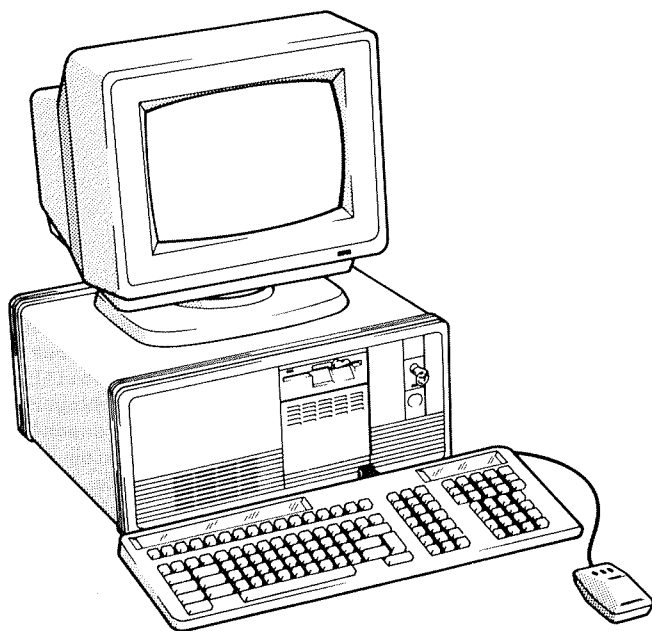


BUTTERFLY - 110



TECHNICAL REFERENCE MANUAL

ND PRINTING HISTORY		
Issue	Date	Notes
1	7/86	Preliminary issue
2	, 9/86	Pilot issue
3	4/87	Revised. Title now "Butterfly-110 "

Disclaimer

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

Copyright

Copyright ©1987 by Norsk Data A.S.

The information in this document is protected by copyright. It may not be photocopied, reproduced, or translated without the prior consent of Norsk Data A.S.

Acknowledgements

IBM is a registered trademark of International Business Machines Corporation USA.

MS-DOS and MS-Windows are trademarks of Microsoft Corporation USA.

EIS is a trademark of Ericsson Information Systems AB Sweden.

ORDER INFORMATION

This manual is in loose leaf form for ease of handling and of updating. It is designed to be placed in a standard 4-ring binder, which provides greater protection and convenience of use for a manual of this size. The capacity of the binder required is 40mm.

A special Butterfly-110 ring binder, with pre-printed titles, is available for this manual.

Additional copies of this manual and its Butterfly-110 ring binder may be ordered.

Please send your order to your local ND office,

or in Norway to Documentation Department
 Norsk Data A.S
 Olaf Helsets Vei 5
 Post Box 25
 Bogerud
 0621 Oslo 6

or in UK to Documentation Department
 Norsk Data Ltd
 Benham Valence
 Newbury
 Berkshire RG16 8LU

quoting your name, company and address, and specifying the ND part number for this manual.

UPDATES

Manuals can be updated in two ways:

- complete new issues
- revised pages.

New issues consist of a complete new manual which replaces the earlier issue of the manual. A new issue incorporates all the revisions since the previous issue.

Revisions consist of one or more pages which replace pages in the existing issue of the manual. Each revised page is listed on the new Printing Record which accompanies the revision pages. The previous Printing Record should be replaced by the new one.

New issues and revisions are announced in the ND Bulletin, and can be ordered through the normal ND channels.

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 it. All comments are welcome, both detailed and general.

PREFACE

THE PRODUCT

BUTTERFLY-110 is a complete system, comprising a version of the ND-110CX computer (called the ND-110PCX), fitted into the enclosure of an IBM-compatible PC-AT computer.

The Butterfly-110 product offers in one machine the functionality of an IBM PC-AT with all its currently available software running under MS-DOS, coupled with the extensive range of ND NOTIS software which runs under ND's SINTRAN III operating system. Information in Butterfly-110 is freely interchangeable between the PC and ND operating systems, i.e. data created under the PC system may then be used by the ND system, and vice versa.

A centronics parallel printer port is available as standard. Options include a 4-port serial communications controller, which may be used to connect ND-compatible terminals to the ND-110PCX part of the workstation, making a Butterfly Teamstation environment.

THE READER

- Technical managers and system personnel requiring a functional appreciation of the Butterfly-110 product.
- Hardware engineers requiring a detailed description of the ND-110PCX and its microprogram.
- Software engineers requiring detailed information about the programming changes and additions to the ND software (including the ND-PC communication system), and to the MS-DOS environment.
- Maintenance personnel requiring information about preventive and corrective maintenance for Butterfly-110, including diagnostic test programs and parts data.

THE MANUAL

This manual provides technical information for the Butterfly-110, cross-referencing other manuals where appropriate. It includes a detailed description of the ND-110PCX part of the system, its interface with the PC-AT, plus maintenance and parts information.

RELATED MANUALS The other manuals in the suite of Butterfly-110 product manuals are:

Butterfly Teamstation Installation Guide	ND-30.056
Butterfly Teamstation Supervisor Guide	ND-30.057
Butterfly Teamstation User Guide	ND-60.242

Detailed information on ND-110, which is also applicable to the ND-110PCX, can be found in the following manuals:

ND-110 Functional Description	ND-06.026
ND-110 Instruction Set	ND-06.029
Rask Gate Arrays Technical Description	ND-05.016
ND-100 Test Program Description	ND-30.005

Information on the PC and its MS-DOS operating system is given in:

PC Technical Reference Manual	ND-06.028
MS-DOS Reference Manual	ND-60.271

The ND operating system SINTRAN, and all ND applications, are covered in associated ND user guides. In particular, reference is made to the following:

SINTRAN III System Supervisor Guide	ND-30.003
SINTRAN III System Documentation, Appendix A - Data Fields	ND-60.112

TABLE OF CHAPTERS

CHAPTER 1 OVERVIEW AND DATA SUMMARY

CHAPTER 2 FUNCTIONAL DESCRIPTION - SYSTEM AND HARDWARE

CHAPTER 3 FUNCTIONAL DESCRIPTION - SOFTWARE

CHAPTER 4 HARDWARE DESCRIPTION

CHAPTER 5 MICROPROGRAM DESCRIPTION

CHAPTER 6 SOFTWARE DESCRIPTION

CHAPTER 7 MAINTENANCE

CHAPTER 8 PARTS DATA

APPENDIX A LOGIC DIAGRAMS

APPENDIX B SIGNALS LIST

APPENDIX C PC EXPANSION BUS

APPENDIX D PAL SPECIFICATIONS

APPENDIX E BUTTERFLY-110 LINK SETTINGS

APPENDIX F BUTTERFLY KEYBOARD SPECIFICATION

GLOSSARY

INDEX

DETAILED TABLE OF CONTENTS

Chapter		Page
1	OVERVIEW AND DATA SUMMARY	
1.1	INTRODUCTION	1-5
1.2	SYSTEM CONFIGURATION	1-7
1.2.1	Basic hardware	1-7
1.2.2	Cluster configuration plus options	1-8
1.2.3	Communications Controller	1-9
1.2.4	Summary of components	1-10
1.3	BUTTERFLY-110 SYSTEM UNIT	1-12
1.3.1	PC features	1-12
1.3.2	ND-110PCX	1-12
1.4	BUTTERFLY DISPLAY UNIT	1-13
1.5	BUTTERFLY KEYBOARD /MOUSE /BAR CODE READER	1-13
1.5.1	Keyboard	1-13
1.5.2	Mouse	1-14
1.6	CENTRONICS PRINTER PARALLEL INTERFACE	1-14
1.7	SERIAL INTERFACE	1-14
1.8	DATA SUMMARY	1-15
1.8.1	Mechanical	1-15
1.8.2	Electrical	1-15
1.8.3	Environmental	1-16
2	FUNCTIONAL DESCRIPTION - SYSTEM AND HARDWARE	
2.1	OVERVIEW	2-5
2.1.1	Butterfly Main Units	2-5
2.1.1.1	System Unit	2-6
2.1.1.2	Keyboard	2-7
2.1.1.3	Display	2-7
2.1.1.4	Mouse	2-7
2.1.1.5	Hardware options	2-8

2.1.2	System Control	2-8
2.1.2.1	Functional elements	2-8
2.1.2.2	System start-up	2-9
2.1.2.3	I/O control	2-10
2.1.2.4	ND-PC communication	2-10
2.2	ND-110PCX FUNCTIONAL BLOCKS	2-12
2.2.1	Microprogram and processor (CPU)	2-12
2.2.2	Register File	2-13
2.2.3	Interrupt System	2-13
2.2.4	Memory Management System (MMS)	2-14
2.2.5	Test (console) port	2-15
2.2.6	Print Status	2-15
2.2.7	Installation Number	2-15
2.2.8	Memory (RAM)	2-16
2.2.9	I/O Control	2-18
2.2.10	Oscillator	2-18
2.2.11	Power Fail	2-18
2.3	THE KEYBOARD	2-19
2.3.1	Summary of Features	2-19
2.3.2	IBM Compatibility	2-23
2.3.3	ND-NOTIS Compatibility	2-23
2.3.4	Added Features	2-24
2.3.4.1	Mouse	2-24
2.3.4.2	Bar Code Reader port	2-25
2.3.4.3	Function Keypad option	2-25
2.3.4.4	Template cards	2-26

3 FUNCTIONAL DESCRIPTION - SOFTWARE

3.1	INTRODUCTION	3-5
3.2	INSTRUCTION SET	3-7
3.3	SOFTWARE OVERVIEW	3-8
3.3.1	General	3-8
3.3.2	Signal box	3-8
3.3.3	Interrupts	3-10
3.3.4	Mailbox	3-11
3.3.5	IOX simulators	3-13
3.3.6	PC-NDI/O	3-14
3.3.7	System manager	3-14
3.3.8	Keyboard handler	3-15
3.3.9	BIOS sharing module	3-15
3.3.10	File access	3-16
3.3.11	Print spooler	3-19
3.3.12	PC server	3-19
3.3.13	Software memory map	3-20

3.4	DATA HANDLING	3-21
3.5	MAILBOXES	3-22
3.5.1	Mailbox format	3-22
3.5.2	Disk mailbox data buffer	3-22
3.5.3	Async mailbox data buffer	3-22
3.5.4	Butterfly DMA data buffer	3-23
3.6	SENDING A MAILBOX	3-24
3.6.1	Mailbox queues	3-24
3.6.1	Send mailbox to PC (NDSOURCE)	3-24
3.6.2	Send mailbox to ND (PCSOURCE)	3-25
3.7	ND SIDE OF THE MAILBOX	3-26
3.7.1	Butterfly interrupt handling	3-26
3.7.2	IDENT handling	3-30
3.7.3	IOX(T) handling	3-31
3.8	PC SIDE OF THE MAILBOX	3-32
3.8.1	Interrupt handling	3-32
3.8.2	Disk handling	3-34
3.8.3	Asynchronous (serial port) handling	3-34
3.8.4	DMA device handling	3-35
3.9	FILE ACCESS	3-36
3.9.1	PC trap (filter)	3-36
3.9.2	Inter-CPU buffer messaging	3-36
3.9.3	Fileserver (PCFSERV)	3-37
3.9.4	Butterfly device drivers / buffers	3-37
3.10	SYSTEM MANAGER AND KEYBOARD HANDLER	3-38

4 HARDWARE DESCRIPTION

4.1	HARDWARE MODULES	4-5
4.2	THE PC	4-7
4.2.1	References	4-7
4.2.2	PC interrupt system	4-7
4.2.3	PC memory assignments	4-7
4.3	THE ND-110PCX COMPUTER	4-8
4.3.1	Block diagram	4-8
4.3.2	Basic clocks	4-10
4.3.3	Internal bus system	4-10

4.3.4	Processor	4-12
4.3.4.1	General	4-12
4.3.4.2	Control Store and pipeline	4-13
4.3.4.3	The MIC	4-14
4.3.4.4	Cycle timing and control	4-19
4.3.4.5	Register File	4-26
4.3.4.6	The ALU	4-26
4.3.4.7	The MAC	4-32
4.3.4.8	Control signal decodes/functions	4-34
4.3.4.9	CPU I/O registers	4-35
4.3.4.10	Traps	4-36
4.3.4.11	Print status/installation number	4-37
4.3.5	Memory	4-38
4.3.5.1	Memory management principles	4-38
4.3.5.2	Memory map	4-38
4.3.5.3	Memory addressing	4-42
4.3.5.4	Read/write/refresh cycles	4-48
4.3.5.5	Parity and error checking	4-54
4.3.6	Interrupt system	4-56
4.3.6.1	Interrupt mechanism	4-56
4.3.6.2	Interrupt controller	4-58
4.3.6.3	Signal box interrupts	4-59
4.3.6.4	Internal interrupts	4-60
4.3.7	PC-NDI/O system	4-64
4.3.7.1	PC expansion bus	4-64
4.3.7.2	PC access to ND memory	4-64
4.3.7.3	PC program I/O	4-64
4.3.7.4	PC-ND status and control registers	4-66
4.3.8	ND-I/O system	4-67
4.3.8.1	HDLC & DMA	4-68
4.3.8.2	Console port	4-70
4.3.8.3	Panel processor	4-70
4.3.9	Real-time clock	4-71
4.3.10	Power fail and stop	4-71
4.3.11	Clear	4-71
4.4	HARDWARE OPTIONS	4-72
4.4.1	Serial Communications Board plus Connector Box	4-72
4.4.2	Tape streamer	4-72
4.4.3	Additional 1Mbyte RAM	4-72
4.4.4	HDLC communications	4-73
4.4.5	Plug-in modem	4-73
4.4.6	External Winchester disk	4-73
4.5	ND-110PCX TERMINAL DISPLAY UNITS	4-74
4.6	ND-110PCX TERMINAL KEYBOARD	4-74
4.7	MOUSE	4-74
4.8	BAR CODE READER	4-74

5 MICROPROGRAM DESCRIPTION

5.1	INTRODUCTION	5-5
5.2	HARDWARE ON THE MICROCODE BUS	5-5
5.3	ND-110PCX MICROPROGRAM WORD FORMAT	5-9
5.3.1	Summary of changes from ND-100	5-9
5.3.2	Microinstruction word: field definitions	5-10
5.4	RETURN FROM SIMULATION INSTRUCTION: RTNSIM	5-21
5.4.1	Interface to the simulation routines	5-21
5.4.2	Entry point interface (UPSIM routine)	5-21
5.4.3	Return from simulation (RTNSIM instruction)	5-22
5.5	WRITING MICROPROGRAM	5-23
5.5.1	Summary	5-23
5.5.2	Design of microcode	5-24
5.5.3	Assembly language considerations	5-24
5.5.4	Generating control store PROM	5-26
5.6	MNEMONICS LIST	5-27

6 SOFTWARE DESCRIPTION

6.1	INTRODUCTION	6-7
6.2	MAILBOX	6-10
6.2.1	Mailbox devices	6-10
6.2.2	Standard mailbox locations	6-12
6.2.3	Disk mailboxes	6-12
6.2.3.1	Floppy disk mailbox format	6-13
6.2.3.2	Hard disk mailbox format	6-14
6.2.4	Asynchronous (serial port) mailbox format	6-15
6.2.5	Butterfly DMA devices mailbox format	6-17
6.3	IOX SIMULATION	6-19
6.3.1	Operation	6-19
6.3.2	IOX simulator modules	6-21
6.3.2.1	Tables and queues	6-21
6.3.2.2	Signal box	6-23
6.3.2.3	IOX simulator switch (IOXSWITCH)	6-26
6.3.2.4	IDENT handler (IDIOX)	6-27
6.3.2.5	PC interrupt (PCINT)	6-27
6.3.2.6	Timer (TMINT)	6-28
6.3.3	IOX instruction simulators	6-29
6.3.3.1	Terminal IOXs	6-29
6.3.3.2	Hard disk IOXs	6-31
6.3.3.3	Floppy disk IOXs	6-32
6.3.3.4	Standard disk status and control words	6-32

6.3.4	IOX 30x: the PC console as Terminal-1	6-35
6.3.5	IOX 10-13: Real time clock	6-36
6.3.6	Login display following logout	6-36
6.3.7	Other Butterfly-110 IOXs	6-37
6.3.8	Effects of no cache	6-37
6.4	BUTTERFLY-110 ASYNC DRIVER	6-38
6.4.1	Operation	6-38
6.4.2	MON 373: special INSTR	6-40
6.4.3	Terminal datafield extension	6-40
6.5	PC-NDI/O	6-41
6.5.1	Basic structure	6-41
6.5.2	Interrupts - PC signal box	6-44
6.5.3	Calendar handling	6-45
6.5.3.1	Get and Set commands	6-45
6.5.3.2	Operation through signal box	6-46
6.5.3.3	Get time	6-46
6.5.3.4	Set time	6-47
6.5.4	Terminal-1 operation	6-48
6.5.4.1	Output	6-48
6.5.4.2	Input	6-49
6.5.5	PC Peripheral device handling	6-49
6.5.5.1	Request queuing and device reservation	6-49
6.5.5.2	Device switch	6-50
6.5.5.3	Floppy disk handling	6-51
6.5.5.4	Hard disk handling	6-53
6.5.5.5	PC console display handling	6-56
6.5.5.6	PC console keyboard handling	6-56
6.5.5.7	Serial port handling	6-56
6.5.5.8	Parallel port handling	6-59
6.5.6	System initialisation	6-59
6.5.7	ND-kernel	6-59
6.5.8	System communications area	6-60
6.5.9	PC interrupts used	6-61
6.6	MULTITASKING PC SOFTWARE	6-62
6.6.1	MS-DOS and BIOS	6-62
6.6.2	Additional requirements for Butterfly-110	6-63
6.6.3	Prerequisite MS-DOS information	6-64
6.6.3.1	MS-DOS structure	6-64
6.6.3.2	PC interrupt system	6-65
6.6.4	Multitasking	6-66
6.6.4.1	The sharing requirement	6-66
6.6.4.2	PC-NDI/O multitasking control	6-66
6.6.4.4	BSM to PC-NDI/O multitasking control	6-68
6.6.4.5	Terminal manager multitasking	6-69

6.7	SYSTEM MANAGER	6-70
6.7.1	Overview	6-70
6.7.2	System modules	6-70
6.7.3	Keyboard compatibility	6-72
6.7.3.1	PC	6-72
6.7.3.2	ND	6-73
6.7.4	New keyboard features	6-74
6.7.5	System manager	6-76
6.7.6	Keyboard driver	6-77
6.7.7	Character format in input buffer	6-80
6.8	ND SERVERS	6-81
6.8.1	ND file services available to the PC	6-81
6.8.2	Communication between ND and PC	6-82
6.8.2.1	Extensions to MON 144 (MAGTP)	6-82
6.8.2.2	Status registers	6-85
6.8.3	File access requests from PC to ND-servers	6-86
6.9	PC SERVICE PROGRAMS	6-90
6.9.1	File access	6-90
6.9.1.1	Facilities	6-90
6.9.1.2	PC trap handler	6-92
6.9.1.3	Inter-CPU buffer messaging	6-94
6.9.3	File transfer program (DUPLI)	6-96
6.9.4	Print spooler service program (PSSP)	6-99
6.9.5	PC servers	6-99
6.10	SECRET MEMORY LOCATIONS	6-100
6.10.1	PROM addresses	6-100
6.10.2	NV-RAM addresses	6-100
6.10.3	DRAM addresses	6-100

