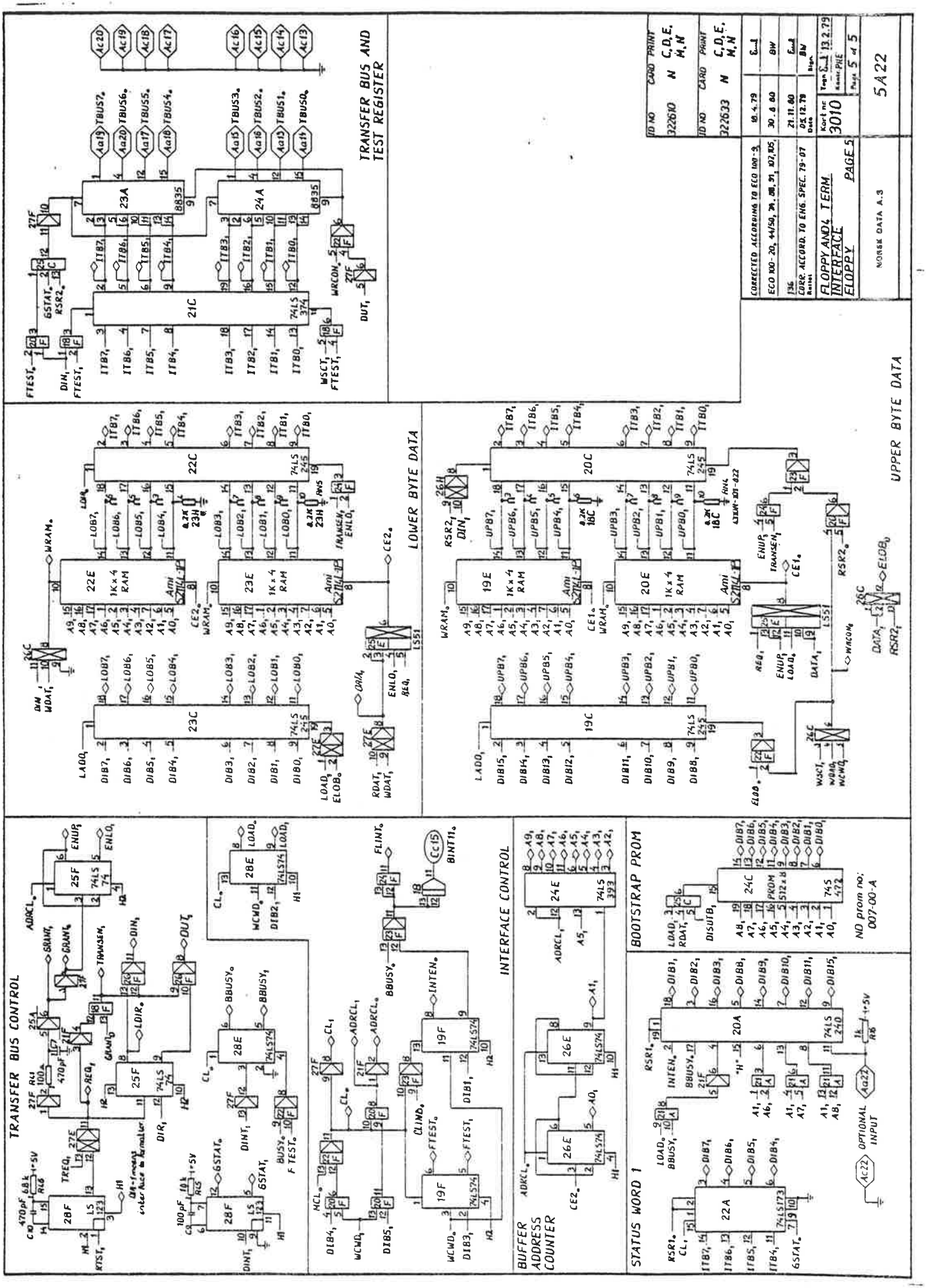

FLOPPY AND 4 TERM INTERF
PAGE 2. TERMINALS
3010
NORSK DATA A.S
Oslo, Norway
5A21
ADDRESSING, INTERRUPT AND CONTROL
TIMING
SCANNING
DATA
UART
IM-6402
EECO
217057 GF
thumb-wheel
for initial
baud-rate
X-TAL OSC
4.9152 MHz
K1115A
GROUP0
GROUP1
GROUP2
GROUP3