7 MAINTENANCE

7.1	SYSTEM UNIT ASSEMBLY	7-6
7.2	PREVENTIVE MAINTENANCE	7-9
7.2.1	Fan	7-9
7.2.2	Calendar battery	7-9
7.2.3	Floppy disk drive	7-9
7.2.4	Tape streamer	7-10
7.2.5	Printer	7-11
7.2.6	Care of magnetic media	7-12
7.2.7	Routine cable inspection	7-12
7.3	SYSTEM COMPONENTS	7-13
7.3.1	Butterfly terminals	7-13
7.3.2	IBM-compatible PC-AT System Unit	7-13
7.3.3	EGA Board	7-13
7.3.4	Display Unit	7-13
7.3.5	Winchester Disk Drive	7-15

7.3.6	Floppy Disk Drive	7-15
7.3.7	Disk Controller Board	7-16
7.3.8	Keyboard	7-16
7.3.9	Mouse	7-16
7.3.10	ND-110PCX	7-16
7.3.10.1	Variants	7-16
7.3.10.2	Installation PROM	7-17
7.3.10.3	Memory upgrade	7-19
7.3.10.4	HDLC upgrade	7-19
7.3.11	Installing software on Winchester Disk	7-20
7.3.12	Extra RAM for MS-DOS	7-22
7.4	INSTALLATION TOOLS	7-23
7.4.1	PC Setup	7-23
7.4.2	Butterfly Supervisor: System Parameter Definition	7-31
7.4.3	Butterfly Supervisor: Serial Port 1 & Modem Setup	7-35
7.5	MOVING THE EQUIPMENT	7-39
7.6	CORRECTIVE MAINTENANCE	7-41
7.6.1	Basic system checks	7-41
7.6.2	Power-on self-test	7-44
7.6.2.1	Start-up sequence	7-44
7.6.2.2	Self-test operations	7-45
7.6.2.3	Self-test status flag (STS-FLAG)	7-46
7.6.2.4	Self-test error log (ERRLOG)	7-47
7.6.2.5	Self-test mailbox (st-mail)	7-50
7.6.3	Diagnostic programs	7-51
7.6.3.1	Butterfly diagnostic suite	7-51
7.6.3.2	Which diagnostic?	7-53
7.6.3.3	PC diagnostics	7-55
7.6.3.4	PC-ND interface diagnostics	7-62
7.6.3.5	ND diagnostics	7-68
7.6.4	Butterfly Supervisor maintenance facilities	7-69
7.6.4.1	Console facility	7-71
7.6.4.2	Warm Start and Cold Start	7-76
7.6.4.3	Telefix	7-80

8 PARTS DATA

8.1	SYSTEM UNIT	8-3
8.1.1	The PC	8-3
8.1.2	ND-110PCX CPU board 3401	8-4
8.1.3	ND-110PCX Local Bus board 3402	8-8
8.1.4	ND-110PCX Installation PROM board 3404	8.13
8.2	DISPLAY UNIT	8-13
8.3	KEYBOARD	8-13
8.4	MOUSE	8-13

APPENDIX A : LOGIC DIAGRAMS

Board 3401 : ND-110PCX CPU (4 sheets)

Board 3402 : ND-110PCX Local Bus (4 sheets)

APPENDIX B : SIGNALS LIST

APPENDIX C : PC EXPANSION BUS

APPENDIX D: PAL SPECIFICATIONS

PILOT BUTTERFLY PAL 1 : COMMAND DECODE 1	D-3
PILOT BUTTERFLY PAL 2 : COMMAND DECODE 2	D-4
PILOT BUTTERFLY PAL 3 : COMMAND DECODE 3	D-5
PILOT BUTTERFLY PAL 4 : COMMAND DECODE 4	D-6
PILOT BUTTERFLY PAL 5 : COMMAND DECODE 5	D-7
PILOT BUTTERFLY PAL 6 : IDB SOURCE DECODE 1	D-8
PILOT BUTTERFLY PAL 7 : IDB SOURCE DECODE 2	D-9
PILOT BUTTERFLY PAL 8 : BREAK DETECT	D-10
PILOT BUTTERFLY PAL 9 : CYCLE TYPE DECODE	D-11
PILOT BUTTERFLY PAL 10: CYCLE LENGTH DECODE	D-12
PILOT BUTTERFLY PAL 11: CONDITION CONTROL	D-13
PILOT BUTTERFLY PAL 12: CLOCK GENERATION	D-14
PILOT BUTTERFLY PAL 13: MMU CONTROL	D-15
PILOT BUTTERFLY PAL 14: TRAP 1	D-16
PILOT BUTTERFLY PAL 15: TRAP 2	D-17
PILOT BUTTERFLY PAL 16: TRAP 3	D-18
PILOT BUTTERFLY PAL 17: PC I/F CONTROL	D-19
PILOT BUTTERFLY PAL 18: PC I/O ADDRESS DECODE	D-20

PILOT BUTTERFLY PAL 19: LOCAL BUS ADDRESS DECODER	D-21
PILOT BUTTERFLY PAL 20: LOCAL BUS ADDRESS ARBITER	D-23
PILOT BUTTERFLY PAL 21: LOCAL BUS CONTROL	D-26
PILOT BUTTERFLY PAL 22: I/O BUS CONTROL	D-27
PILOT BUTTERFLY PAL 23: MEMORY CONTROL	D-28

APPENDIX E: BUTTERFLY-110 LINK SETTINGS

E.1 ND-110PCX LOCAL BUS BOARD (3402)	E-3
E.2 PC SYSTEM BOARD	E-4

APPENDIX F: KEYBOARD SPECIFICATION

1	INTRODUCTION	F-5
1.1	IBM compatibility	F-5
1.2	NOTIS compatibility	F-6
1.3	Standard keyboard layout and legends	F-6
2	NEW FEATURES	F-15
2.1	Mouse	F-15
2.1.1	Mouse position	F-15
2.1.2	Position coding	F-16
2.1.3	Push button coding	F-17
2.2	Serial port / Bar Code Reader	F-18
2.2.1	Serial input	F-18
2.2.2	Serial output	F-18
2.2.3	Data format	F-18
2.3	Separate function keypad	F-19
2.4	Template card guides	F-20
3	SOFTWARE SPECIFICATIONS	F-21
3.1	Initialization	F-21
3.2	System Unit model	F-21
3.3	Interface signal standards	F-21
3.4	Communication protocols	F-22
3.5	Self-test	F-22
3.6	Initialization acknowledge	F-22
3.7	Scan codes	F-23
3.8	Special case handling	F-23
3.8.1	Separation of numeric pad and cursors	F-23
3.8.2	Numlock	F-24
3.8.3	Lock indicators	F-24
3.9	Typematic action	F-25
3.10	Ring buffer	F-25
3.11	PROM Size	F-26

4	ELECTRO-MECHANICAL SPECIFICATION	F-27
4.1	Keystation layout and keytop colours	F-27
4.2	Function pad	F-27
4.3	Mouse port	F-29
4.4	LED layout	F-30
4.5	Bar Code Reader	F-30
4.6	System Unit cable and connectors	F-30
4.7	Other engineering specifications	F-31
5	KEYBOARD TO SYSTEM UNIT COMMUNICATION	F-32
5.1	Communication with standard IBM PC	F-32
5.2	Communication with IBM PC-AT	F-33
5.2.1	Standard commands from DOS/BIOS	F-33
5.2.2	Standard messages from keyboard	F-34
5.2.3	Additions to standard commands to the keyboard	F-35
6	CONVERSION OF KEYSTROKE TO SYSTEM SCAN CODES AND BIOS OUTPUT .	F-36
6.1	Direct IBM PC equivalent keys	F-37
6.2	NOTIS-compatible keys	F-39
6.3	Keys not having same scan codes in all states	F-40
6.4	Mouse push buttons	F-42
7	SYSTEM TO "AT" SCAN CODE CONVERSION TABLE	F-43

CHAPTER 1
OVERVIEW AND DATA SUMMARY**TABLE OF CONTENTS**

Section		Page
1.1	INTRODUCTION	1-5
1.2	SYSTEM CONFIGURATION	1-7
1.2.1	Basic hardware	1-7
1.2.2	Cluster configuration plus options	1-8
1.2.3	Communications Controller	1-9
1.2.4	Summary of components	1-10
1.3	BUTTERFLY-110 SYSTEM UNIT	1-12
1.3.1	PC features	1-12
1.3.2	ND-110PCX	1-12
1.4	BUTTERFLY DISPLAY UNIT	1-13
1.5	BUTTERFLY KEYBOARD /MOUSE /BAR CODE READER	1-13
1.5.1	Keyboard	1-13
1.5.2	Mouse	1-14
1.6	CENTRONICS PRINTER PARALLEL INTERFACE	1-14
1.7	SERIAL INTERFACE	1-14
1.8	DATA SUMMARY	1-15
1.8.1	Mechanical	1-15
1.8.2	Electrical	1-15
1.8.3	Environmental	1-16

LIST OF ILLUSTRATIONS

Figure		Page
1.1	The Butterfly-110	1-5
1.2	Example Teamstation configuration	1-6
1.3	Interconnection of units	1-7
1.4	Enhancement capability in the Butterfly-110 system	1-8
1.5	Butterfly-110 communications ports	1-9

1.1 INTRODUCTION

BUTTERFLY-110 (fig 1.1) is a ND-110PCX computer which coexists with an IBM-compatible PC-AT computer, inside the enclosure of the PC-AT.

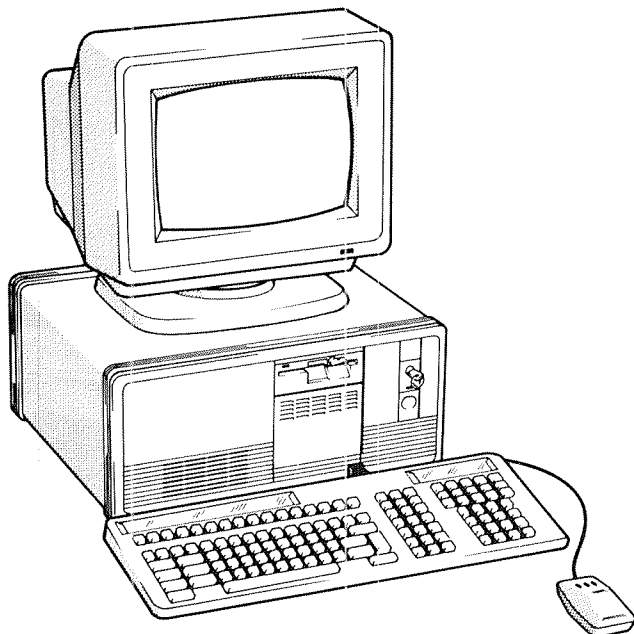


Fig 1.1: The Butterfly-110

The OWS-110 offers in one unit the functionality of an IBM PC-AT, with all its currently available software running under MS-DOS, coupled with the extensive range of ND software (e.g. the NOTIS range) which runs under ND's SINTRAN operating system. Information in OWS-110 is freely interchangeable between the PC and ND operating systems, so that files created using the PC system may then be used by the ND system, and vice versa.

A centronics parallel printer port and RS232 serial port are provided as standard. Also included as standard is a Mouse, which connects to the Keyboard. A Bar Code Reader port is included in the keyboard, to suit connection of a Bar Code Reader wand.

Options include a Communications Controller board. This board provides a further 4xRS232 serial ports, for connection of ND-compatible terminals/devices to Butterfly-110. When one or more terminals are connected to these serial ports, then a clustered environment is created around the Butterfly-110, with the terminals running from the ND-110PCX. This configuration is called a Butterfly TEAMSTATION. An example of such an arrangement is shown in fig 1.2.

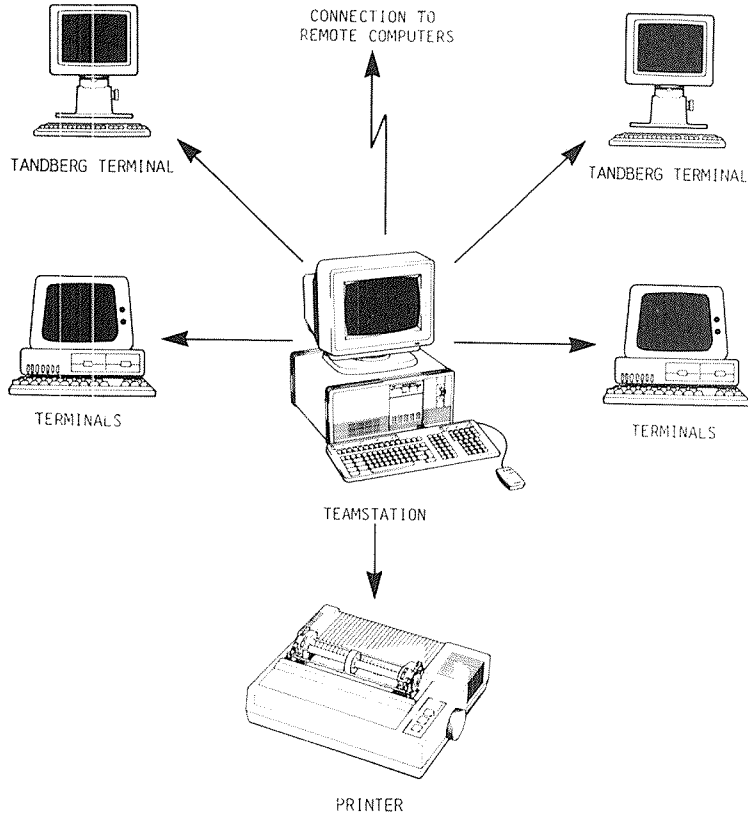


Fig 1.2: Example Teamstation configuration

In this configuration, the Butterfly-110 User may run ND or IBM-PC-compatible application packages, while the other users in the cluster may run ND application packages.

If these cluster terminals are also PCs, then a user environment is created in which all the terminals may be operated as PCs and as ND terminals.

1.2 SYSTEM CONFIGURATION

1.2.1 Basic hardware

Fig 1.3 shows the interconnection between the main units in Butterfly-110:

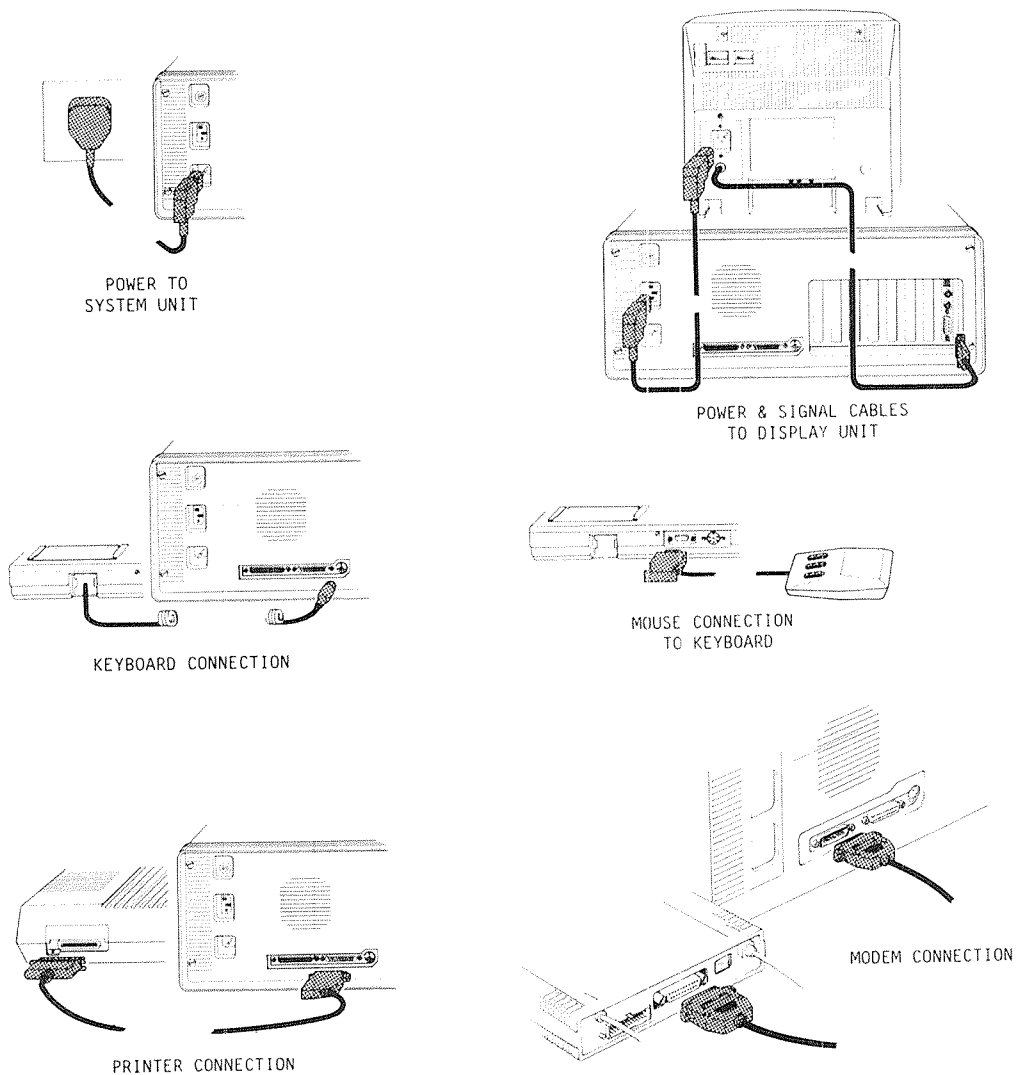


Fig 1.3: Interconnection of units

1.2.2 Cluster configuration plus options

Fig 1.4 illustrates how the basic Butterfly-110 may be enhanced, by adding:

- a Communications Controller board, with connector box. See section 1.2.3.
- extra 1Mbyte RAM (to give a total of 2Mbytes).
- HDLC communications to a local computer system, for example, using ND's Computerlink.

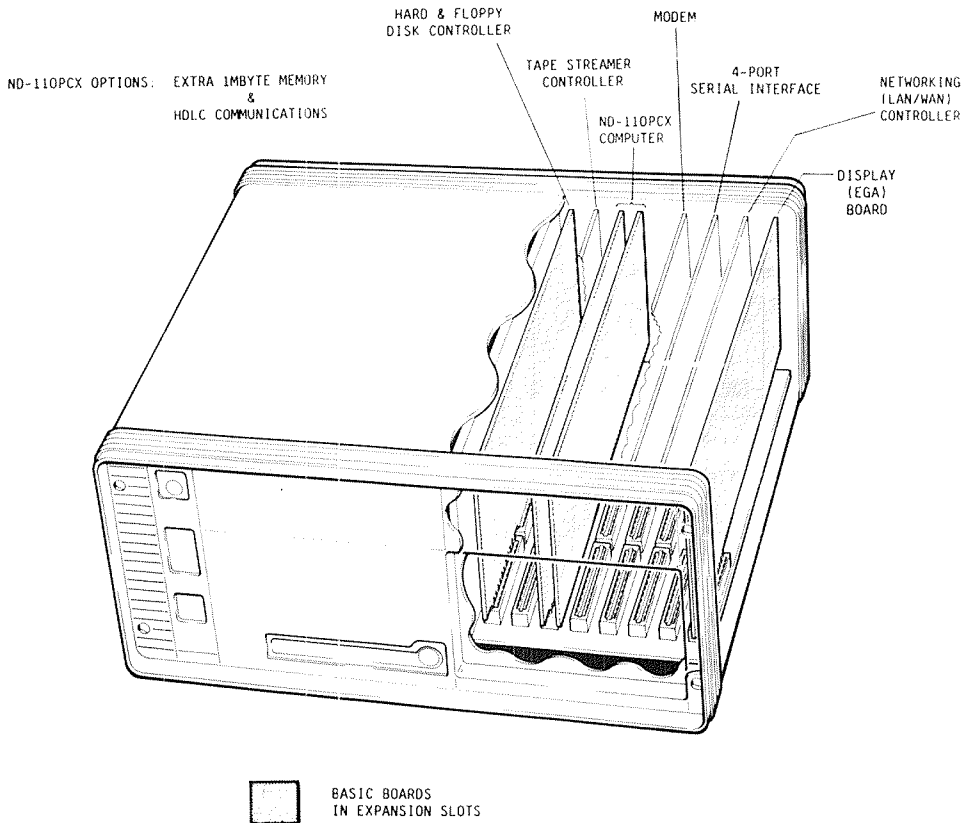


Fig 1.4: Enhancement capability in the Butterfly-110 system

Space is available in the expansion card cage of the PC-AT to fit further cards. For example:

- a tape streamer controller card, connecting to a Tape Streamer cartridge device.
- an asynchronous modem communications card.
- a networking (LAN/WAN) interface card.