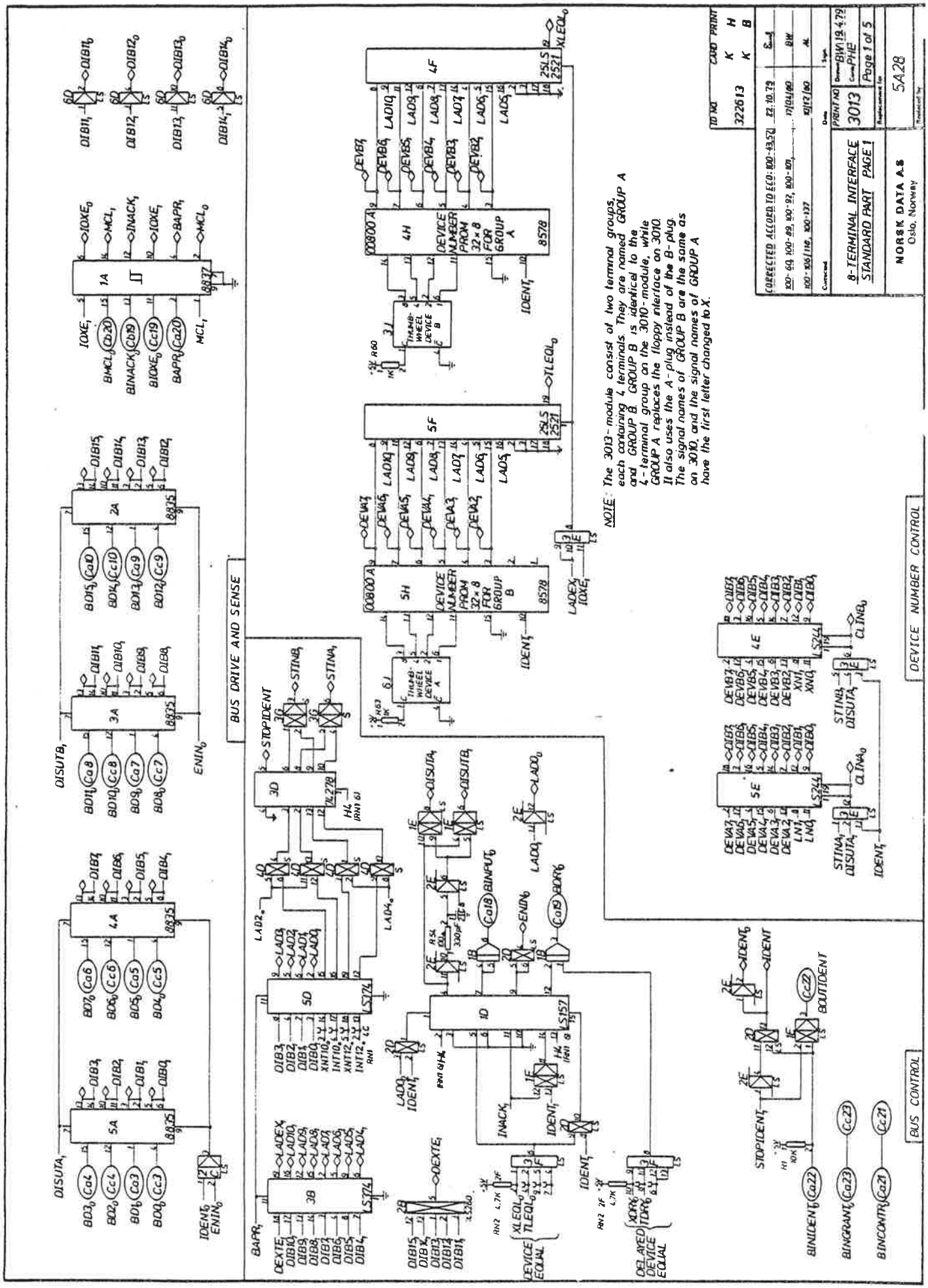


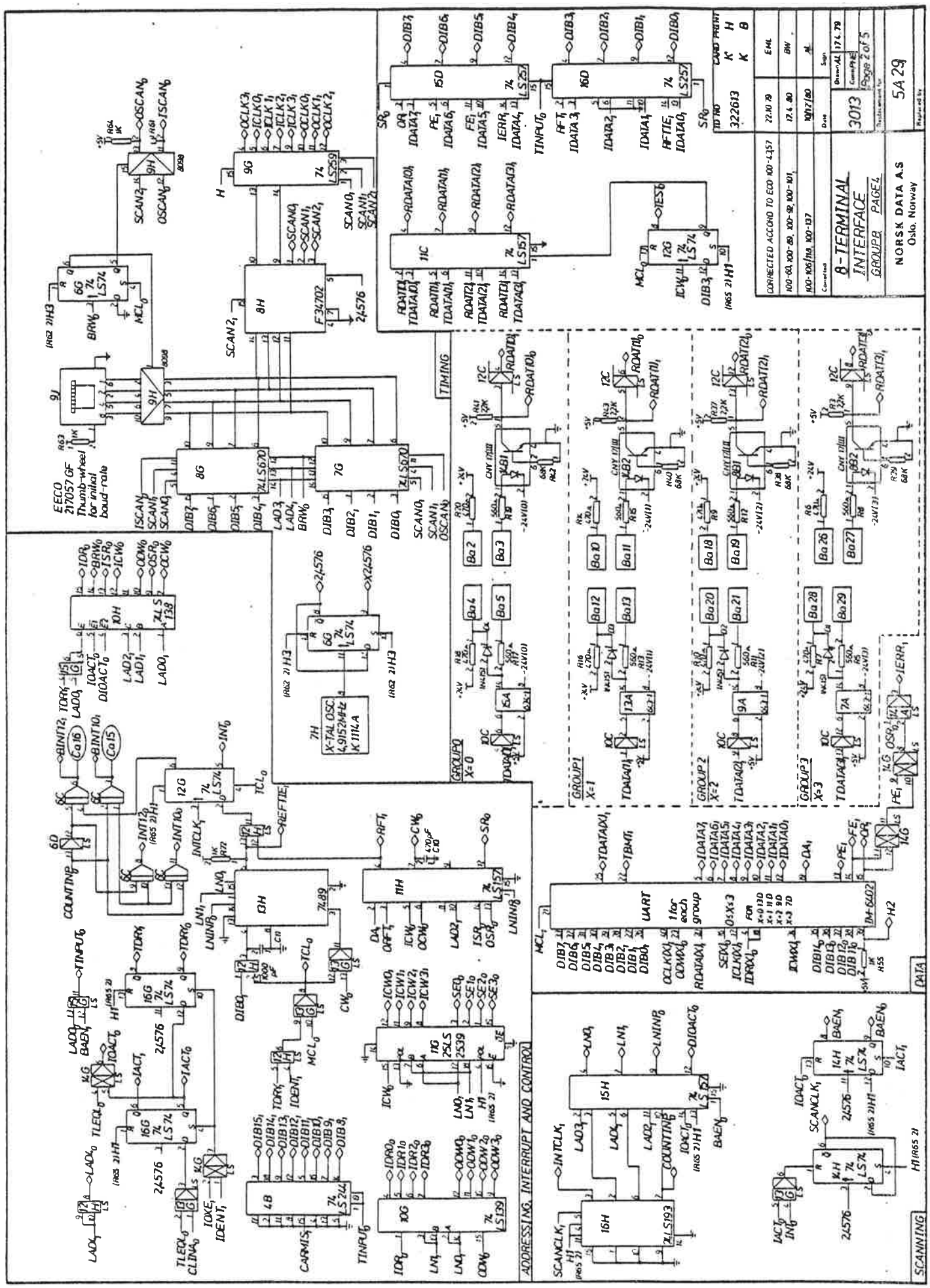
ND-06.016.01

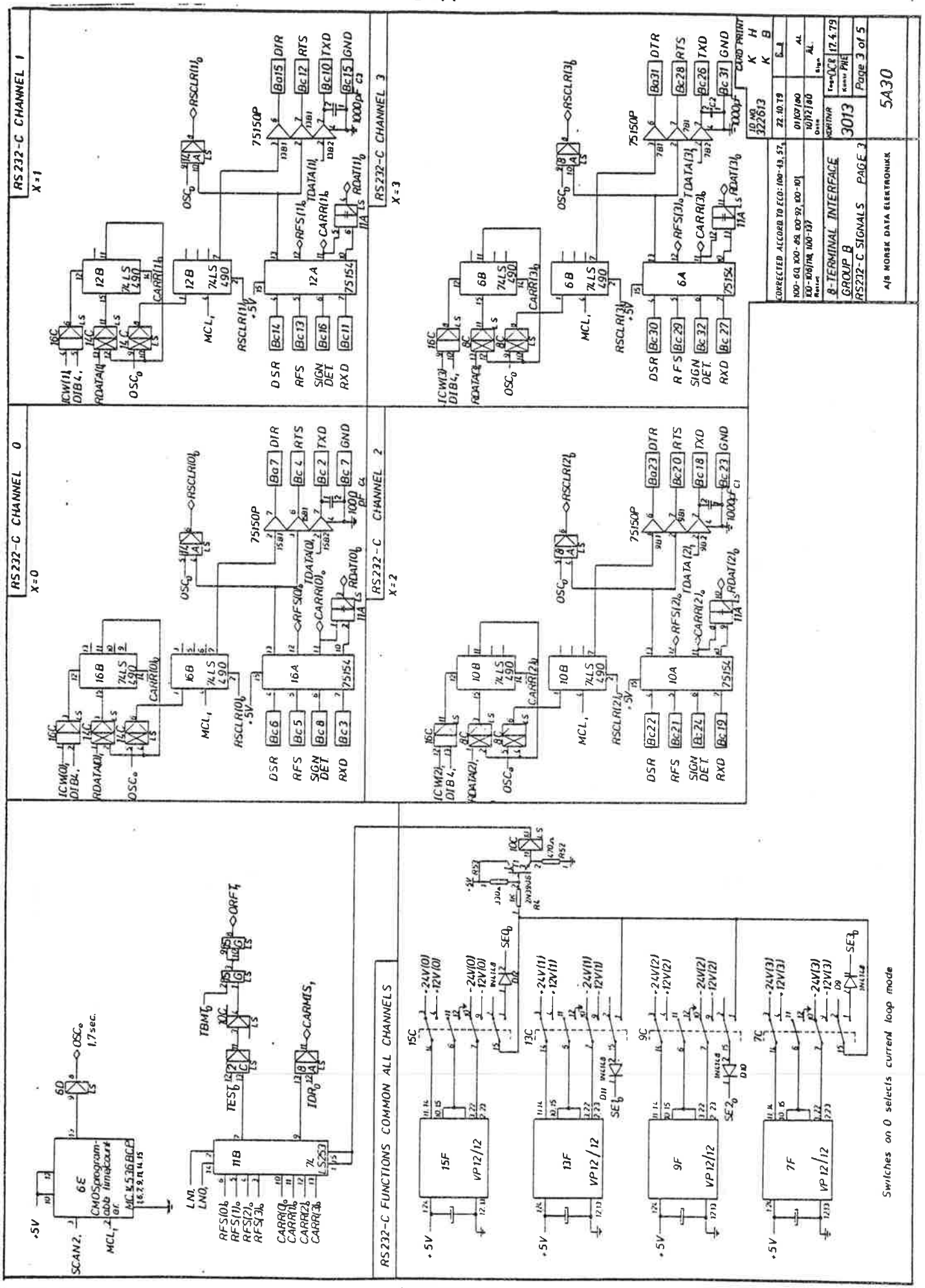
G-9

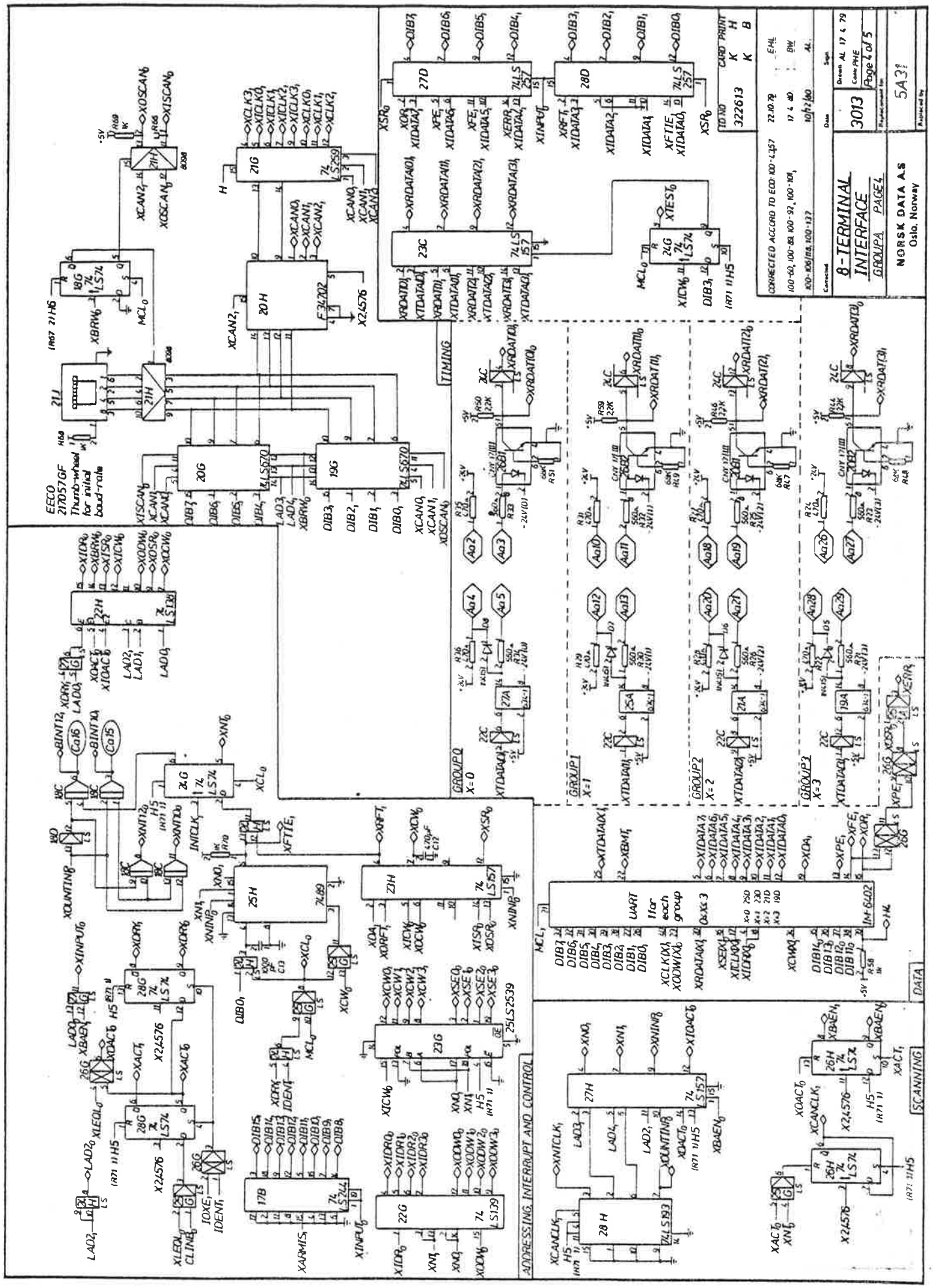

IOX DECODING
IOX TIMING OSCILLATOR
FORMATTER CONTROL SIGNALS
FLOPPY AND 4 TERM. INTERFACE FLOPPY PAGE 4
A/S NORSK DATA-ELEKTRONIKK
5435
DLEOL becomes one at this time for all IOX-ES not generating WRCON. IE not sending control info. to the formatter.
t1 = 180-250ns
t2 = 100-200ns
t3 = 100-200ns
WCWD
RSR2
RSR1
DIB14
DIB13
DIB12
DIB11
WSCT
RTST
WDAD
RSR2
WCWD
RSR1
WDAT
RDAT
LAD2
LAD1
LAD0
IOXE
FLEOL
DISUTB
FLEOL
WRCON
FLOSC
STRB
DLEOL
BUSY
GRANT
FTEST
RESET
GSENS
GSTAT
LCUM
LSECT
LDIFF
LOADR
DBUSY
DINT
DIR
TREQ
WDAD
BUSY
STRB
DIB0