In addition to the Mouse, the Keyboard includes facility to connect a Bar Code Reader and an external Function Keypad.

1.2.3 Communications Controller

Fig 1.5 shows how adding the InterQuadram Communications Controller board extends the Butterfly-110 terminal to provide TEAMSTATION capability for up to four more terminals.

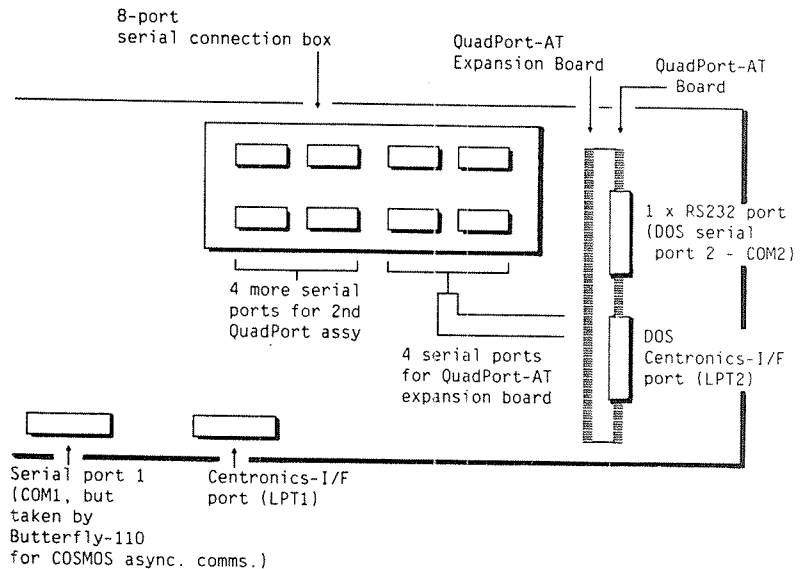


Fig 1.5: Butterfly-110 communications ports

DOS serial ports 1 and 2	<p>Serial port 1 (native to the PC) is the DOS "COM1" port. In Butterfly-110, it is taken away from DOS by the Butterfly software which runs on the PC, and used for ND's COSMOS asynchronous communication facility.</p> <p>Serial port 2 on the Quad-port-AT board is the DOS "COM2" port. To allow DOS to use this port, the Butterfly-110 software in the PC redirects it to the new value for COM2.</p>
Parallel ports LPT1 and LPT2	<p>Port LPT1 (native to the PC) provides a centronics-compatible interface for connection of a suitable printer. The print-spooling software shares this port between the PC's DOS and ND's SINTRAN operating systems.</p> <p>Port LPT2 also provides a centronics-compatible interface. It is however only available to DOS.</p>
Quad-port "TEAMSTATION" serial ports	<p>A connection on the Quad-port-AT Expansion board provides the interface for four serial ports. These are brought out to a 4-port serial connection box, which mounts on the rear of the System Unit. Any serial-driven device supported by the software running on the ND-110PCX may be connected to this interface - e.g. ND-compatible terminals, or serial printer.</p> <p>If ND-compatible terminals are connected, they may run any of the ND application programs available in Butterfly-110. Any OWS-10/12 terminal which is also a PC may then be used both as a stand-alone PC and as an ND terminal running on Butterfly-110's ND-110PCX.</p>

1.2.4 Summary of components

Standard hardware	System Unit (floppy disk and 40Mbyte hard disk, RS232 port and parallel port), Colour Display Unit with EGA graphics, Butterfly Keyboard (NOTIS and PC compatible), Mouse.
Operating system software	<p><u>MS-DOS for the PC</u></p> <p>Additional Butterfly-110 programs run under the PC's MS-DOS to handle communication between the PC's 80286 microprocessor and the ND-110PCX, and to accommodate the System Manager and associated software which enables the PC and the ND-110PCX to share the same display, keyboard, disk storage, printer, etc.</p> <p><u>SINTRAN III/VSX for the ND-110PCX</u></p> <p>This is extended to handle the integration of the ND-110PCX with the PC environment, particularly with regard to its method of communication with the PC (IOX simulation), and provision of new Servers for handling tasks between the ND's SINTRAN and PC's MS-DOS operating systems.</p>

-
- | | |
|--------------------------------|---|
| Basic PC software | <ul style="list-style-type: none">• Tutorial: this is a "getting started" self-teach aid.• PC-Diagnostics for Butterfly-110• Display controller (EGA) utilities. |
| ND applications software | <ul style="list-style-type: none">• NOTIS-WP• User Environment for Butterfly-110• DO system for Butterfly-110• COSMOS (with server for asynchronous line)• Telefix files• Backup-System• File-Manager• NOTIS-DS/ID for Butterfly-110• ND-Diagnostics for Butterfly-110 |
| ND software options | <ul style="list-style-type: none">• Accounting system• Job execution control• Symbolic debugger• BRF linker• Subsystem package• Subsystem package II |
| Butterfly-110 Hardware options | <ul style="list-style-type: none">• Communications Controller board (5xserial plus 1xparallel ports) with 4-port serial connection box (see section 1.2.3).• HDLC communications (up to 153.6 kbaud)• Extra 1Mbyte RAM (to give a total of 2Mbytes)• Extra Terminals, for connection to the 4-port serial interface ports (to create a multi-user Teamstation system).• Parallel-interface printer, on one of the parallel interface ports.• Bar Code Reader• Keyboard Function Keypad (future option). |

1.3 BUTTERFLY-110 SYSTEM UNIT

1.3.1 PC features

Features of the PC include:

- Intel 80286 microprocessor
- eight I/O slots
- 512 kbytes memory (normally expanded to 640 kbytes, by donating 128 kbytes of ND-memory to PC-DOS)
- one RS232 serial port
- one Centronics-compatible parallel printer port
- one 40-45Mb Winchester disk with fast access time
- one 1.2Mb/360Kb floppy disk
- a keyswitch.

1.3.2 ND-110PCX

The ND-110PCX computer comprises a two-board assembly (CPU board, and Local Bus board), which plug into one slot in the PC expansion card cage.

CPU Board

- processor
- memory management
- microprogram control
- console port

Local Bus Board

- Input/Output control (program I/O)
- 1Mbyte dynamic random access memory, with option to extend by a further 1Mbyte.
- Real-time clock
- HDLC, up to 153.6 kbaud.
- Interrupt system
- PC bus interface

1.4 BUTTERFLY DISPLAY UNIT

The colour display offers full 16-colour from a palette of 64 colours, on a 12-inch or 14-inch crt. The display area provides 25 lines of 80 characters per line. Graphic resolution may be 640x350 pixels or 320x200 pixels.

A monochrome display may be offered if required. Likely features would include:

- "black and white" display
- 12-inch crt, high resolution
- display area: 25 lines of 80 characters per line
- graphic resolution 640 x 350 pixel.

2 BUTTERFLY KEYBOARD /MOUSE /BAR CODE READER

2.1 Keyboard

The keyboard combines the functionality of a standard IBM-PC keyboard with that of a ND-NOTIS keyboard.

It provides "soft" Push-keys, programmable by the user to hold multi-key sequences.

It includes a mouse interface/port.

It includes a bar code reader interface/port, for connection of a wand (pen unit).

2.2 Mouse

The Mouse is a standard 3-button device. Each button may be programmed as a Push-key. The Mouse may be configured for left or right-handed operation.

3 CENTRONICS PRINTER PARALLEL INTERFACE

The System Unit provides an IBM-type 25-way D-type connection for centronics-compatible printers.

4 SERIAL INTERFACE

This is fully compatible with IBM-PC serial ports:

- supports communication at 9600 baud over all ports simultaneously
- offers RS232C/V24
- the software controls set-up of UART parameters
- transmission speeds may be 50-19200 baud
- character length may be 5, 6, 7, or 8 -bit
- number of stop bits may be 1, 1.5, or 2
- parity generation and checking is included.

This option is available to connect any serial device to Butterfly-110.

5 DATA SUMMARY

5.1 Mechanical

System Unit	<ul style="list-style-type: none"> • Height - 183 mm • Width - 440 mm • Depth - 410 mm • Weight - 20 kg (approx.) 		
Colour Display Unit		<u>Hitachi (12-inch)</u>	<u>Wyse (14-inch)</u>
	<ul style="list-style-type: none"> • Height - 295 mm • Width - 330 mm • Depth - 380 mm • Weight - 13 kg (approx) 	336 mm 362 mm 387 mm	336 mm 362 mm 387 mm
Keyboard	<ul style="list-style-type: none"> • Height - 58 mm (maximum) • Width - 520 mm • Depth - 192 mm • Weight - 2 kg (approx) 		

5.2 Electrical

Nominal voltage	-	115V or 230V, 47 to 63 Hz
Power consumption	-	750VA (max), including switched output
Inrush current	-	30A peak
Line voltage	-	190-255V 50 Hz 95-135V 60 Hz
Transients		
high energy	-	+/-500V, 0.1µs rise time, 50µs half-value width
low energy	-	+/-1000V, 5ns rise time, 0.1µs half-value width

5.3 Environmental

Acoustic noise	-	max 45dbA (no printer, no floppy seek)
Operating temp	-	10 to 35 °C
Relative humidity	-	20 to 85% RH, non-condensing
Vibration	-	5-150 Hz, acceleration 5 mm/s ² displacement 0.035 mm
Shock	-	1-3 Hz, amplitude 10 mm (operating) 50 mm (storage).
Altitude	-	sea-level to 10,000 ft (approx 3000 m)

CHAPTER 2 SYSTEM AND HARDWARE

TABLE OF CONTENTS

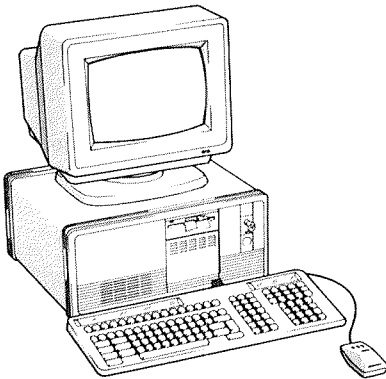
Section	Page
2.1 OVERVIEW	2-5
2.1.1 Butterfly Main Units	2-5
2.1.1.1 System Unit	2-6
2.1.1.2 Keyboard	2-7
2.1.1.3 Display	2-7
2.1.1.4 Mouse	2-7
2.1.1.5 Hardware options	2-8
2.1.2 System Control	2-8
2.1.2.1 Functional elements	2-8
2.1.2.2 System start-up	2-9
2.1.2.3 I/O control	2-10
2.1.2.4 ND-PC communication	2-10
2.2 ND-110PCX FUNCTIONAL BLOCKS	2-12
2.2.1 Microprogram and processor (CPU)	2-12
2.2.2 Register File	2-13
2.2.3 Interrupt System	2-13
2.2.4 Memory Management System (MMS)	2-14
2.2.5 Test (console) port	2-15
2.2.6 Print Status	2-15
2.2.7 Installation Number	2-15
2.2.8 Memory (RAM)	2-16
2.2.9 I/O Control	2-18
2.2.10 Oscillator	2-18
2.2.11 Power Fail	2-18
2.3 THE KEYBOARD	2-19
2.3.1 Summary of Features	2-19
2.3.2 IBM Compatibility	2-23
2.3.3 ND-NOTIS Compatibility	2-23
2.3.4 Added Features	2-24
2.3.4.1 Mouse	2-24
2.3.4.2 Bar Code Reader port	2-25
2.3.4.3 Function Keypad option	2-25
2.3.4.4 Template cards	2-26

LIST OF ILLUSTRATIONS

Figure		Page
2.1	Butterfly main units	2-5
2.2	Basic PC System Unit	2-6
2.3	ND-110PCX in PC expansion chassis	2-7
2.4	Butterfly-110 System Unit - block diagram	2-8
2.5	ND-PC communication	2-10
2.6	Functional blocks in the ND-110PCX	2-12
2.7	ND-110PCX memory map	2-17
2.8	Butterfly keyboard - ND/IBM US-English (VT100)	2-20
2.9	Butterfly keyboard - ND/IBM International-English (ANSI)	2-21
2.10	Butterfly keyboard - 8 special PC keys (green legend)	2-22
2.11	Mouse pushbuttons and movement	2-24
2.12	Push-keys template	2-26
2.13	Function Keypad template	2-26

2.1 OVERVIEW

2.1.1 Butterfly main units



The main units in the Butterfly-110 are the:

- system unit
- keyboard
- display
- mouse

Fig 2.1: Butterfly main units

The System Unit comprises the Ericsson IBM-compatible PC-AT personal computer, with an ND-110PCX computer (on a single assembly comprising two PC-size expansion boards) fitted into its expansion card cage. This standard PC-AT configuration includes 512-kbytes of RAM, a 360/1200-kbyte floppy disk drive, and a 40-Mbyte internal Winchester hard disk.

The keyboard is specific to Butterfly. It represents two functional keyboards in one unit. In one mode of operation, it emulates the standard IBM-PC keyboard. In the other, it performs as a ND-NOTIS keyboard. It includes control interfaces for the Mouse and Bar Code Reader.

The Mouse and the Bar Code Reader are standard products from external suppliers. They plug in directly to the keyboard. The Mouse is supplied as part of the basic Butterfly system; the Bar Code Reader (wand) is an optional extra.

The Display is available in two forms: full colour, and monochrome.

2.1.1.1 System unit

The base elements in the PC parts of the System Unit are:

- microprocessor-controlled main logic board incorporating memory (640 kbytes).
- 1.2Mbyte/360kbyte floppy disk control and drive
- display controller (EGA card)
- keyboard input
- parallel printer (Centronics-compatible) interface
- RS232 serial port
- 40Mbyte Winchester hard disk control and drive
- power supply
- 8-slot expansion card cage.

All these components are described in the associated PC Technical Manual (ND-06.028, or in supplier's technical manuals.

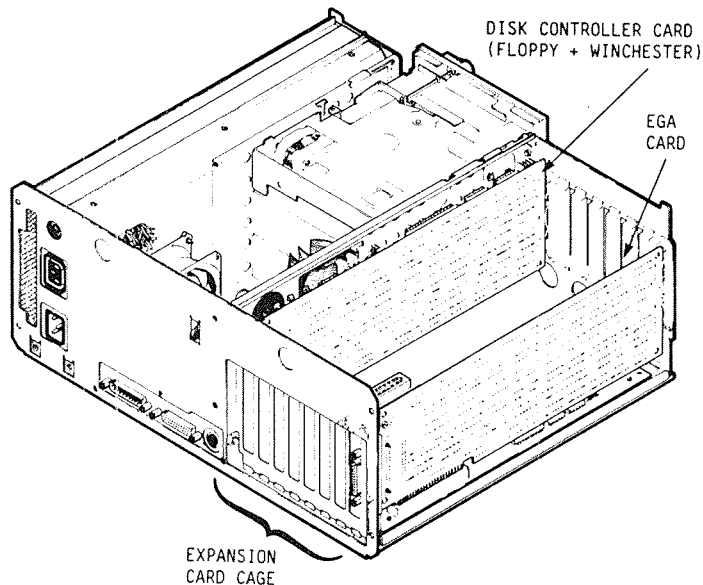


Fig 2.2: Basic PC System Unit

The basic Butterfly-110 System Unit takes the base PC unit (fig 2.2), and adds the 2-board assembly comprising the ND-110PCX computer (fig 2.3), which plugs into the PC's expansion card cage. The two boards form an interconnected sandwich, with component sides facing outwards.

A functional overview for the ND-110PCX computer is presented in section 2.3. A more detailed hardware description is given in Chapter 4. Microprogram is described in Chapter 5.

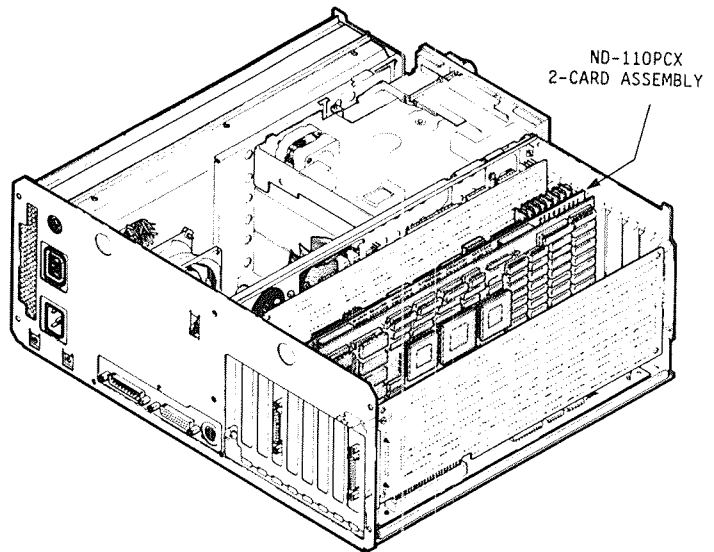


Fig 2.3: ND-110PCX in PC expansion chassis

2.1.1.2 Keyboard

The Keyboard specification is given in Appendix F.

2.1.1.3 Display

Technical descriptions for the monochrome and colour display units are included in associated technical manuals.

2.1.1.4 Mouse

This is described in its associated technical manual.

2.1.1.5 Hardware options

The currently available Butterfly-110 enhancement options are listed in Chapter 1. Those of ND design (extra 1Mbyte DRAM, HDLC communications) are described in technical detail in Chapter 4. All other hardware options are the subject of associated technical descriptions, available from the relevant suppliers.

2.1.2 System control

2.1.2.1 Functional elements

Fig 2.4 shows a hardware block diagram for the Butterfly-110 System Unit. Essentially, it comprises a ND-110PCX computer - functionally equivalent to the ND-110CX (RASK) computer - integrated with an IBM-compatible PC-AT. In the ND environment, the fundamental role of the PC is to operate as an "intelligent" I/O system for the ND-110PCX.

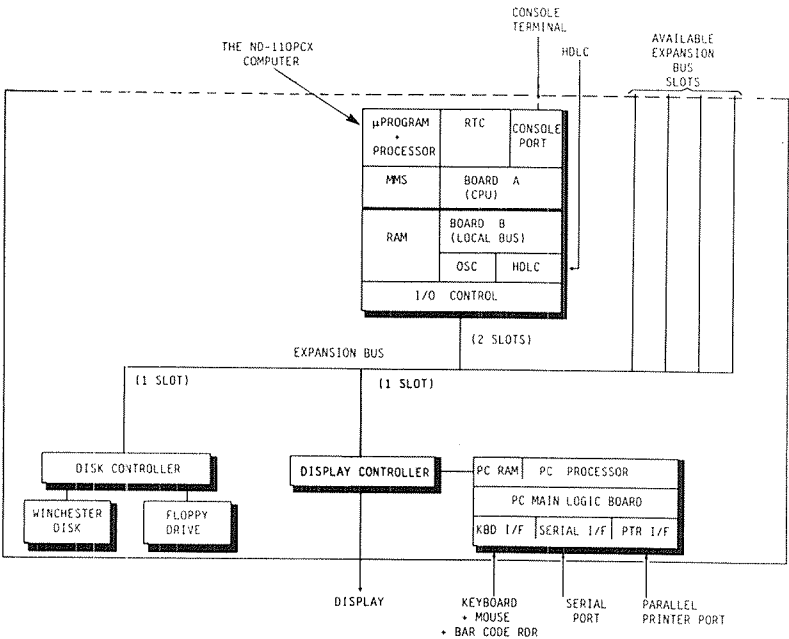


Fig 2.4: Butterfly-110 System Unit - block diagram

Fig 2.4 shows the PC elements as comprising:

- the PC Main Logic Board.

This incorporates the CPU, memory, floppy disk interface, keyboard interface, RS232 serial interface and parallel printer interface.

- the display controller
- the expansion control buffer to the PC Bus for the 8-slot expansion card cage
- the Winchester hard disk and floppy disk, and the disk controller card (which occupies one slot on the PC expansion bus
- the ND-110PCX computer, occupying two adjacent slots in the PC-AT expansion chassis. It uses the PC as an "intelligent" controller for its I/O requirements via the PC Bus. The PC part is of course available to run any PC programs, independently of the ND-110PCX.
- 4 further expansion bus slots available in the system.

A functional overview of the PC elements is given in the PC Technical Reference Manual (ND-06.028).

2.1.2.2 System start-up

MS-DOS 3.1 is the chosen operating system for the PC part of the Butterfly machine.

SINTRAN III/VSX is the operating system for the ND-110PCX.

This software, together with the specified MS-DOS and ND application programs, plus supporting software utilities, is installed on the internal Winchester hard disk drive, prior to delivery.

When the machine is switched on, the system first performs a self-test on the PC part of the machine. On successful completion, it loads MS-DOS and other PC programs into the PC. Then it tests the ND-PC communication, and if this passes the minimum requirements, it runs self-test on the ND-110PCX. On successful completion of these self-tests, it loads SINTRAN into the ND-110PCX.

2.1.2.3 I/O control

The block diagram in fig 2.4 shows that I/O control for the Butterfly-110 keyboard and display, floppy and hard disks, serial port (COM1) and printer port (LPT1), resides with the PC processor. I/O operations with these devices are therefore performed under MS-DOS.

Putting the required I/O control functionality into the Butterfly-110 MS-DOS environment involves making modifications and additions to various areas of the PC software. These are presented as a functional overview in Chapter 3, with a more detailed description in Chapter 6.

It may be noted that any further expansion cards inserted into Butterfly-110 (e.g. QuadPort serial interface, streamer tape controller, modem) must also operate under I/O control from the PC. So, whenever the ND computer initiates a requirement to perform I/O with such a device, it effectively commissions the PC to do that I/O operation.

2.1.2.4 ND-PC communication

There are four types of "input/output devices" that the ND-110PCX sees across its interface to the PC:

- mailbox, for I/O transfers with most peripheral devices connected to the PC side of Butterfly-110 - see section 2.1.2.3
- ND "terminal-1" input } for OPCOM, using the
- ND "terminal-1" output } Butterfly-110 keyboard and display as "terminal-1"
- calendar, for accessing the calendar device in the PC side of the machine.

Additions are made to the ND and PC software to provide this communication between the PC's 80286 and ND's ND-110PCX processors.

The essence of this communication scheme is that both have access to an area of ND memory, called the "mailbox" (fig 2.5). This is organised into discrete areas for operational purposes, but the concept is that tasks are "posted" from one processor to the other, through mailboxes.

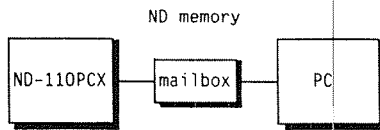


Fig 2.5: ND-PC Communication.

If the PC requires the ND to perform a task, it specifies that task by writing it into mailbox, then sends an interrupt to the ND to advise it that a mailbox requires service.

The ND communicates with the PC in the same way.

The modifications and additions made to both the ND software and the PC software to accommodate this ND-PC communication are presented as a functional overview in Chapter 3, with a more detailed description in Chapter 6.

2.2 ND-110PCX FUNCTIONAL BLOCKS

This section presents an appreciation of the main functional areas in the ND-110PCX. The description is based on a section of the function block shown in fig 2.4, which is reproduced in fig 2.6.

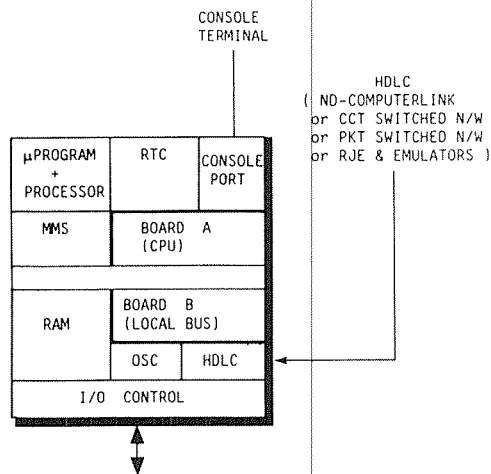


Fig 2.6: Functional blocks in the ND-110PCX

2.2.1 Microprogram and processor (CPU)

The central processing unit (CPU) is a 16-bit processor, controlled by an 8kx64-bit microprogram. Each machine (macro-) instruction is implemented through execution of a number of microinstructions which reside in the microprogram control store, along with instruction mapping and the HDLC simulation.

Three gate arrays (which are common to the ND-110CX (RASK) computer) are used:

- MIC (microprogram instruction controller)
- MAC (macroprogram address controller)
- ALU (arithmetic and logic unit)

MIC

This chip determines the addresses for each microprogram instruction, generating the correct sequence of Control Store addresses, for microprogram to execute each successive machine instruction. This sequence takes into account the status of any machine conditions that the instructions provoke. It also supports addressing of the internal CPU working registers (the Register File).

MAC

The MAC computes the memory addresses for all CPU-controlled memory accesses, both for instruction fetches and data read/writes.

ALU

The ALU is a 16-bit arithmetic and logic unit, which performs the same function as the 4xAM2901 ALU chips of the ND-100, but which also incorporates features that enable it to work closely with the relevant control bits of the microinstruction word, and so optimise ALU operations.

2.2.2 Register File

The Register File contains 16 register sets, one set for each of 16 program levels within the CPU. These levels are denoted 0 (lowest priority) to 15 (highest priority). Of the 16 registers in each level, 8 are available to hold the program level context information (7 MAC registers plus 1 status register), this representing the current CPU working condition at that level. The other 8 registers (8x16 levels) are used by the microprogram, as scratch area.

At any one time, one of these sets of 16 registers - corresponding to the current interrupt level of the CPU - is the current register set that is being used by the CPU.

2.2.3 Interrupt system

The ND-110PCX includes the standard ND-100-series priority interrupt system (interrupt levels 0-15), which permits the processor to be interrupted by conditions inside or outside

the CPU. Each level is assigned a complete set of working registers, which are located in the Register File. Context switching from one level to another consists of changing to another set of working registers. This is controlled by the microprogram, with addressing control via the MIC.

In Butterfly-110, external interrupts may be generated from the PC (for any of the I/O devices attached to the PC, or for access to shared files), the HDLC facility, or the Test (console) port. In respect of ND-PC communication, the PC may interrupt the ND, and the ND may interrupt the PC. PC interrupts to the ND are handled on a special Butterfly-110 software level-16, which is higher priority than level 15 (the highest priority in the ND-110CX structure).

For ND interrupts to the PC, the PC is interrupted on the IRQ5 line of its expansion bus.

2.2.4 Memory management system (MMS)

MMS is based on two major subsystems:

- Paging
- Memory Protection system.

Paging

The Butterfly-110 memory system works in what in ND-100/110 computer terms is known as "extended" mode. This allows the CPU an addressing capability up to 16M words. The ND paging system organises memory into pages which are 1kwords in size. A page map in the MMS then translates what the MAC generates as a "logical" (virtual) address into a "physical" address to the memory system.

Protection

The memory protection system provides the same features as in the ND-100:

- Page Protect
- Ring Protect.

The Page Protect system allows a page to be protected from read, write or instruction fetch accesses, or any combination of these.

The Ring Protect system places each page program on one of four priority rings. A page on one specific ring may not be accessed by a program that is assigned a lower priority ring number. This system is used to protect system programs from user programs, the operating system from its subprograms, and the system kernel from the rest of the operating system.

2.2.5 Test (console) port

The ND-110PCX incorporates microprogram for communication between the operator and the machine. This program is called MOPC (Microprogrammed Operator's Communication) and is also referred to as "OPCOM". It is used for operational control of the ND-110PCX. It includes such functions as memory and register examine and deposit, breakpoint control, and bootstrap loading.

A UART on the CPU board provides this test port into ND-110PCX, which connects to a console "terminal-1". Alternatively the Butterfly-110 display and keyboard may be used as the terminal-1 device.

2.2.6 Print status

This is an 8-bit wire-link station which may be configured to indicate the modification status of the CPU board. This status may be examined by the macroinstruction VERSN.

2.2.7 Installation number

Butterfly-110 uses an Installation PROM piggy-back board. There may be up to six PROMs fitted to this board. These PROM(s) contain encoded information which characterise the machine with a unique identity (a CPU number). They also describe what software versions and hardware configurations are authorised to be run.

The information in these PROMs is used when Butterfly-110 is powered-up. As well as checking for authorised use of the various configurations available in Butterfly-110, it provides identification information when the CPU is part of a network, and allows system software to be configured for each machine. This is described further in Chapters 4 (hardware description) and 7 (use of configuration by the system).

Each installation PROM chip contains a copyright statement, so rendering it illegal to copy them without written authority from ND.

2.2.8 Memory (RAM)

Memory space

The memory consists of:

- 1Mbyte of dynamic random access memory (DRAM), optionally expandable to 2Mbyte.
- 2kbyte of non-volatile RAM (NVRAM). This is expandable up to 8kbyte of addressing.
- 8k of PROM, expandable up to 32k of addressing.

DRAM has 1 bit of parity checking for each byte.

There is no cache memory in ND-110PCX, though the bus architecture does retain the cache address and data highways, as remnants of the conversion of the ND-110CX (RASK) into ND-110PCX.

Requests for memory access may arise from the ND-110PCX CPU, the PC, the HDLC I/O, and the refresh control logic for DRAM.

The Memory Map of fig 2.7 shows the addressing range allocation for DRAM (up to 1M), PROM (up to 32k), NV-RAM (up to 8k), and "ND I/O" for HDLC (8k).

The ND I/O area of memory is not actual memory; instead, any memory-reference operation to an address within this address range performs an I/O operation with the HDLC I/O subsystem of the ND-110PCX.

Extended PC memory

The PC has 512kbytes of its own RAM. Since it is highly desirable to increase this to 640kbytes, an additional 128kbytes are "donated" to the PC from the ND's memory space. This facility is enabled via a link on the ND-110PCX. When enabled, the donated 64kwords of ND DRAM (allocated in the address range 0M256-0M320) cannot be used by the ND; it can only be accessed by the PC.

Logical/physical address

Fig 2.7 also shows the relationship between logical memory address ranges in the ND-110PCX, and physical address ranges as applied to the memory. In particular, it shows how memory address range 8M048-8M064 (part of the "secret memory" area) is physically mapped to DRAM memory addresses 240-256k. This is a reserved memory area, used among other things for the "mailboxes" in ND-PC communication.

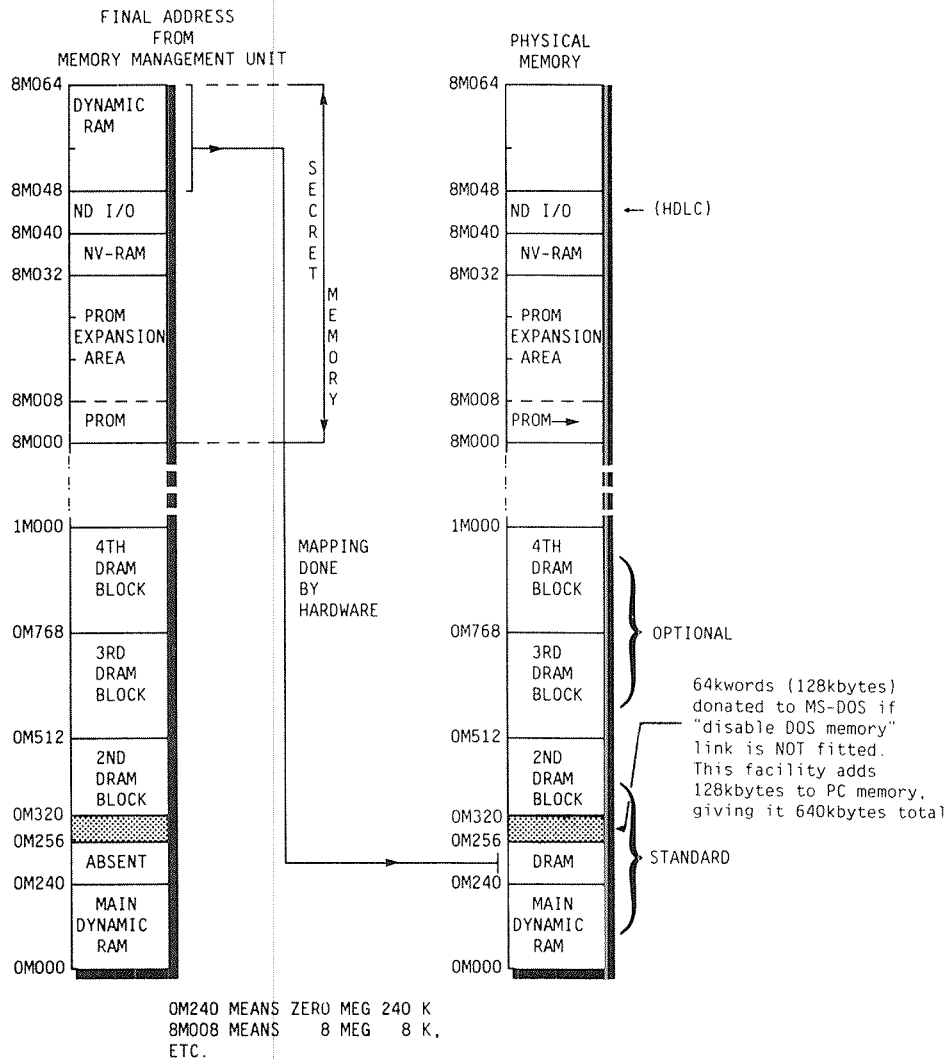


Fig 2.7: ND-110PCX memory map

2.2.9 I/O control

The ND-110PCX has three I/O channels:

- via the PC, for all I/O operations except HDLC and the test (console-UART) port.
- via the test (console) port to a console terminal (for engineering operational access to the ND-110PCX).
- via the HDLC facility, if this option is fitted.

I/O operations with the PC are set up via the "mailbox" arrangement. Where appropriate (e.g. for block transfer of data between disk and ND memory), the actual data transfer is effected directly with ND memory, under memory access control from the PC. The PC can access all of the ND's address space. It does this by specifying an "offset" address, which points to the required 64kbyte block of ND memory.

Communications facility in ND-110PCX is provided by the HDLC/DMA subsystem. This provides CCITT RS422 (X.21/X.27) and RS232 (V.24), and ND's "computer-link" protocols. It does not offer ND's "mega-link" protocol.

2.2.10 Oscillator

ND-110PCX timing is generated from a 39.3216 MHz crystal oscillator. This provides the basic 25 nano-second timing interval of the ND-110PCX. All other timings in the ND, including the real-time clock, are derived from this.

2.2.11 Power fail

The standard power fail and master clear facilities of the ND-100 are included in the ND-110PCX. A power fail condition is sensed and generates an interrupt to the CPU, as a result of which the ND operating system takes the necessary steps towards a well defined stop point, with its registers saved in memory. When the main power is restored, sensed by the Power Fail logic, the operating system goes through a restart procedure, enabling the executing programs to resume. In Butterfly-110, this power fail detection includes use of the PC's PCFAIL flag. It is also used to generate an interrupt to the ND if the PC fails to respond to an ND-to-PC interrupt.

2.3 THE KEYBOARD

2.3.1 Summary of features

The Butterfly keyboard provides plug-compatibility with an IBM-PC keyboard. It also provides the features required to operate as an ND-NOTIS keyboard.

Language key layout variants are available as follows:

- ND/IBM US-English (VT100). The key layout for this is shown in fig 2.8.
 - ND/IBM International-English (ANSI). The differences in key layout from the US-English (VT-100) variant are shown in fig 2.9.
 - Norwegian
 - Danish
 - Swedish
 - German
 - French
- The differences from the US-English (VT100) key layout are included in Appendix F.

There are several keys with special PC significance, and these have added legends in green; these keytops are identified in fig 2.10.

In addition, the Butterfly keyboard provides:

- a Mouse interface inside the keyboard
- a Bar Code Reader interface inside the keyboard
- port for an external Menu/Function Pad
- positions above the keypad area for insertion of two templates. One may be used to mark the user-programmed contents of the three alternative sets of eight user-defined Push-keys, plus three Mouse buttons. The other is provided for marking the functions of each key in the Function Keypad.