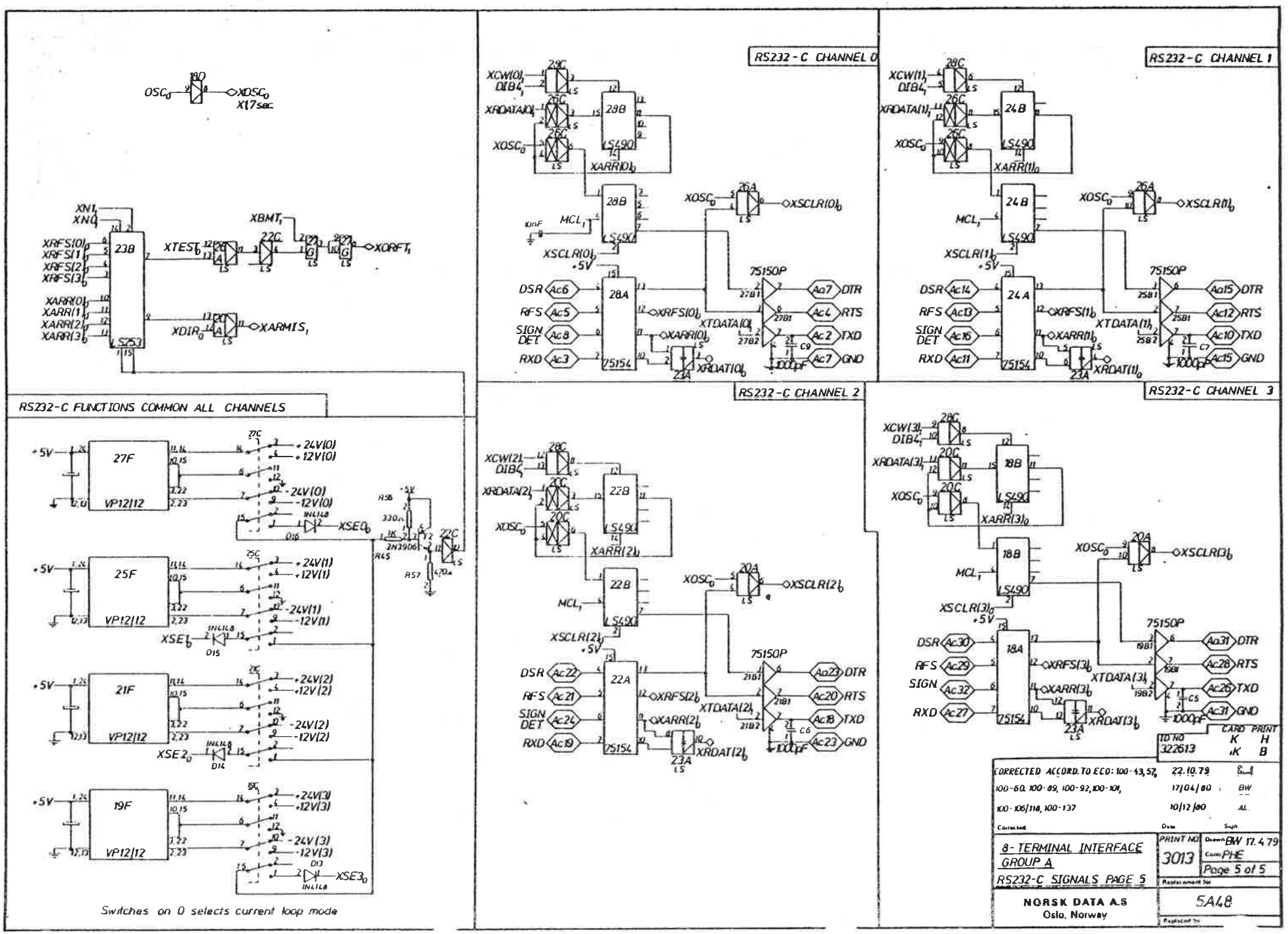

TRANSFER BUS CONTROL
INTERFACE CONTROL
BUFFER ADDRESS COUNTER
STATUS WORD 1
BOOTSTRAP PROM
TRANSFER BUS AND TEST REGISTER
LOWER BYTE DATA
UPPER BYTE DATA
FLOPPY AND TERM INTERFACE
ELOPPY
PAGE 5
3010
5A22
NORSK DATA A.S
CORRECTED ACCORDING TO ECO 100-3
ECO 100-20, 44/50, 74, 88, 91, 103, 105,
CORR. ACCORD. TO ENG. SPEC. 79-07
ID NO 322610 CARD PRINT N C,D,E, M,N
ID NO 322633 CARD PRINT N C,D,E, M,N
OPTIONAL INPUT
NO prom no; 007-00-A


8-TERMINAL INTERFACE
STANDARD PART PAGE 1
3013
Page 1 of 5
NORSK DATA A.S
Oslo, Norway
5A28
BUS DRIVE AND SENSE
DEVICE NUMBER CONTROL
BUS CONTROL
NOTE: The 3013-module consist of two terminal groups, each containing 4 terminals. They are named GROUP A and GROUP B. GROUP B is identical to the 4-terminal group on the 3010-module, while GROUP A replaces the floppy interface on 3010. It also uses the A-plug instead of the B-plug. The signal names of GROUP B are the same as on 3010, and the signal names of GROUP A have the first letter changed to X.

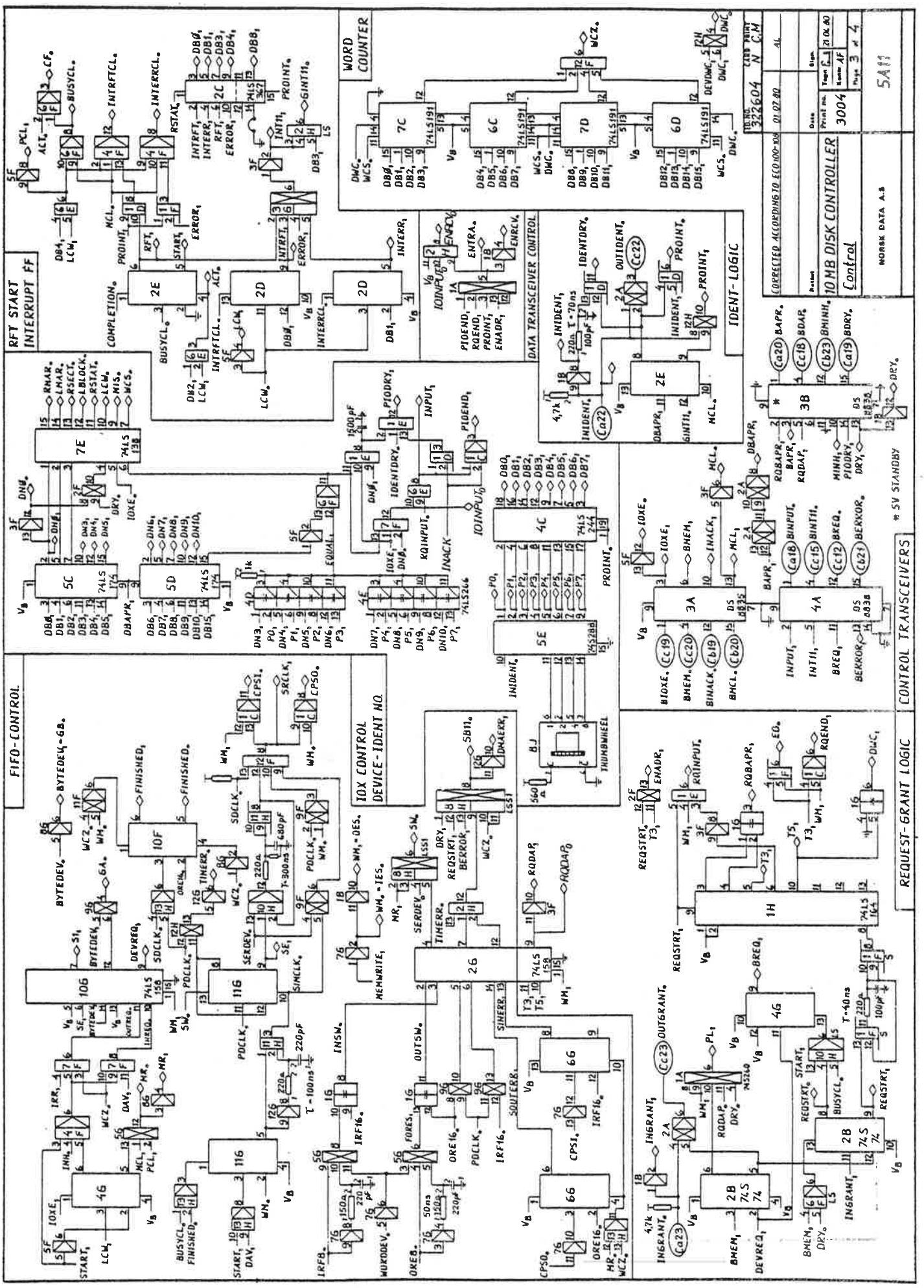

8-TERMINAL INTERFACE
GROUP8 PAGE4
NORSK DATA A.S
Oslo, Norway
3013
5A 29
322613
22.10.79
17.4.80
10/7/80
CONNECTED ACCORD TO ECO 100-4357
100-103, 100-88, 100-92, 100-101,
100-106/114, 100-137
ADDRESSING, INTERRUPT AND CONTROL
SCANNING
DATA
GROUP0 X=0
GROUP1 X=1
GROUP2 X=2
GROUP3 X=3
UART
TIMING
COUNTING
SCANNING