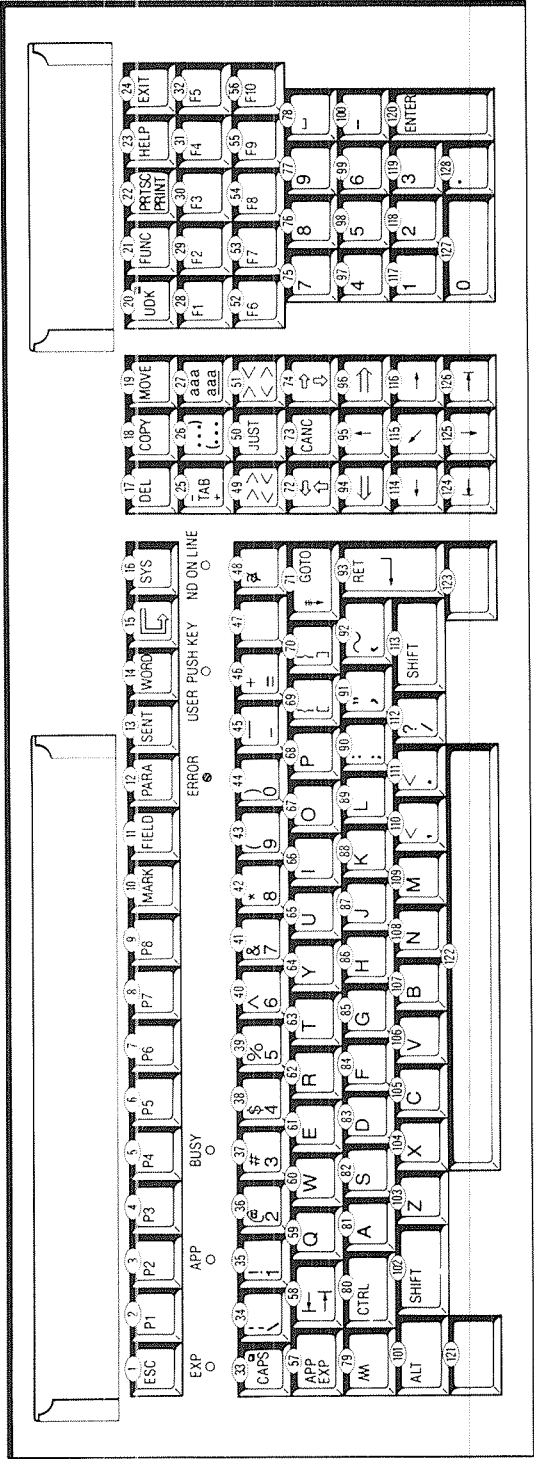


Fig 2.8: Butterfly keyboard - ND/IBM US-English (VT100)

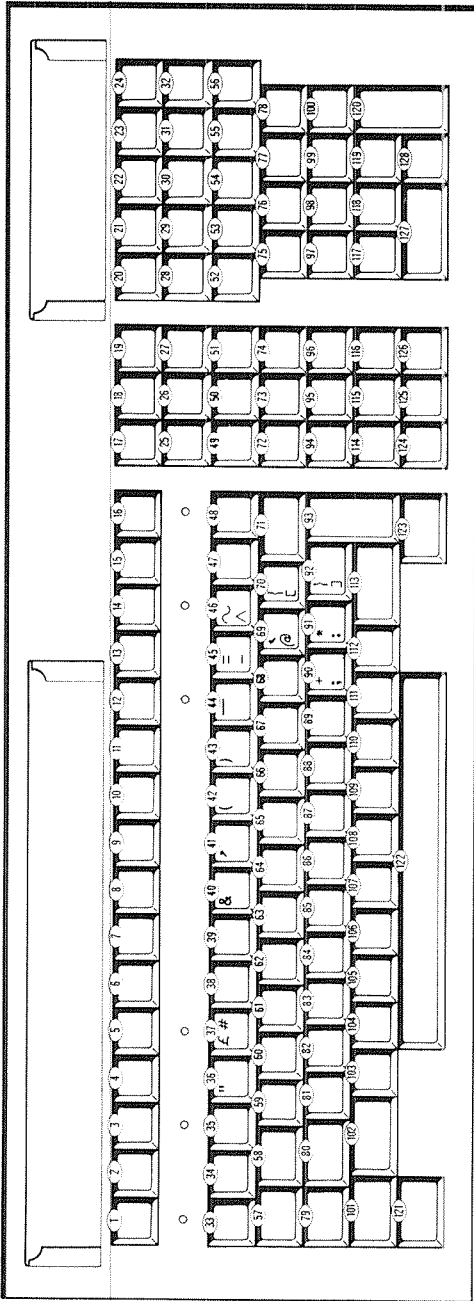


Fig 2.9: Butterfly keyboard - ND/IBM International-English (ANSI)

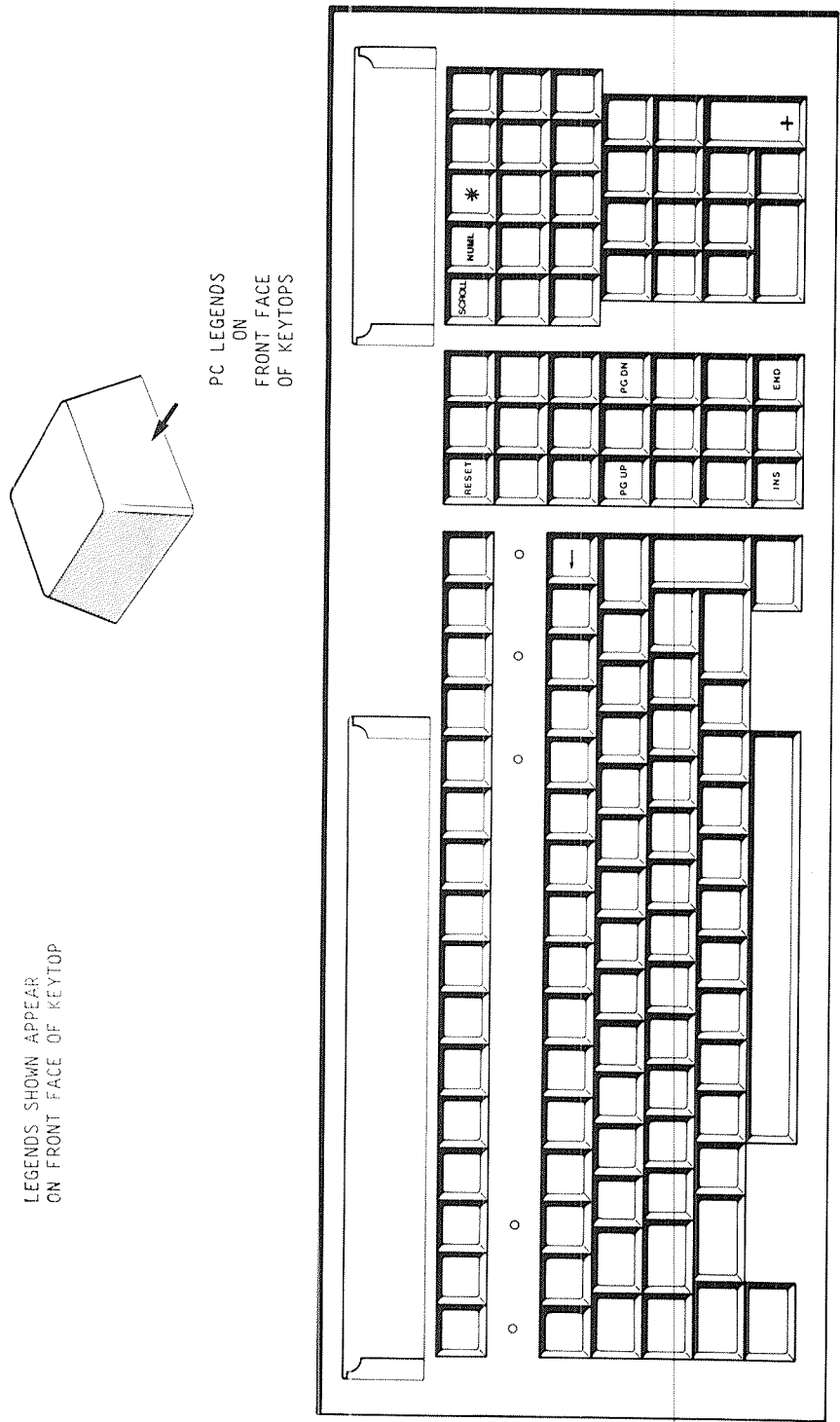


Fig 2.10: Butterfly keyboard - special PC keys (green legend)

2.3.2 IBM compatibility

The keyboard works with the IBM-assigned keys, without changing the keyboard driver in the basic I/O system program (BIOS) of the PC. The keystrokes that are output by the driver are converted by a special keyboard handler, before they are sent on to the ND-110PCX. It is this handler that converts keystrokes to the same codes and escape sequences that an ND-NOTIS terminal produces.

However, a large number of special NOTIS keys are not recognized by BIOS. To allow these codes to pass, as well as to enable programming of Push-keys, the keyboard driver is lifted out of BIOS, and the existence of the new keys defined. In this way, all the IBM key functions are implemented with the correct IBM Scan Code from the keyboard - even the apparently redundant Numlock key.

As a result, any IBM-PC programs which make their own version of the keyboard driver, under the assumption that they receive original IBM Scan Codes from the keyboard (for instance the Microsoft Flight Simulator), will run on the Butterfly keyboard.

2.3.3 ND-NOTIS compatibility

The following features may be noted:

- Two NOTIS keys - LOCK and LOCAL - do not exist in their original form. However LOCAL is substituted by SYS, which will serve the same functions plus a few more.
- Four new IBM keys have been introduced:



- There are three other new keys, as yet unassigned. These are mainly for NORTEXT typesetting systems, and future graphics features in NOTIS.
- One key function has been redefined: SHIFT PRINT now means "Print Screen" instead of "Activate Formatter".

2.3.4 Added features

2.3.4.1 Mouse

In the middle of the rear edge of the keyboard is a "mouse hole", for the connection of an industry-standard Mouse. The mouse connection is a 9-pin D-type connector.

Fig 2.11 gives a pictorial representation of mouse functions.

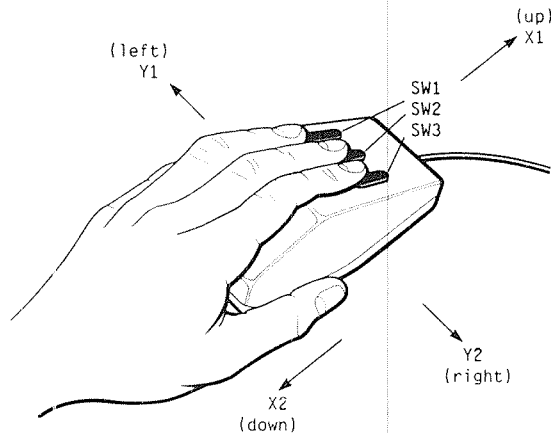


Fig 2.11: Mouse pushbuttons and movement

The Mouse transmits four position signals (X1, X2, Y1 and Y2), where X direction is UP/DOWN and Y direction is LEFT/RIGHT.

- X1/Y1 indicates movement up/left.
- X2/Y2 indicates movement down/right.

It has three pushbuttons SW1, SW2 and SW3 - which are operated by the INDEX, MIDDLE and RING finger, respectively. For the benefit of left-handed people, it is possible to swap the meanings of SW1 and SW3, so that the index finger operates the most used button.

2.3.4.2 Bar Code Reader port

The serial input (7-bit ASCII code - the eighth bit is always zero) is converted into equivalent key strokes (IBM scan codes), from the keyboard.

The number of possible codes that can be received from Bar Code Readers is at present around 40-50. However, the conversion table is prepared with 128 entries, so that Norsk Data can make modifications in the future.

The Header keystroke is Ctrl @ (=NUM).

The termination keystroke is CR (return).

A serial device may receive control information via this interface, through the keyboard. The 7-bit ASCII code may come with the command E8 hex (Serial Output Port Data). These codes are passed on by the keyboard on the Serial Transmit Line, without any conversion.

2.3.4.3 Function keypad option

Facility is provided for connecting a separate Function Keypad to the keyboard.

This option is not yet available.

The provision of this connector on the keyboard enables future addition of facilities that are compatible with an IBM function pad or a numeric pad. The function keypad will contain ten function keys which work in parallel with the existing function keys on the keyboard. One key on the keypad will be reserved as a prefix for command input from the pad.

2.3.4.4 Template cards

Above the Push-keys, guide slots are fitted to accept a template card (fig 2.12), which the User may then mark to indicate what is programmed in each Push-key. There are 24 Push-keys (P1-P8, in unshift, shift, and control). These may be set up to contain the User's own defined key sequences, or pre-programmed key values for special application programs.

CTRL										INDEX	MIDDLE	RING
SHIFT												
BASE												

Fig 2.12: Push-keys template

Above the Function Keys are a further set of guides, for accepting a function keypad template card (fig 2.13). A repertoire of cards may then be marked by the User to indicate the functions assigned to each key in the Function Keypad, for different programs.

Fig 2.13: Function Keypad template

CHAPTER 3 FUNCTIONAL DESCRIPTION - SOFTWARE

TABLE OF CONTENTS

Section	Page
3.1 INTRODUCTION	3-5
3.2 INSTRUCTION SET	3-7
3.3 SOFTWARE OVERVIEW	3-8
3.3.1 General	3-8
3.3.2 Signal box	3-8
3.3.3 Interrupts	3-10
3.3.4 Mailbox	3-11
3.3.5 IOX simulators	3-13
3.3.6 PC-NDI/O	3-14
3.3.7 System manager	3-14
3.3.8 Keyboard handler	3-15
3.3.9 BIOS sharing module	3-15
3.3.10 File access	3-16
3.3.11 Print spooler	3-19
3.3.12 PC server	3-19
3.3.13 Software memory map	3-20
3.4 DATA HANDLING	3-21
3.5 MAILBOXES	3-22
3.5.1 Mailbox format	3-22
3.5.2 Disk mailbox data buffer	3-22
3.5.3 Async mailbox data buffer	3-22
3.5.4 Butterfly DMA data buffer	3-23
3.6 SENDING A MAILBOX	3-24
3.6.1 Mailbox queues	3-24
3.6.1 Send mailbox to PC (NDSOURCE)	3-24
3.6.2 Send mailbox to ND (PCSOURCE)	3-25
3.7 ND SIDE OF THE MAILBOX	3-26
3.7.1 Butterfly interrupt handling	3-26
3.7.2 IDENT handling	3-30
3.7.3 IOX(T) handling	3-31

3.8	PC SIDE OF THE MAILBOX	3-32
3.8.1	Interrupt handling	3-32
3.8.2	Disk handling	3-34
3.8.3	Asynchronous (serial port) handling	3-34
3.8.4	DMA device handling	3-35
3.9	FILE ACCESS	3-36
3.9.1	PC trap (filter)	3-36
3.9.2	Inter-CPU buffer messaging	3-36
3.9.3	Fileserver (PCFSERV)	3-37
3.9.4	Butterfly device drivers / buffers	3-37
3.10	SYSTEM MANAGER AND KEYBOARD HANDLER	3-38

LIST OF ILLUSTRATIONS

Figure		Page
3.1	CPU microprogram control	3-5
3.2	Mailbox communication	3-6
3.3	Butterfly-110 software structure	3-9
3.4	Butterfly mailbox "devices"	3-12
3.5	IOX simulation programs for level 16	3-13
3.6	PC-NDI/O control of mailboxes	3-14
3.7	File access	3-16
3.8	Software memory map	3-20
3.9	ND additions to PC software	3-33

3.1 INTRODUCTION

The Butterfly-110's ND-110PCX computer is derived from the ND-110CX, which in turn is developed from the ND-100. The ND-110 system architecture and functionality is retained, so that much of the explanation in the ND-110 Functional Description manual (ND-06.026) applies to the ND-110PCX.

In particular, the following sections in the ND-110 Functional Description manual apply:

CPU

- Chapter 2, Central Processor Unit (CPU):

The ND-110PCX CPU provides all the functions that are offered in the ND-110 (RASK), using the same 64-bit microinstruction format to implement the same repertoire of machine (macro-) instructions. It uses the same three gate arrays (MIC, MAC, ALU), in the same configuration, as illustrated in fig 3.1.

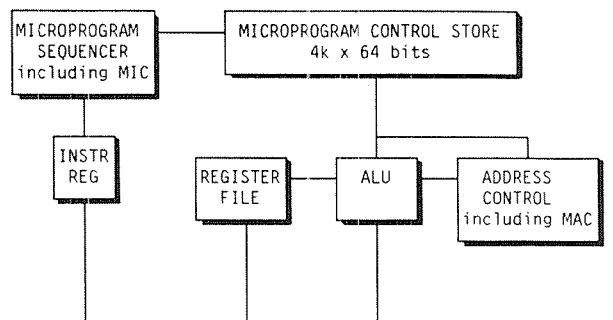


Fig 3.1: CPU microprogram control

ND-110PCX does not however perform instruction prefetch operations. Neither does it have any cache memory.

Interrupts

- Chapter 2, Interrupt System:

The ND-110PCX "program levels" structure conforms with that of the ND-110.

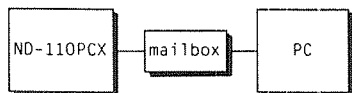


Fig 3.2: Mailbox communication

However, an additional program level - microcode level 16 - is used by the programs which communicate between the ND-110PCX and PC processors. These programs pass tasks and responses to each other through "mailboxes", which are located in the ND shared memory area (fig 3.2).

The ND-110PCX "virtual" program level 16 (which operates within the microprogram itself) is used for communication with mailbox. This calls the "IOX simulator" programs, which perform the required operations as if the mailboxes are input/output devices. For this purpose, there are four main routines in the IOX simulators:

- level-16 interrupt handler, for mailbox interrupts
- IDENT instruction simulator,
- IOX(T) simulator,
- an interrupt Timer, to optimise the number of actual interrupts between the PC and the ND.

MMS

- Chapter 3, Memory Management System (MMS):

ND-110PCX incorporates the same paging and ring protection systems of the ND-110. The paging system is in "extended mode", i.e. 24 address bits, giving physical address capability up to 16Mwords. The MMS emulates that of the ND-110CX but without using cache memory.

Memory

- Chapter 5, Memory:

Memory organisation in ND-110PCX is designed to meet the specific requirements of Butterfly-110. Only the general memory concepts described in ND-110 Chapter 5, and the PEA/PES registers, apply.

I/O system

- Chapter 6, Input/Output System:

The general system descriptions for programmed I/O and DMA operations apply. The ND-110PCX has a specific set of requirements to perform in Butterfly-110 (see Chapter 2, section 2.2.10), and these are described more fully in Chapter 4 of this manual.

3.2 INSTRUCTION SET

Reference The ND-110 Instruction Set manual (ND-06.029) description of the instruction types, word formats and instruction set, applies also to the ND-110PCX.

Overview The ND-110PCX has the same software functionality as the ND-110CX. It provides a comprehensive instruction set, which includes operations on bits, bytes, single words (16 bits), double words, triple words, BCD, floating point, arithmetic and logical operations, and system control functions.

All ND-100-series machines are controlled by a microprogram. In ND-110PCX, this microprogram is stored in a 4kx64bit read-only-memory - the Control Store.

One microprogram instruction is processed (fetched and executed) over a variable period of time. This time is in multiples of "nano-cycles". These are optimised for each machine (macro-) instruction, to suit the activities that the CPU must perform to fetch and execute the instruction. In ND-100PCX, a nano-cycle is 25nsec long. Thus a number of different microcycle periods are possible, depending on which microinstruction is being executed, and on external events. This period may vary, from what is termed "short" (275ns), to "long" (425ns). Microcycle timing is described in greater detail in Chapter 4.

New Instruction "Return from Simulation" (RTNSIM, but also known in the IOX simulator software as SECRE) - op code 150403 - is a new instruction, added to the microcode for ND-110PCX.

RTNSIM

It is the level 16 version of the WAIT instruction.

Level 16 is described in section 3.4. The detailed execution of the RTNSIM instruction is described in Chapter 5.

3.3 SOFTWARE OVERVIEW

3.3.1 General

From the system control viewpoint, Butterfly-110 consists of an IBM-compatible PC computer running MS-DOS, co-existing with an ND-110PCX computer running SINTRAN. The latter is a modified ND-110CX processor, with 1 or 2 Mbyte memory, direct I/O ports for HDLC and Console/Test (UART), and an interface to the PC.

The PC controls the major peripheral I/O devices:

- floppy disk
- hard disk
- Butterfly terminal, comprising the display, keyboard and mouse
- local terminal devices, connected via RS232 interfaces to the PC's expansion bus
- devices connected to the PC's standard serial port (used in Butterfly-110 for ND's COSMOS asynchronous communications facility) and parallel port (available for connecting a local printer).

Fig 3.3 shows Butterfly-110's main operating program modules, their control relationships, and their functional connections with the main peripheral I/O devices.

3.3.2 Signal box

The ND must pass tasks to the PC, in order to use the devices connected to the PC (Butterfly display & keyboard, floppy disk, hard disk, local terminal devices, etc.), and to share access to data. Similarly, the PC must make requests to the ND, to enable the PC to give tasks to the ND-110PCX, and to share access to data.

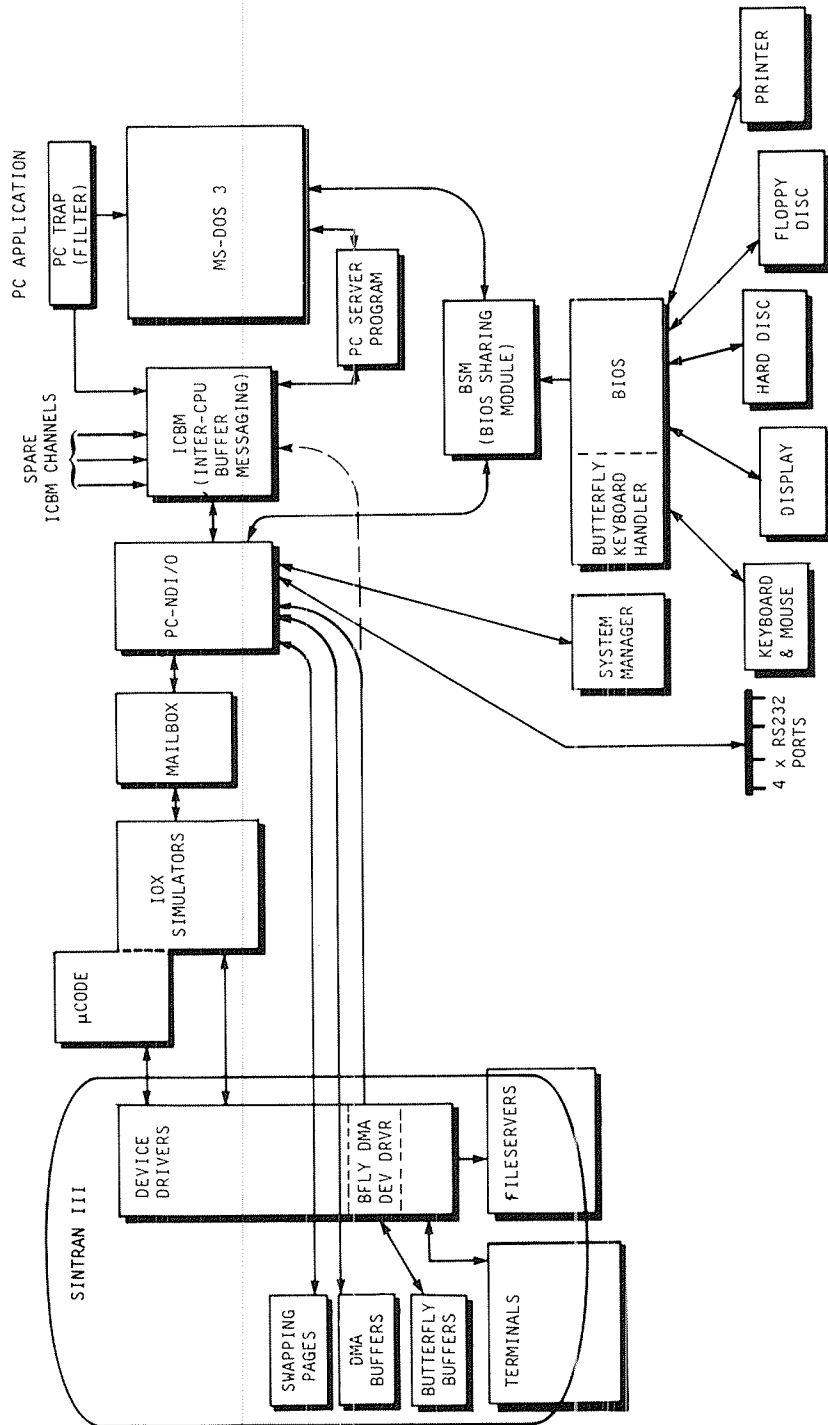


Fig 3.3: Butterfly-110 Software Structure

In Butterfly-110, the lowest level of activity for achieving this communication is provided by "signal boxes". There is a signal box for each processor:

- for the ND side, it is in microcode
- for the PC side, it is in PC-NDI/O.

This is described further in sections 3.7 for the ND, and 3.8 for the PC.

3.3.3 Interrupts

In Butterfly-110, there are two interrupt lines between the ND and PC, one in each direction.

- ND → PC
This uses the IRQ5 line to the PC - see Chapter 6 section 6.6.
- PC → ND
This is decoded in the ND-110PCX as signal PSPTON to its Interrupt Detect circuitry - see Chapter 4 section 4.3.6.

For the purposes of communicating between the ND and PC sides of the machine, there are four interrupt sources in each direction:

- mailbox input/output - see Mailbox, below.
- terminal-1 output - this is the "ND console" facility, whereby in Butterfly-110 it is possible to use the Butterfly terminal (display & keyboard) as the "terminal-1" device.
- terminal-1 input - as for terminal-1 output, but in the opposite direction.
- calendar input/output. The PC's battery-backed calendar is used as the Butterfly-110 system calendar, and can therefore be set up (output) and read (input) from the ND.

These interrupts are described further in sections 3.7.1 and 3.8.1.

3.3.4 Mailbox

Mailboxes allow more complex tasks to be passed between the ND and PC than is possible to communicate through the signal box mechanism. Such tasks divide into three types:

- block data transfers, such as are required for floppy and hard disks
- asynchronous data transfers, for byte-oriented devices such as terminals
- DMA data transfers, using 2kword memory areas which in the ND are seen as "Butterfly devices".

These types of transfer account for most of the traffic across the PC-ND interface.

For terminal devices, a mailbox is assigned for each device in the PC that the ND wishes to be able to communicate with, in each direction of transfer (output and input). Floppy disk and hard disk have one mailbox only.

The current list of mailbox devices in Butterfly-110 is shown in fig 3.4. More detailed information, giving device numbers and ND/PC identity numbers, is included in the Mailbox description in Chapter 6.

Mailbox	Use
floppy disk 1	DOS drive A
hard disk 1	DOS drive C
floppy disk 2	DOS drive B (not yet implemented)
hard disk 2	DOS drive D (not yet implemented)
terminal 1 O/P	ND console
terminal 5 O/P	COSMOS async port output
terminal 6 O/P	printer port output
terminal 7 O/P	emulator for SINTRAN login
terminal 8 O/P	emulator for UE login
terminal 9 O/P	} QuadPort card output
terminal 10 O/P	
terminal 11 O/P	
terminal 12 O/P	
terminal 13 O/P	} provision for 2nd QuadPort card (included in ND side only)
terminal 14 O/P	
terminal 15 O/P	
terminal 16 O/P	
terminal 5 I/P	COSMOS async port input
terminal 6 I/P	printer port input
terminal 7 I/P	Bfly terminal ND-mode SINTRAN
terminal 8 I/P	Bfly terminal ND-mode UE
terminal 9 I/P	} QuadPort card input
terminal 10 I/P	
terminal 11 I/P	
terminal 12 I/P	
terminal 13 I/P	} provision for 2nd QuadPort card (included in ND side only)
terminal 14 I/P	
terminal 15 I/P	
terminal 16 I/P	
Bfly buffer 0	} ICBM channels & Bfly DMA devices
Bfly buffer 1	
Bfly buffer 2	
Bfly buffer 3	
Bfly buffer 4	

- Notes: 1 terminal-1 = ND console
terminals 2,3,4 do not exist in Butterfly-110.
- 2 The ND (IOX simulator) has terminals 13-16 defined, which could be used to handle a second QuadPort card. No corresponding handlers currently exist in the PC (PC-NDI/O).

Fig 3.4: Butterfly mailbox "devices"

Fig 3.4: Butterfly mailbox "devices"

3.3.5 IOX Simulators

Standard IOXs to mailbox devices

When the ND is required to perform an I/O operation with a mailbox device, the ND-110PCX enters a special area of Butterfly microcode, which in turn makes calls to a set of Mailbox Handler programs held in the ND's secret memory. These programs are referred to as the "IOX simulator" code. They perform the operations involved in communicating between the ND and a mailbox.

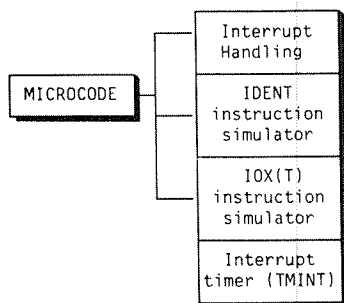


Fig 3.5: IOX simulation programs for level 16

These operations involve (fig 3.5):

- handling mailbox interrupts (PC to ND, and ND to PC). This is performed by a combination of microcode and macrocode
- simulating the IDENT instruction, to service the "mailbox devices"
- simulating the IOX(T) instructions for each of the mailbox devices.
- using a Timer to optimise the number of interrupts between the ND and the PC. This ensures that the interrupt handling overhead is minimised for asynchronous (byte) transfers.

These simulators are transparent to the rest of the microcode, which sees an interrupt, IDENT or IOX/T operation with mailbox like any other. To achieve this, the mailbox IOX simulators run at a special program level 16, which is internal to the Butterfly microcode.

Special IOXs to async mailboxes

The timer (TMINT) significantly reduces the overhead involved in running the level-16 interrupt handler and IDENT instruction simulator. However, for traffic between the ND and mailbox, this is decreased still further by modifications to SINTRAN which allow the ND to read and write asynchronous data transfers directly from or to a mailbox, bypassing the IOX simulators. This is described further in Chapter 6.

3.3.6 PC-NDI/O

The PC-NDI/O module runs under the PC's MS-DOS operating system. It performs an equivalent task to the ND's IOX simulator code, for the PC side of Butterfly.

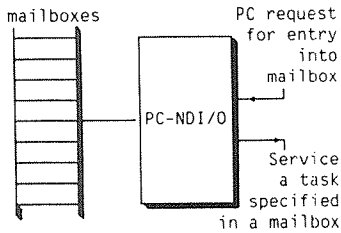


Fig 3.6: PC-NDI/O control of mailboxes

- It translates PC requests into the required form, for entry into requisite device mailboxes, the PC then interrupting the ND.
- In the other direction, it services requests and responses put into the mailboxes by the ND, translating these into appropriate activities within the PC side.

In this way, the workings of other programs in the PC side have no direct effect on the way that the PC and ND sides of the machine communicate.

- It handles the requirement to share peripheral devices (disks, parallel port) between DOS and the ND.
- It handles terminal-1 and calendar data transfers.

3.3.7 System manager

The System Manager program supervises orderly context switching between the ND and PC modes of operation of Butterfly-110's keyboard and screen. This involves saving context-specific parameters and operating conditions, before restoring those for the newly selected mode.

3.3.8 Keyboard handler

At any one time, Butterfly may operate in ND-mode or in PC-mode. These two environments share a single keyboard and display.

The Butterfly keyboard combines the requirements of both the ND and the PC, in one design. In addition, it includes new features which provide enhanced performance and functionality over standard ND-NOTIS and PC keyboards.

Because of the different functional requirements of a PC keyboard and an ND keyboard, a Butterfly Keyboard Handler replaces the PC's standard BIOS keyboard handler.

3.3.9 BIOS sharing module

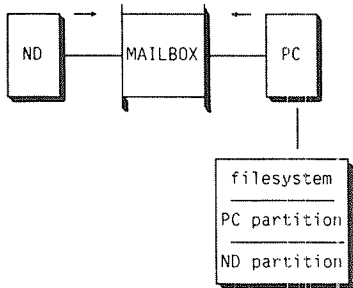
Fig 3.3 indicates that requests for BIOS service originate from two sources:

- from MS-DOS, in response to a PC user program
- from the ND, via PC-NDI/O.

The activity to share BIOS between these two sources is referred to as "multitasking".

In the context of Butterfly-110, both DOS and PC-NDI/O have concurrent access to the disks, PC display, and parallel printer port. The BIOS Sharing Module (BSM) mediates between the two sources of request, to share these resources. It also works with PC-NDI/O to take control of the access to the BIOS routines for these devices.

3.3.10 File access



File access (fig 3.7) is provided so that a PC application may access ND files without modifying that application. This facility is only available to PC applications which limit their DOS file function calls to the subset that the PC Trap module currently recognises.

In addition, Butterfly-110 includes an application program - DUPLI - which uses this subset for copying ND and PC files, since the DOS COPY command uses function calls that are not currently trapped.

Fig 3.7: File access

PC programs to
access ND files

Any DOS application which uses file commands outside the current subset that is trapped (by the PC Trap module - fig 3.3), will not succeed. In these instances, the PC-user must use the DUPLI program (see below) to access the required file.

PC programmers may write applications using the subset of file commands that are currently trapped by the PC Trap module. This will enable a PC-user to access any ND (SINTRAN) file, without needing any special knowledge of the ND filesystem except for following the ND-filenaming conventions.

By preceding the ND filename with the characters

ND:

a PC file request for an ND file is trapped by the PC trap module. The current subset of DOS INT21h function calls that are trapped include open a file, close a file, read a file, write a file, and create a file. Chapter 6 includes a detailed list of these function calls. These thus provide the basic facilities for passing files between the PC and ND, in both directions.

The mechanism for handling this file access involves several areas of software in Butterfly-110 (see fig 3.3):

- a resident PC Trap program in the PC traps PC-user requests for access to ND files, and passes these requests to the Inter-CPU Buffer Messaging (ICBM) module.
- The ICBM sets up information defining the required file access function to a specified Butterfly Buffer in the ND, then sends a request for service through a (DMA) mailbox.
- This interrupts the ND, and the resulting IOX simulator operation activates the Butterfly Device Drivers.
- These in turn call the ND's PC-Fileserver (an RT program called PCFSERV), which performs the file access function required. The information involved in the transfer is entered into the appropriate Butterfly Buffer, and the Butterfly Device Drivers then update the (same) mailbox to advise that the required information is in the requisite Butterfly Buffer.
- The resulting ND-to-PC interrupt causes PC-NDI/O to set a "ND file access ready" flag.
- The ICBM responds by copying the data from the requisite Butterfly Buffer, to the corresponding PC buffer.
- Then, the ICBM hands this on to the PC Trap program, which filters the file handle which has been returned from the ND filesystem, then in turn passes control to the DOS application.

In this way, the requesting PC-user application is unaware of which filesystem (ND or PC) is being used, and has access to all files in Butterfly-110 - or anywhere on the COSMOS network.

The software modules are described individually in Chapter 6.

DUPLI program

The DUPLI (duplicate file) program is a resident Butterfly-110 program, running under MS-DOS. It enables a PC-user to copy complete files:

- 1 to create a PC-file copy of an existing PC file.
This would normally be done using MS-DOS' COPY command.
- 2 to create an ND-file copy of an existing ND file.
This would normally be done by an ND-user, using the SINTRAN COPY-FILE command.

- 3 to create a PC-file copy of an existing ND file.
This accesses an ND file (e.g. filename JACK) and creates a copy of it in the PC filesystem, with whatever destination filename is specified (e.g. JILL):

DUPLI ND:JACK,TO,JILL *

- 4 to create an ND-file copy of an existing PC file.
This accesses a PC file (e.g. filename FRED) and creates a copy of it in the ND filesystem, with whatever destination filename is specified (e.g. BILL):

DUPLI FRED,TO,ND:BILL *

* commas between parameters may be spaces.

Items 3 and 4 are the major uses of the DUPLI program.

The formats of the commands available are:

- DUPLI filename,TO,filename
This format complies with the MS-DOS command convention of SOURCE / DESTINATION.
- DUPLI filename,FROM,filename
This format complies with the SINTRAN command convention of DESTINATION / SOURCE.
- DUPLI
If only the command name (with no further parameters) is entered, the command sequence prompts the user for the parameters it requires.

The filenames will be either:

- MS-DOS filenames, with or without a pathname (directory-filename)
- SINTRAN filenames, preceded by the characters ND:

The full string of parameters involved is then

ND:machine(user(password)).(directory:user)filename:type;version

plus the -7 -8 -9 extension to the file type for odd even or no parity data transfer - see Chapter 6 section 6.9.1.1 (File data formats), although in the same machine this may be abbreviated to

ND:(user(password)).filename:type;version

Further details are provided in Chapter 6.

3.3.11 Print spooler

One printing task must be allowed to complete before allowing another printing task to gain access to the printer.

A printing task for the ND is spooled in the normal way, and then handled through the IOX simulators and mailbox, to interrupt the PC and request print service. PC-NDI/O receives this interrupt, and checks to see if MS-DOS is using the printer. If not, PC-NDI/O takes control, and the complete job is printed. If during this time, MS-DOS wishes to use the printer, it is obliged to wait until PC-NDI/O releases its control of the printer.

In a similar way, if MS-DOS is using the printer, PC-NDI/O must wait until MS-DOS releases it before it can proceed with the ND task.

3.3.12 PC server

Provision has been made to include a PC-Server program in the PC side of Butterfly, to mirror the function provided by the ND Fileserver (PCFSERV) program.

PCFSERV allows a Butterfly-110 user to access the ND filesystem while operating in PC-mode (i.e. running a PC application). The purpose of a PC-Server would be to provide the same facility to the ND, so that the PC filesystem is available to a user while operating in ND-mode.

Currently, a PC-Server program is not included in the Butterfly-110, so an ND-user who wishes to access the PC filesystem must first change environments to become a PC-user, then write the required PC file(s) to the ND filesystem, then revert to being an ND-user. Later, to update those files in the PC filesystem, the user must repeat this process, in the opposite direction.

3.3.13 Software memory map

The various software components shown in fig 3.3 reside in either ND memory or PC memory. Fig 3.8 indicates their location.

See also the Memory functional description in Chapter 2 (section 2.2.8), and the detailed Memory hardware description in Chapter 4 (section 4.3.5).

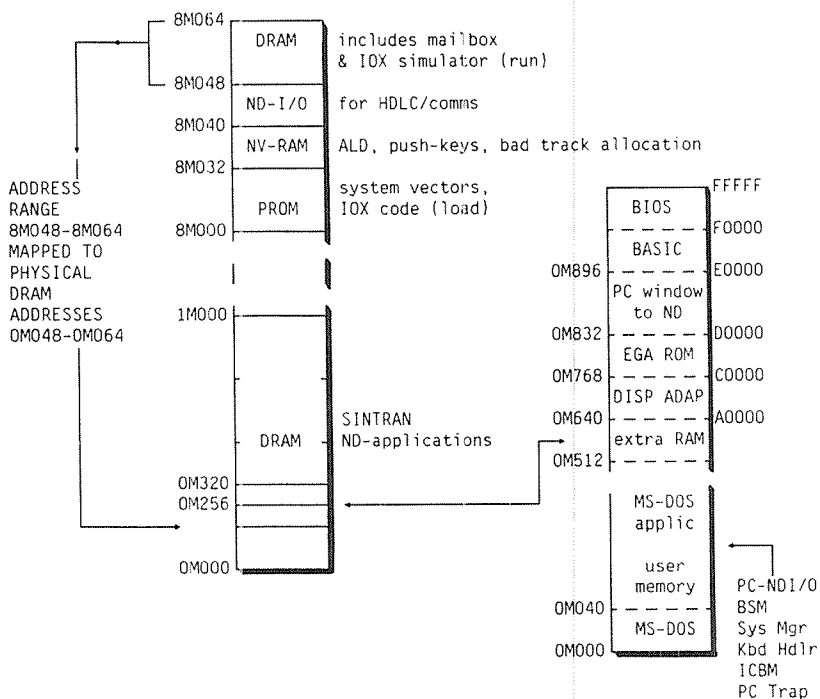


Fig. 3.8: Software Memory Map

3.4 DATA HANDLING

There are three broad categories of data that are transferred between the ND and PC:

- standard peripheral I/O. This is achieved by control operations through the mailbox area. A functional description of how this is handled is given in sections 3.5.2 - 3.5.4.
- console communication (OPCOM), to enable the Butterfly terminal itself to be used as the ND console (terminal-1). This is effected through a 4-word reserved area of ND secret memory, called "st-mail".
- panel and other processor status information (e.g. active level, interrupt on, paging on, PCR information), again using the Butterfly as the terminal. This information is also passed through "st-mail".