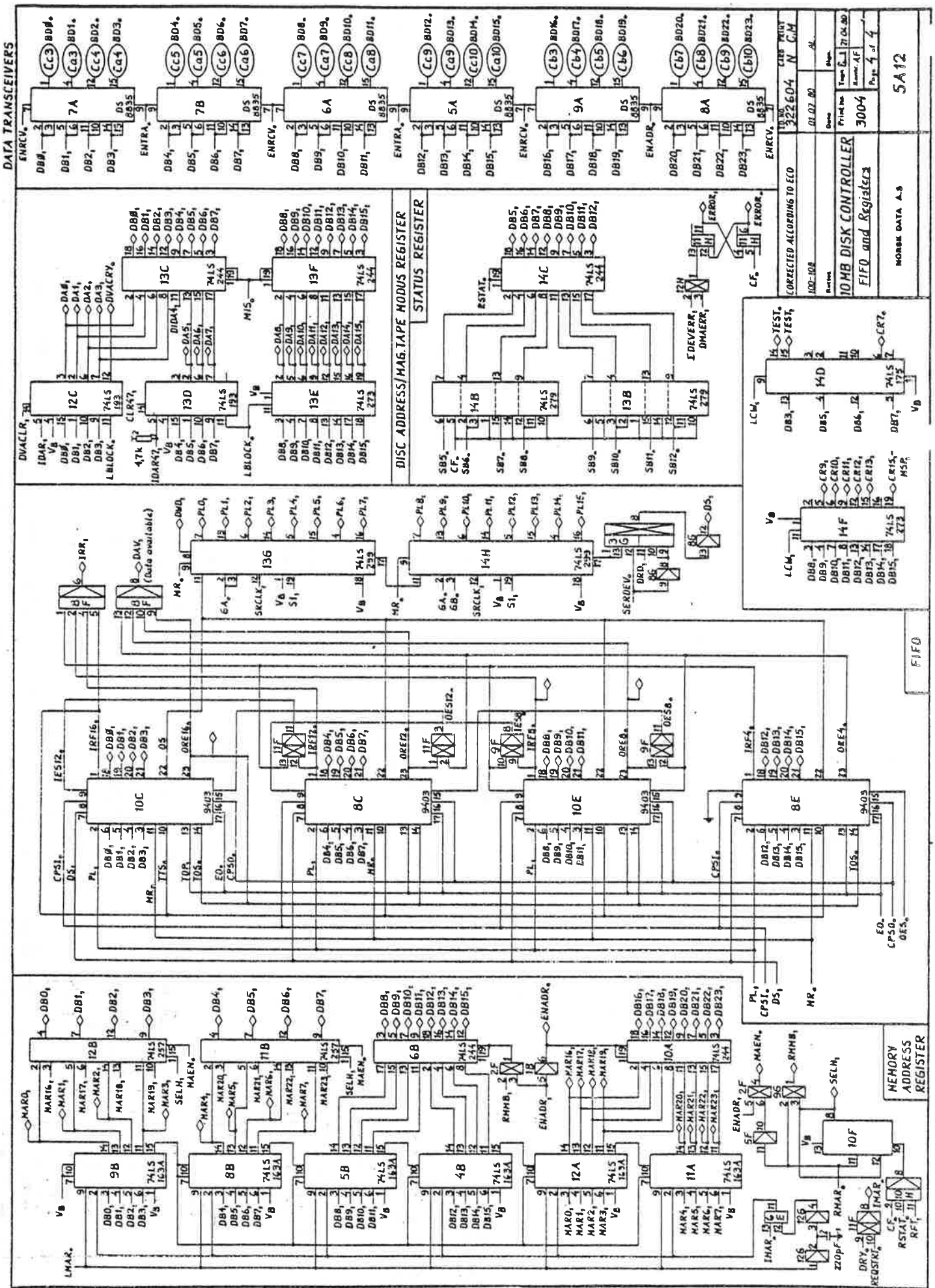

RS232-C CHANNEL 1
X=1
RS232-C CHANNEL 0
X=0
RS232-C CHANNEL 3
X=3
RS232-C CHANNEL 2
X=2
RS232-C FUNCTIONS COMMON ALL CHANNELS
8-TERMINAL INTERFACE
GROUP B
RS232-C SIGNALS PAGE 1
A/S NORSK DATA ELEKTRONIKK
5430
Page 3 of 5
3013
Switches on 0 selects current loop mode
DSR
RFS
SIGN DET.
RXD
DTR
RTS
TXD
GND


8-TERMINAL INTERFACE
GROUP A PAGE 4
NORSK DATA A.S
Oslo, Norway
3013
Page 4 of 5
5A31
TIMING
GROUP 0 X=0
GROUP 1 X=1
GROUP 2 X=2
GROUP 3 X=3
UART 11ar each group
ADDRESSING, INTERRUPT AND CONTROL
DATA
SCANNING

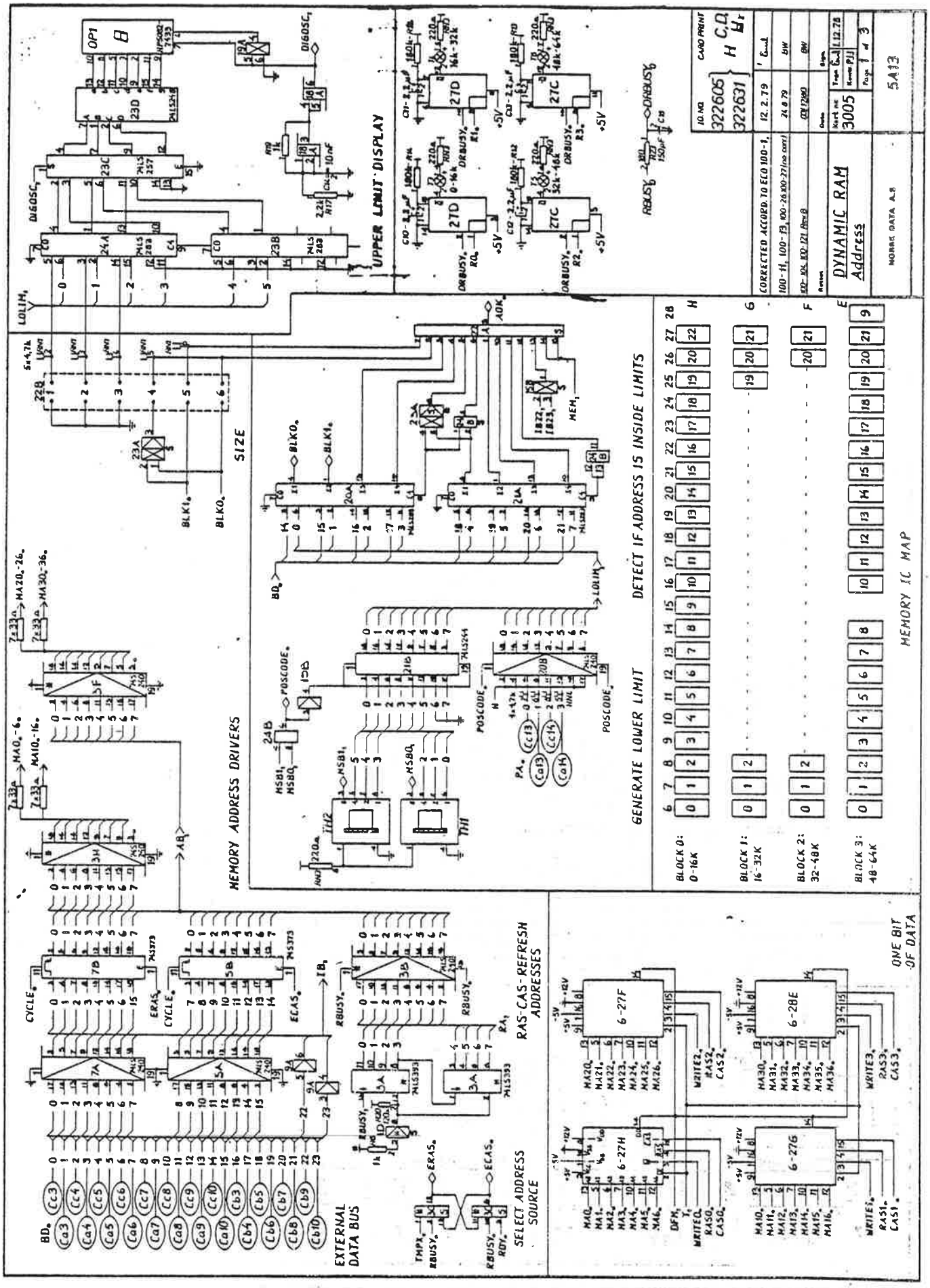

RS232-C CHANNEL 0
RS232-C CHANNEL 1
RS232-C CHANNEL 2
RS232-C CHANNEL 3
RS232-C FUNCTIONS COMMON ALL CHANNELS
Switches on 0 selects current loop mode
8-TERMINAL INTERFACE GROUP A
RS232-C SIGNALS PAGE 5
NORSK DATA A.S
Oslo, Norway
PRINT NO 3013
Page 5 of 5
5A48