"st-mail" is also used for diagnostic program purposes - st = self-test, see Chapter 7.

The standard use of ND-Telefix is to attach a modem to the ND console interface. In Butterfly, this can be done via a PC plug-in modem in conjunction with a PC service program, or directly via a Telefix modem through the ND's console UART (so bypassing the PC).

3.5 MAILBOXES

Mailboxes operate as I/O control interfaces, for the ND and PC to use as if they are peripheral I/O devices. There is one mailbox per "terminal" device, for each direction of data transfer (ND to PC, PC to ND), and one each for floppy and hard disks.

A mailbox is an area of ND secret memory. It contains information which describes the type of mailbox and the input or output operation required.

3.5.1 Mailbox format

The structure of the three basic types of mailbox identified in section 3.3.4:

- block devices (disks)
- asynchronous devices (serial ports)
- Butterfly DMA devices

and their use by the IOX simulators and by PC-NDI/O, is described in Chapter 6.

3.5.2 Disk mailbox data buffer

Disk accesses to the ND filesystem are initiated by the ND sending a disk-type mailbox to the PC. The information in this mailbox includes two words (MADHI and MADLO) which specify a 24-bit address in ND shared memory, for use in the data transfer with disk. SINTRAN assigns an appropriate number of "pages" (1kword blocks) of memory, to accommodate the number of words (WDCNT value, also specified in the mailbox) it wants to be transferred.

3.5.3 Asynchronous mailbox data buffer

FIFO data buffer The data buffer area for async mailboxes is in the form of a FIFO store. The sending side of the async mailbox PUTS data byte-by-byte into the FIFO, and the receiving side GETs data from the FIFO. This FIFO area forms part of the memory area

used by each mailbox (see Chapter 6). Its size is set according to the buffering requirement for how many bytes a particular async mailbox should accept before the receiving side empties it.

The design of the FIFO buffer is influenced by the need to prevent both processors from writing to a location at the same time. The processor (ND or PC) which puts a byte into it modifies the PUT pointer, and the processor taking a byte out modifies the GET pointer. The number of bytes in the buffer is calculated by subtracting the GET pointer from the PUT pointer, adjusted to allow for wraparound. Async mailboxes include the PUT and GET values within their structure (see Chapter 6).

The FIFO is defined to be empty when the GET and PUT pointers are equal, and is full when the PUT pointer is one less than the GET pointer.

The MAX BUFF LEN parameter in the mailbox gives the maximum number of characters the FIFO can contain, which for the reason given above is one less than the actual size of the buffer.

The FIFO data buffer is an array of 16-bit words with one byte per word, in the lower 8 bits. The GET and PUT pointers are word offsets into the buffer, i.e. they are numbers in the range 0 to "MAX BUFF LEN".

ND → PC
byte transfer

The IOX simulators initiate output of a byte by sending a mailbox to PC-NDI/O. The transmit interrupt for the port is enabled (port is ready to accept characters), and characters are taken from the mailbox by PC-NDI/O's interrupt handler. When the mailbox is empty the interrupt handler turns transmit interrupts off again.

PC → ND
byte transfer

Characters can always be received from a serial port because the receive interrupt is never disabled. The PC-NDI/O interrupt handler puts characters in the mailbox, and interrupts the ND if the mailbox was empty; otherwise the ND is already servicing that mailbox (see Interrupt timer, Chapter 6) and so does not require a further interrupt.

3.5.4 Butterfly DMA data buffer

The use of Butterfly buffers in File Access operations is outlined in section 3.3.10, and is described further in section 3.9.4.

3.6 SENDING A MAILBOX

3.6.1 Mailbox queues

The mechanism used by PC-NDI/O to send a mailbox to the ND is to put its address into a FIFO queue (called PCSOURCE), and if that queue was previously empty, to send an interrupt request to the ND.

In a similar way, the IOX simulators send a mailbox to the PC by putting the mailbox's address into a PC mailbox FIFO queue (called NDSOURCE), and interrupting the PC.

The structure that is used to contain these mailbox FIFOs is that of an asynchronous mailbox. Pointers to these two FIFOs are held in fixed locations in secret memory:

ND address	PC offset address	Name	Use
140030	8030h	PCSOURCE	Pointer to PC→ND mbx queue
140032	8032h	NDSOURCE	Pointer to ND→PC mbx queue

3.6.2 Send mailbox to PC (NDSOURCE)

When the ND requires an operation via a mailbox device, it first sets up the required mailbox with information specifying that operation. Then it enters the address of that mailbox into a mailbox queue to the PC (NDSOURCE). This is a FIFO buffer, into which the ND places every mailbox it wishes to send to the PC. Finally it sets the ND-to-PC signal box ATTEN1 flag, and prepares to interrupt the PC (via IRQ5):

- If it is NOT an async mailbox, the ND-to-PC interrupt is generated directly.
- If it IS an async mailbox, the program is directed to a timer table which lists those mailboxes which may use the level-16 interrupt timer (TMINT, see section 3.3.5). If the timer is required, it runs. In the meantime, the ND may enter more bytes into this async mailbox's FIFO (using IOX x04, see Chapter 6). Then, when the timer times-out it generates the ND-to-PC interrupt.

The PC responds to the interrupt, and as it is IRQ5, directs action to PC-NDI/O. This in turn checks its signal box, sees the ATTEN1 flag set, so looks in NDSOURCE, finds the address of the mailbox requiring service, clears the signal box flag, and calls the appropriate routine to service this mailbox.

Thereafter, the PC-NDI/O mailbox handling routine loops, getting each mailbox from the (NDSOURCE) FIFO and processing it, until this FIFO is empty (i.e. there are no more mailboxes waiting to be processed).

By queuing the mailboxes to the PC in this way, the number of ND-to-PC mailbox interrupts declines as the system becomes busier.

3.6.3 Send mailbox to ND (PCSOURCE)

A similar sequence of operations takes place for handling mailboxes sent from the PC to the ND. Having set up the mailbox, it enters the address of the mailbox into the mailbox queue from PC (PCSOURCE FIFO buffer). Then, it sets the PC-to-ND signal box ATTEN1 flag, and sends an interrupt to the ND (via PC I/O to port 166h, to set the Keys register bit 1 - PSPTON).

The PC-to-ND interrupt initiates the level-16 interrupt handler in the ND. This checks to determine if the interrupt is for an async mailbox: if so, it starts the interrupt timer for that mailbox. While the timer is running, the PC may enter more bytes into that mailbox's FIFO, but since the ND has already been interrupted, it is cued to service the mailbox and so does not need to be given a further interrupt.

When either the timer times-out, or a minimum byte count in the FIFO is reached, the ND level-16 interrupt handler is allowed to proceed. In this way, the system pauses to allow a number of bytes to accumulate before interrupting the ND (rather than generate an interrupt for every byte), but the timer ensures that the ND always responds to a mailbox within a maximum time from the first byte being entered into that mailbox.

The level-16 interrupt handler in turn processes its signal box, sees the ATTEN1 flag set, so looks in PCSOURCE, finds the address of the mailbox requiring service, clears the signal box flag, and calls the appropriate routine to service this mailbox.

3.7 ND SIDE OF THE MAILBOX

3.7.1 Interrupt handling

Butterfly-110 handling of mailbox I/O operations is structured to comply with the standard ND-110 mechanism for handling I/O devices. It involves three basic stages:

- Interrupt handling
- IDENT
- IOX(T).

However, due to the complexity of the operations involved, a lower level of activity, at the microcode rather than macroprogram level, is required. The main function of this activity is to simulate the hardware of the standard ND-110 interrupt system, using microcode, or macrocode routines called by microcode.

Signal Box

For interrupts between the ND and the PC, there is only one interrupt line in each direction (ND-to-PC and PC-to-ND). However, there are four PC interrupt sources (mailbox, terminal-1 output, terminal-1 input, and calendar). The signal box provides the mechanism for multiplexing these four interrupts onto the one line.

The signal box comprises a set of four locations, in secret memory area. These signal the interrupt.

- A calling routine sets the appropriate signal box flag and sets up the interrupt.
- The other processor receives the interrupt and uses the signal box to read the flags.
- When it finds a set flag, it resets it and jumps to the higher level interrupt processing routine.
- When that routine finishes, it may request a return interrupt, but always passes control back to the signal box, which then checks for more interrupts pending.

The signal box is used by the level-16 interrupt handler, which simulates the hardwired ND-bus interrupt system. It runs in microcode.

An equivalent signal box function is implemented for the PC side of the mailbox, in PC-NDI/O (see section 3.8.1).

The signal box is described further in Chapter 6.

Other interrupts
to the ND

Other sources of direct interrupts to the ND are:

- the ND console (test) port
- internal interrupts 5,8,9 and 10, plus others internal to the microcode (see Chapter 6)
- the real-time clock
- HDLC
- panel interrupts: STOP and MCL (master clear).

The panel interrupts are handled in the same way as in ND-110CX, but the others are implemented at a microcode level, in the same way as the mailbox interrupt but without the change to level 16. If appropriate, a PID bit may be set, which will cause the level change check routine to be called.

Level 16

If the signal box issues a PC mailbox request to the ND, the microcode initiates the interrupt handler code on level 16. The microcode performs a pseudo-level shift, and sets the level-16 P register to point at the start of the mailbox interrupt macrocode. Note that level 16 is not a program level in the conventional way, and does not have a PID or PIE bit; neither are the working registers saved on level shifts.

There are four level-16 entry points:

- to handle a PC interrupt (PCINT)
- to invoke the interrupt Timer (TMINT)
- to handle an IDENT instruction for mailbox (IDIOX)
- to handle an IOX instruction to mailbox (IOXSWITCH)

For each one, the microcode calls the appropriate level-16 macro-coded simulation routine.

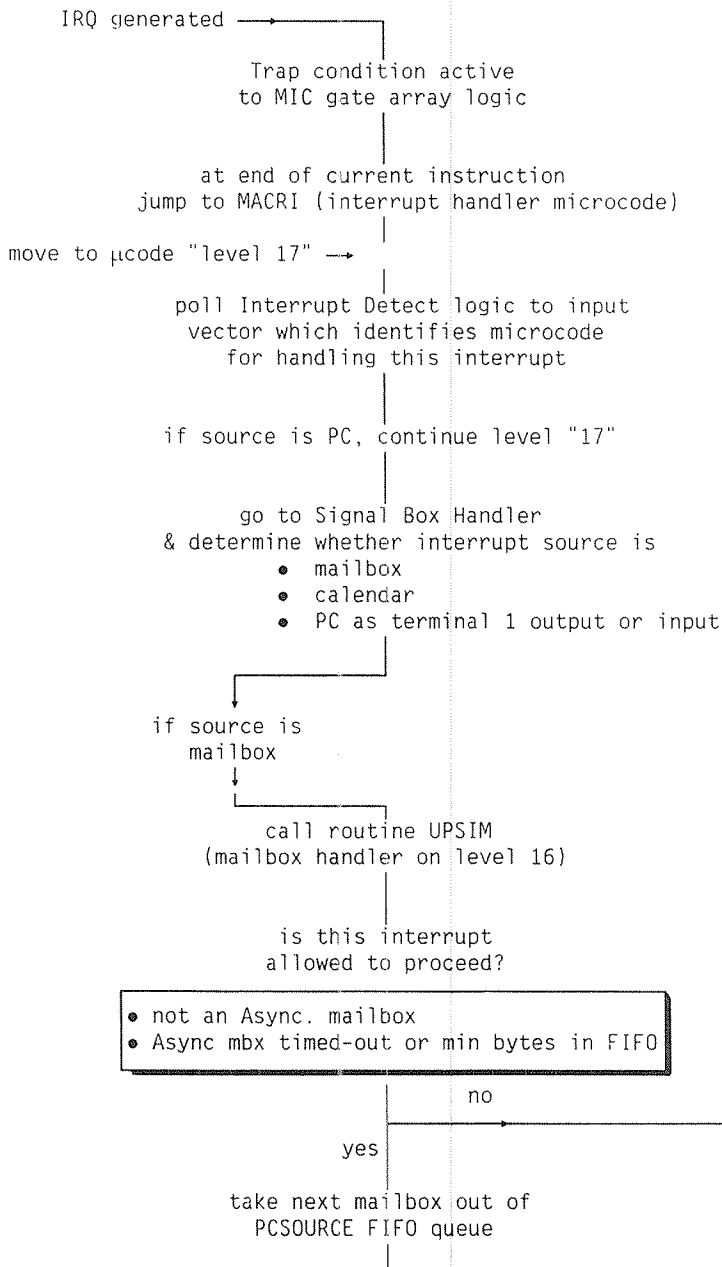
All these routines use a common (privileged) return instruction, called RTNSIM (also referred to in the IOX simulator software as SECRE), which is the Butterfly-110 level-16 equivalent to the WAIT instruction (i.e. returning control to the Interrupt Handler microcode).

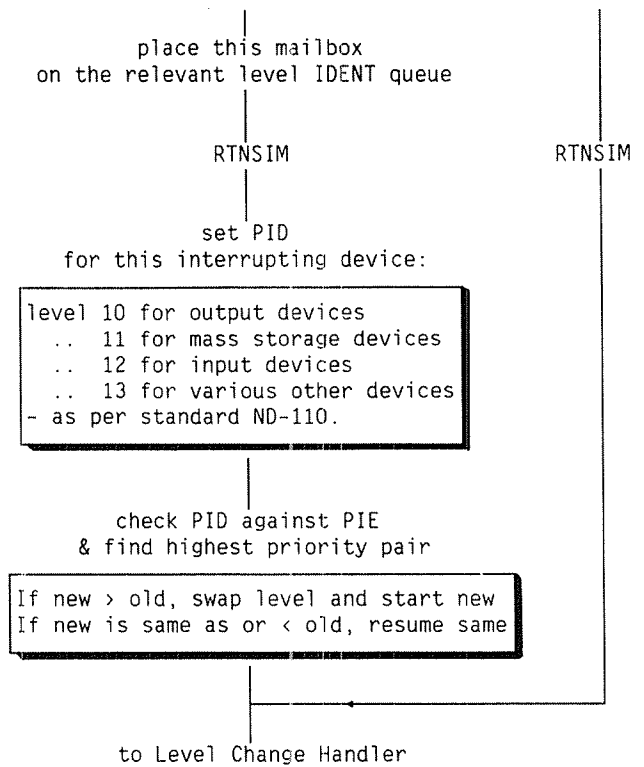
Watchdog timer

A "watchdog timer" is used in the signal box when the ND requires PC input/output. If the PC does not respond by clearing down the active flags in the signal box within a set time (1-2 seconds) of a second request from the ND, then the ND assumes that the PC has failed. In this situation, it sets its Power Fail internal interrupt, which is processed accordingly.

Summary

The sequence of operations in Butterfly-110 interrupt handling is summarised in the following flowchart:



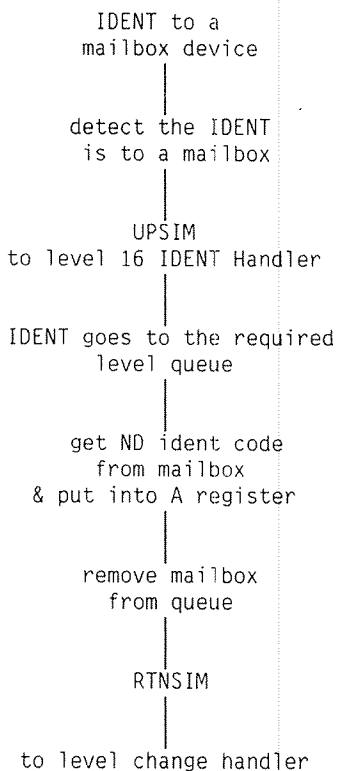


3.7.2 IDENT handling

The ND executes an IDENT instruction to service an active PID/PIE pair. This returns the ident code for the device which is to be serviced next.

The source may be "on-board" the ND, or may be the PC. If it is an "on-board" source, that source returns an ident code as normal. If it is the PC, the microcode moves to level 16 and calls the IDENT simulator (IDIOX) program. On completion, the program returns to the correct position in microcode for the normal IDENT instruction. In this way, the running of the IDENT simulator is transparent to the rest of the machine.

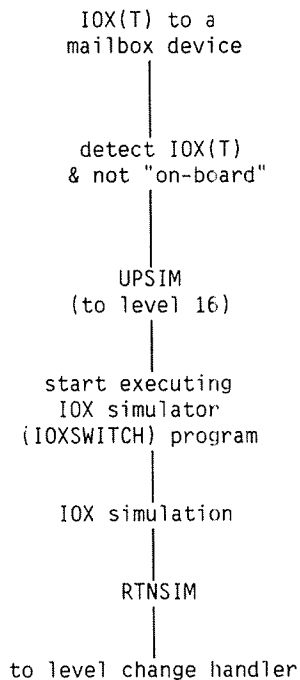
The operation sequence is summarised in the following flowchart.



3.7.3 IOX(T) handling

A similar process to that for IDENT instructions takes place for IOX(T) instructions. The microcode calls the IOX simulator (IOXSWITCH) program to service IOXs for mailbox devices.

The following sequence outlines the activities involved in executing an IOX(T) to a mailbox device.



3.8 PC SIDE OF THE MAILBOX

The main task of PC-NDI/O is to interface to the mailbox for all the programs in the PC side of Butterfly. As such it:

- emulates all those interrupt system functions described for the ND side (see section 3.7.1), including the signal box function
- controls the PC interface to the mailbox I/O "devices"
- handles the communication between the PC terminal (Butterfly display and keyboard), via the system manager and keyboard handler. This includes inhibiting ND access to the display and keyboard when the Butterfly-110 is being used in PC-mode.
- translates peripheral I/O tasks presented from mailboxes into forms which are suitable to pass to the BIOS via the BSM. It also works with the BSM to share BIOS between the ND (in the form of tasks presented via mailbox) and the PC (calls from MS-DOS)
- handles the ND "terminal-1" communication when the PC terminal is used as the ND terminal-1 device.
- handles communication of the PC's calendar information, to and from the ND.

The main software additions into the PC side of the mailbox are represented in fig 3.9.

3.8.1 Interrupt handling

As described in sections 3.3.2 and 3.3.3 (Signal box and Interrupts), communication between the PC and ND-110PCX processors is via interrupts (one in each direction) and shared memory. The lowest level of communication is the "signal box" interrupt handler in PC-NDI/O, which corresponds to the signal box code in the ND-110PCX microcode. At this level, four interrupt sources are "multiplexed" onto the one interrupt line. These are:

- mailbox input/output,
- terminal-1 output,
- terminal-1 input,
- calendar input/ output,

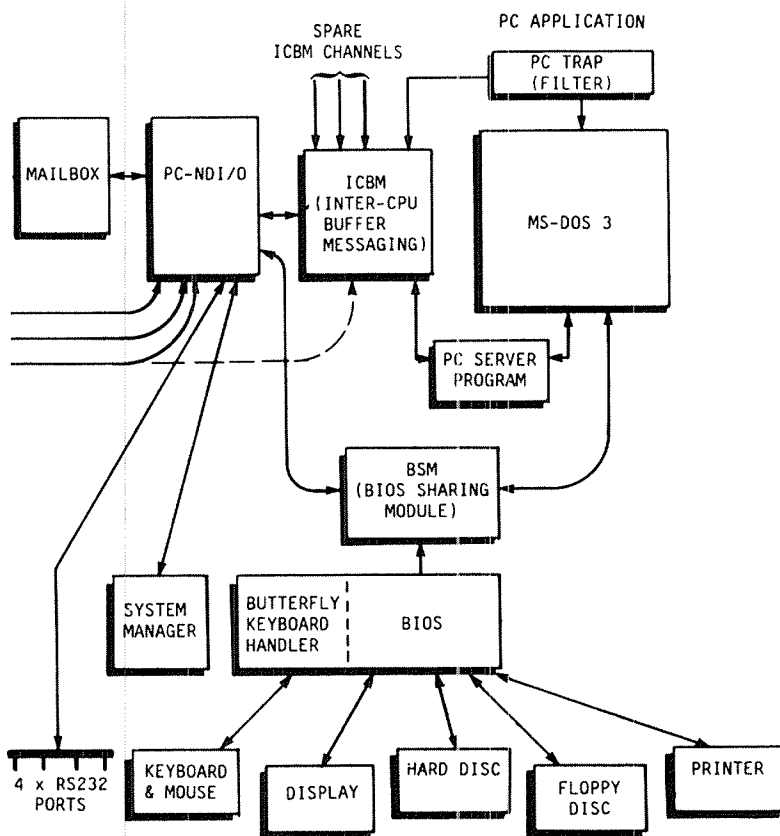


Fig 3.9: ND additions to PC software

Terminal-1 and calendar interrupts and associated data transfer operations are described in Chapter 6. Mailbox handling is outlined below, and described in more detail in Chapter 6.