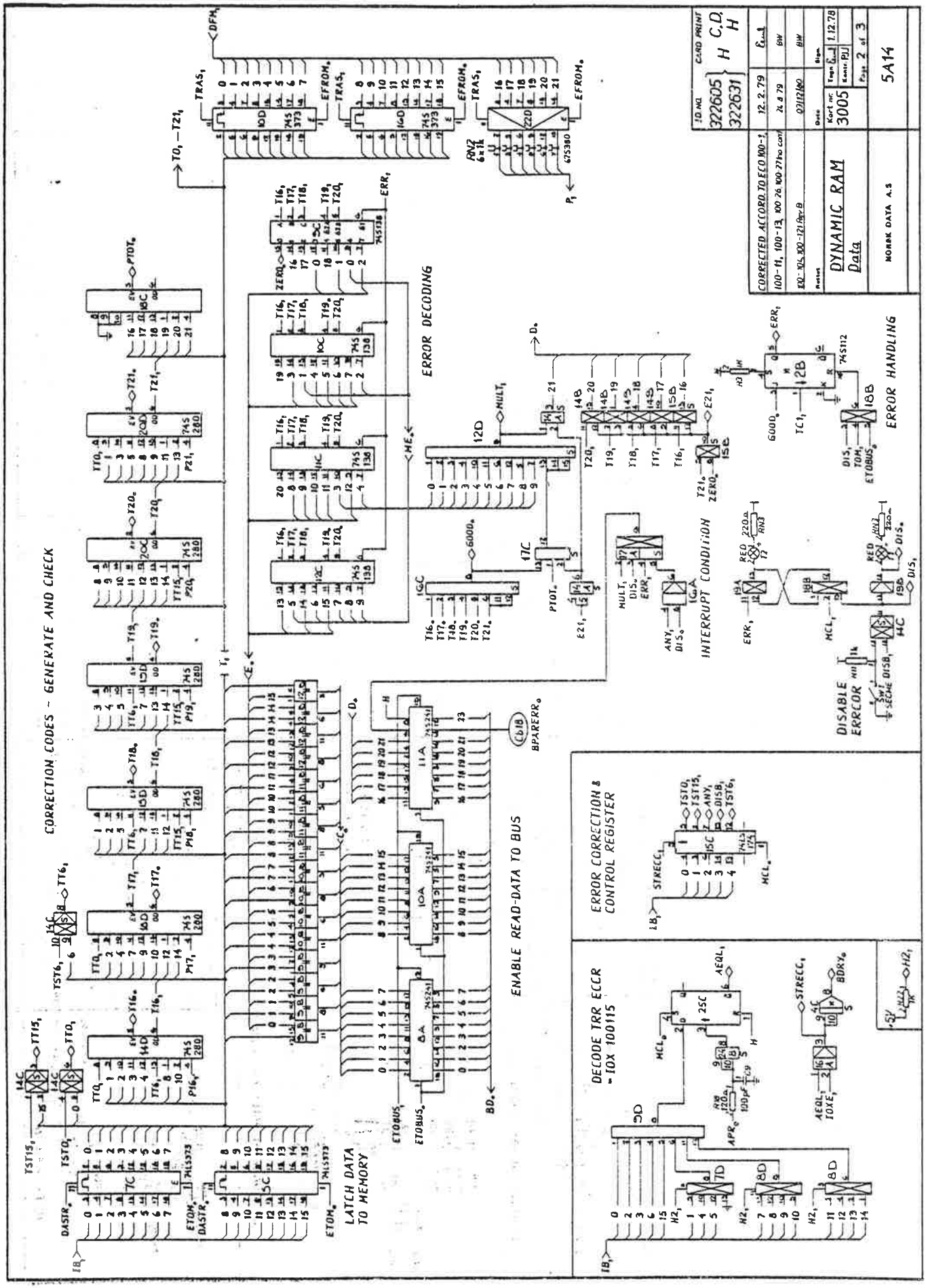
G.4 10mB DISK CONTROLLER

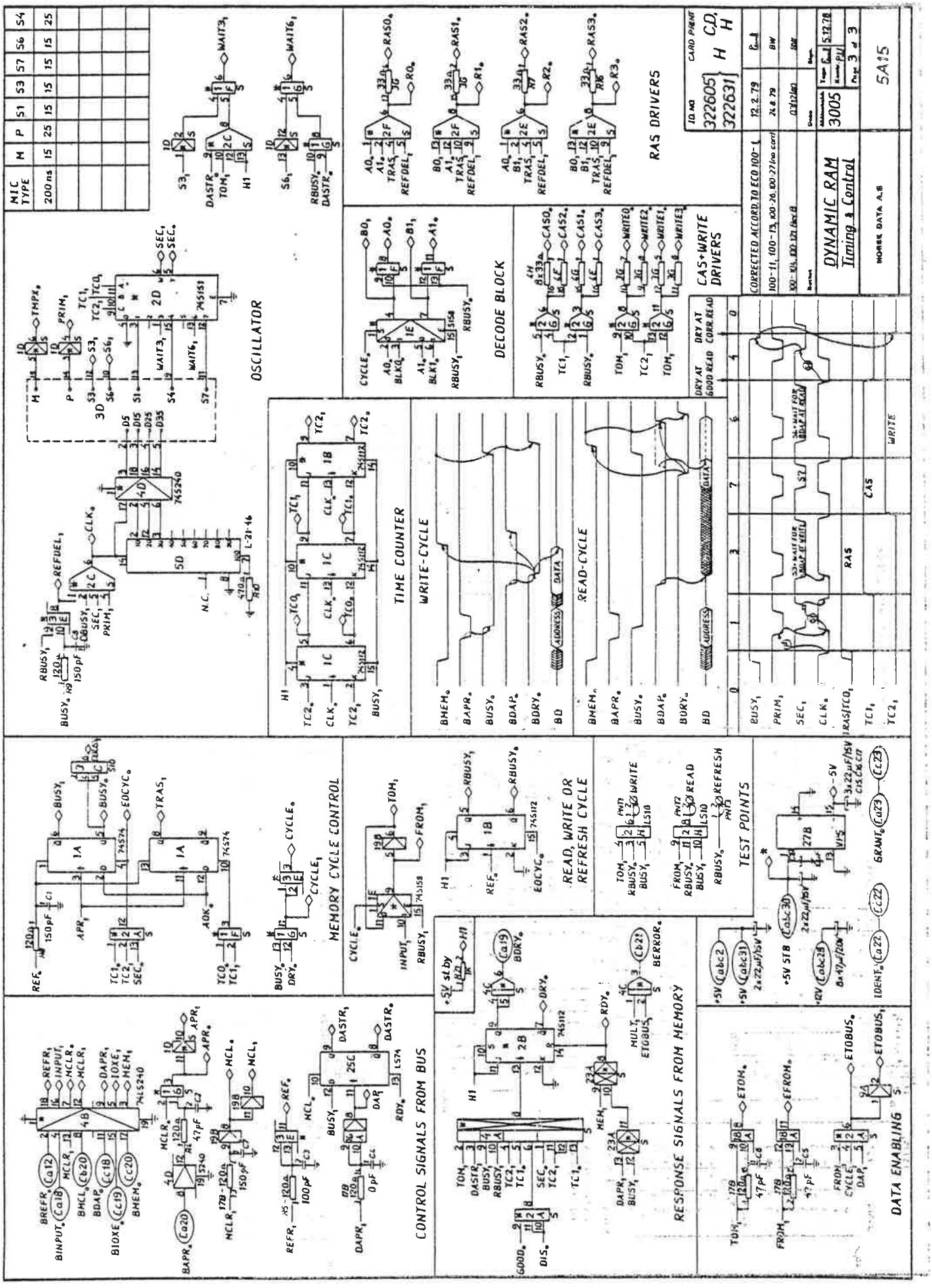



DATA TRANSCEIVERS
10MB DISK CONTROLLER
FIFO and Registers
3004
5A12
DISC ADDRESS/MAG.TAPE MODUS REGISTER
STATUS REGISTER
FIFO
MEMORY ADDRESS REGISTER
ND-06.016.01

G.5 DYNAMIC RAM




CORRECTION CODES - GENERATE AND CHECK
ERROR DECODING
ERROR HANDLING
INTERRUPT CONDITION
DISABLE ERRCOR
ENABLE READ-DATA TO BUS
LATCH DATA TO MEMORY
ERROR CORRECTION & CONTROL REGISTER
DECODE TRR ECCR - IOX 100115
DYNAMIC RAM Data
NORSK DATA A.S
5A14
CORRECTED ACCORD TO ECO 100-1, 100-11, 100-13, 100 26, 100 27 no com
CARD PRINT H C.D H


DYNAMIC RAM
Timing & Control
3005
MORK DATA A.S
5A15
OSCILLATOR
TIME COUNTER
WRITE-CYCLE
READ-CYCLE
DECODE BLOCK
RAS DRIVERS
CAS+WRITE DRIVERS
MEMORY CYCLE CONTROL
READ, WRITE OR REFRESH CYCLE
TEST POINTS
CONTROL SIGNALS FROM BUS
RESPONSE SIGNALS FROM MEMORY
DATA ENABLING