3.8.2 Disk handling

To the ND, the floppy and hard disks are separate devices which can be used in parallel. On the PC, they are accessed via the same controller and therefore have to be used serially.

If one disk is being accessed when a request for the other is received, the second request is queued.

Addressing

Disk addresses from SINTRAN are translated by PC-NDI/O into the appropriate format for the floppy and hard disks which are fitted to Butterfly-110. This translation includes accounting for the partitioning of the hard disk between MS-DOS (which has the first 5Mbytes) and SINTRAN (which has the remaining 35Mbytes), by adding a cylinder address offset for addresses from the ND. It also takes into account the different geometry (number of heads and cylinders) of different types of hard disks, as defined in the PC Setup parameters (disk types E7 and E9 are currently recognised - see Chapter 7). The ND sends its normal disk addresses to mailbox, without need to take account of such considerations.

Floppy formats

PC-NDI/O works with the BSM and PC-BIOS, to handle the standard PC-format floppy disk, and the ND floppy disks type-0 and type-17. The ND type-0 is only used for checking the format of an unreadable disk.

3.8.3 Asynchronous (serial port) handling

PC-NDI/O contains drivers for serial ports, comprising the standard Ericsson serial port, and an optional Quadport with extension board. Both use the INS 8250 serial communications chip.

These ports are used as follows:

Port Name	Address	Interrupt	DOS	ND Terminal
Std. serial	3F8h	IRQ4	COM1 ¹	5
QuadPort F	288h	IRQ3	COM2 ²	-
Quadport G	298h	"	-	9
QuadPort H	2A8h	"	-	10
QuadPort I	2B8h	"	-	11
QuadPort K	2C8h	"	-	12

Notes:

- 1 The COM1 port cannot be accessed from DOS while the Butterfly-110 PC system software is installed, since it is taken for use by ND's COSMOS asynchronous communications facility.
- 2 The COM2 port is not hardware compatible with IBM-PC, since a different address is used, but it can be accessed by BIOS calls and from MS-DOS (see Chapter 6).

It may be possible to add a further QuadPort board, to provide 5 more serial ports. However, this would require significant changes to the way ND-COSMOS (on the serial port) and a local printer (on the parallel port) are currently handled.

As shown in section 3.3.4, there are two mailboxes per async port, one for input and one for output. The data transfer and interrupt control mechanism is described in Chapter 6.

3.8.4 DMA device handling

The use of mailbox in file access operations is described in section 3.3.10, and mailbox structure for this type of operation is described in Chapter 6.

3.9 FILE ACCESS

The File Access facility enables PC access to ND-format files, and ND access to PC-format files. See also section 3.3.10.

File access is provided so that a PC application may access ND files without modifying that application. This facility is only available to those PC applications which limit their DOS file function calls to the subset that the PC Trap module currently recognises.

In addition, Butterfly-110 includes an application program - DUPLI - which uses this subset for copying ND and PC files, since the DUS COPY command uses function calls that are not currently trapped.

3.9.1 PC trap (filter)

The PC trap handler is a resident program in the PC, installed on INT21h - the DOS function request interrupt.

Its task is to filter all file requests presented from a PC-user program, diverting all those for ND files (i.e. those having the filename prefix ND:) for processing by the Inter-CPU Buffer Messaging (ICBM), rather than by the normal MS-DOS INT 21h routine.

3.9.2 Inter-CPU buffer messaging

The ICBM is the program on the PC side of the machine which enables blocks of data to be transferred between the ND and the PC. The corresponding ND program is the Butterfly device drivers. Information is passed in "Butterfly buffers", which are unstructured data areas 4kbytes long. There are 5 such buffers. These are identified in the PC by specifying a channel number (from 0 to 4), and in the ND by a device number. Channels 0 and 1 are reserved for use by the Butterfly file access programs.

The basic functions are write data to a channel, and read data from a channel. Typically, the two application programs will be synchronised, each alternately writing and reading from the channel, so as to optimise the speed of transfer.

3.9.3 Fileserver (PCFSERV)

The Fileserver program calls upon the Butterfly device driver to perform whatever file access function has been specified from the PC, on the ND filesystem. When started up, the PCFSERV program calls the SINTRAN MON144 monitor call (see Chapter 6) to read a block, and then waits until a block is available. It then returns the result into the requisite Butterfly buffer, and passes control back to the Butterfly DMA driver.

There is only one DMA driver, which could start up one of several servers. Currently, only one server (PCFSERV) is needed

3.9.4 Butterfly device drivers / DMA buffers

The correspondence between Butterfly devices and matching ICBM channels in the PC is described in section 3.9.2. Five "Butterfly DMA devices" have been created for the ND side of Butterfly. Each has a datafield, with the normal ND DMA datafield structure and logical device number. The "DMA devices" have their own ND device numbers and ident codes.

Each "device" has a Butterfly Buffer (BB) and Butterfly Buffer Header (BBH). These equate to ND device buffers and device buffer headers, except that each is 2kwords long (i.e. 4kbytes) and is statically associated with the corresponding Butterfly device.

3.10 SYSTEM MANAGER AND KEYBOARD HANDLER

The functional make-up of Butterfly-110 is essentially two separate user-environments (ND-mode and PC-mode) inside one machine. These share a single keyboard and display. Each environment has its own variation and extension of ASCII input codes and screen display functions.

Keyboard

The Butterfly keyboard combines the requirements of both the PC and the ND, in one design. At any one time, it operates as either a PC keyboard or an ND keyboard. In addition, improved features, such as a longer key codes buffer for type-ahead than is normally available for IBM-compatible PCs, and new features such as Mouse, Bar Code Reader and Function Keypad ports, are included in the keyboard design. The keyboard handler program provides this functionality.

Because of the major additional requirements that Butterfly-110 imposes for keyboard handling, a Butterfly keyboard handler program replaces the standard PC-BIOS (INT9h and INT16h) keyboard handler.

System manager

The System Manager program handles orderly context switching between the two distinct PC and ND modes of operation, and passes the appropriate key codes to either the ND (via PC-NDI/O and the mailbox) or the PC.

Terminal handling involves saving the display and keyboard conditions when switching from PC mode to ND mode and vice-versa, then performing the switch.

CHAPTER 4 HARDWARE DESCRIPTION

TABLE OF CONTENTS

Section	Page
4.1	HARDWARE MODULES 4-5
4.2.	THE PC 4-7
4.2.1	References 4-7
4.2.2	PC interrupt system 4-7
4.2.3	PC memory assignments 4-7
4.3	THE ND-110PCX COMPUTER 4-8
4.3.1	Block diagram 4-8
4.3.2	Basic clocks 4-10
4.3.3	Internal bus system 4-10
4.3.4	Processor 4-12
4.3.4.1	General 4-12
4.3.4.2	Control Store and pipeline 4-13
4.3.4.3	The MIC 4-14
4.3.4.4	Cycle timing and control 4-19
4.3.4.5	Register File 4-26
4.3.4.6	The ALU 4-26
4.3.4.7	The MAC 4-32
4.3.4.8	Control signal decodes/functions 4-34
4.3.4.9	CPU I/O registers 4-35
4.3.4.10	Traps 4-36
4.3.4.11	Print status/installation number 4-37
4.3.5	Memory 4-38
4.3.5.1	Memory management principles 4-38
4.3.5.2	Memory map 4-38
4.3.5.3	Memory addressing 4-42
4.3.5.4	Read/write/refresh cycles 4-48
4.3.5.5	Parity and error checking 4-54
4.3.6	Interrupt system 4-56
4.3.6.1	Interrupt mechanism 4-56
4.3.6.2	Interrupt controller 4-58
4.3.6.3	Signal box interrupts 4-59
4.3.6.4	Internal interrupts 4-60
4.3.7	PC-NDI/O system 4-64
4.3.7.1	PC expansion bus 4-64
4.3.7.2	PC access to ND memory 4-64
4.3.7.3	PC program I/O 4-64
4.3.7.4	PC-ND status and control registers 4-66
4.3.8	ND-I/O system 4-67
4.3.8.1	HDLC & DMA 4-68
4.3.8.2	Console port 4-70
4.3.8.3	Panel processor 4-70

4.3.9	Real-time clock	4-71
4.3.10	Power fail and stop	4-71
4.3.11	Clear	4-71
4.4	HARDWARE OPTIONS	4-72
4.4.1	Serial Communications Board plus Connector Box	4-72
4.4.2	Tape streamer	4-72
4.4.3	Additional 1Mbyte RAM	4-72
4.4.4	HDLC communications	4-73
4.4.5	Plug-in modem	4-73
4.4.6	External Winchester disk	4-73
4.5	ND-110PCX TERMINAL DISPLAY UNITS	4-74
4.6	ND-110PCX TERMINAL KEYBOARD	4-74
4.7	MOUSE	4-74
4.8	BAR CODE READER	4-74

LIST OF ILLUSTRATIONS

Figure		Page
4.1	Butterfly terminal - main hardware modules	4-5
4.2	ND-110PCX hardware block diagram	4-9
4.3	Basic oscillator and clocks	4-10
4.4	ND-110PCX bus structure	4-11
4.5	Instruction cycle	4-11
4.6	Basic functional modules in the processor	4-12
4.7	Control store address and microcode pipeline	4-13
4.8	Control store pipeline timing	4-14
4.9	MIC functional block diagram	4-15
4.10	"branch and return" addressing	4-17
4.11	Push/pop stack	4-17
4.12	Basic cycle timing	4-19
4.13	Sub-cycles in ND-110PCX microcycle	4-21
4.14	Register file banks 0-3	4-26
4.15	ALU functional block diagram	4-27
4.16	Data path between ALU and Register File	4-28
4.17	MAC functional block diagram	4-33
4.18	Memory map for ND-110PCX	4-39
4.19	Logical-to-physical address mapping	4-42
4.20	Layout of each entry in the Page Table	4-43
4.21	LBA bus addressing for secret memory	4-46
4.22	PC addressing of ND memory area	4-47
4.23	Donation of 128kbytes of ND DRAM to the PC	4-48
4.24	Memory timing	4-51
4.25	Interrupt control system	4-57
4.26	ND-PC interrupt request control	4-59
4.27	Internal interrupts control system	4-60
4.28	ND I/O subsystem	4-67
4.29	ND I/O - DMA buffer	4-68

4.1 HARDWARE MODULES

A functional description for the main hardware modules in ND-110PCX is provided in Chapter 2. For convenience, they are shown in fig 4.1.

This Chapter provides a detailed hardware description of the modules which are of ND design, and which are not covered in other ND manuals or other suppliers' manuals.

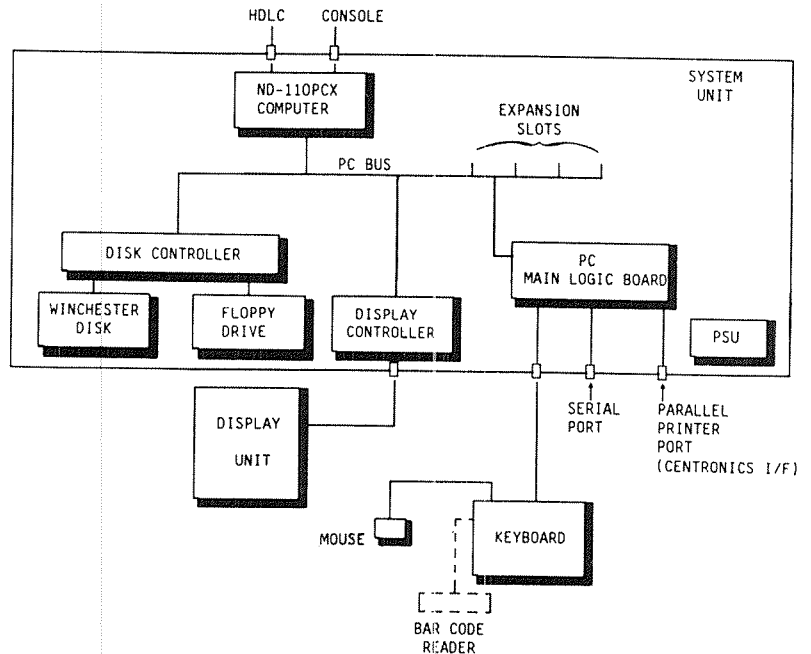


Fig 4.1: Butterfly terminal - main hardware modules

- In the System Unit, all hardware modules except the ND-110PCX computer, are covered in the PC Technical Reference Manual (ND-06 028), or in associated supplier's literature.
- The hardware description for the ND-110PCX computer is given in section 4.3, with an associated description of ND-110PCX microprogram in Chapter 5.

- The Display Unit is covered in a separate supplier's literature.
- The Keyboard is described in Appendix F.
- The Mouse and Bar Code Reader devices are covered in separate supplier's technical literature.

4.2 THE PC

4.2.1 References

Refer to the PC Technical Reference manual (ND-06.028) for a description of the PC components of Butterfly.

Appendix C of this Butterfly-110 Technical Reference Manual gives a summary description of the connectors and signal lines on the PC Expansion Bus.

Appendix F of this Butterfly-110 Technical Reference Manual gives a summary specification for the Butterfly keyboard. This keyboard provides the functionality of both a PC and a ND (NOTIS) terminal.

4.2.2 PC interrupt system

Chapter 6 section 6.6.3 of this manual gives a description of the essential structure of the MS-DOS operating system, in order to provide adequate background information to introduce how Butterfly-110 uses the PC's interrupt system. If the information in Chapter 6 is insufficient for your needs, refer to the PC Technical Reference Manual (ND-06.028) and MS-DOS Reference Manual (ND-60.271).

4.2.3 PC memory assignments

Refer to Chapter 3 section 3.3.13 and fig 3.8 for an overview of the utilisation of the PC's memory space.

4.3 THE ND-110PCX COMPUTER

Circuit Schematic The circuit schematic diagrams for ND-110PCX are included in Appendix A. They comprise the:

- CPU Board: Print No 3401, sheets 1, 2, 3, 4
- Local Bus: Print No 3402, sheets 1, 2, 3, 4.

These are referred to throughout the following description. The format of this reference is in the form:

Print no / sheet no / area on sheet

1-2 / 1-4 / A1 - G4

For example, the Local Bus Board 3402 sheet 4 area C4 is referred to as:

2/4/C4.

PAL Specification The specifications for the PALs used in ND-110PCX are included in Appendix D. Statements in the following description which identify various functions performed by these devices, may be understood in greater depth by referring to these listings.

4.3.1 Block diagram

Fig 4.2 shows a block schematic diagram for the main functional logic areas in the 2-board assembly comprising the ND-110PCX computer.

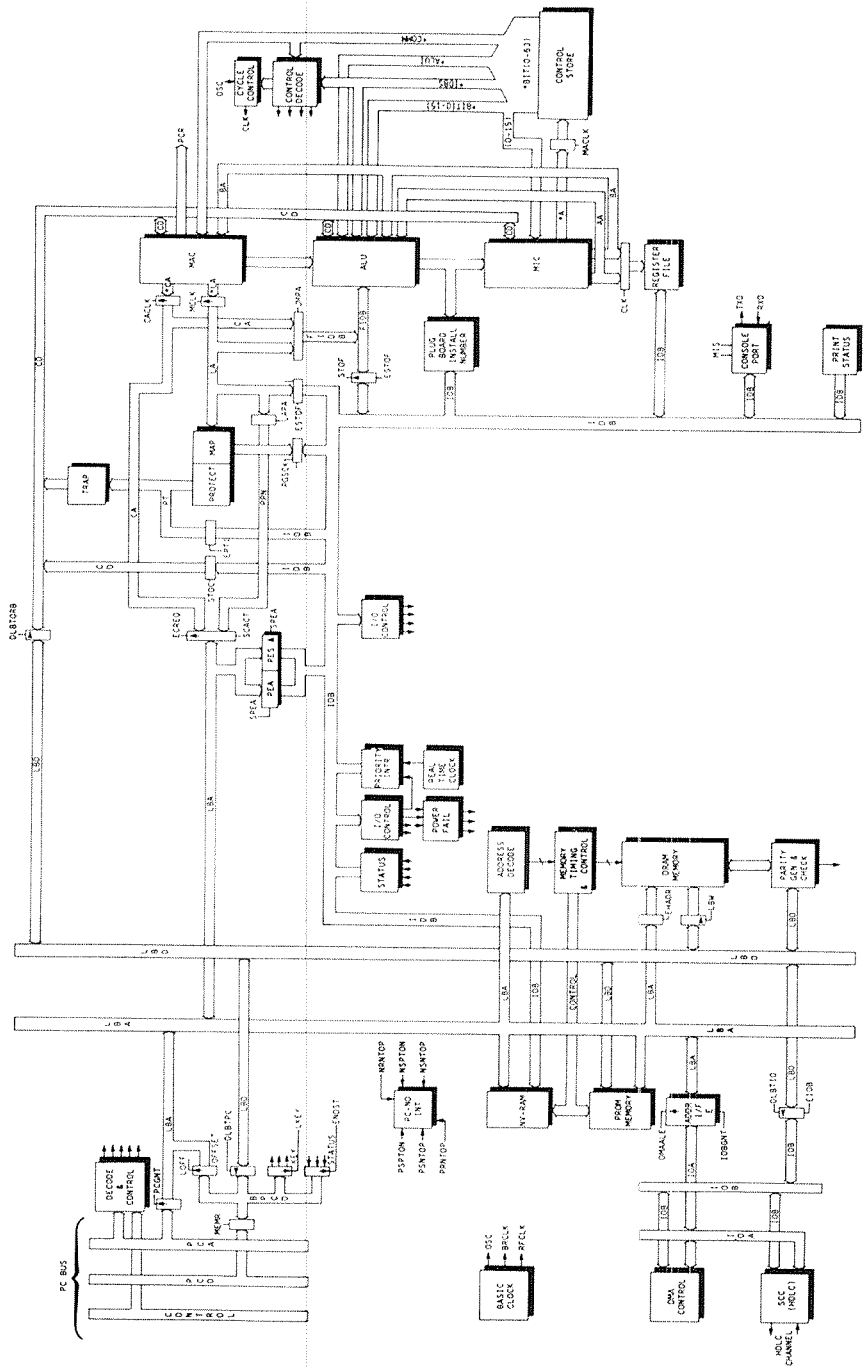
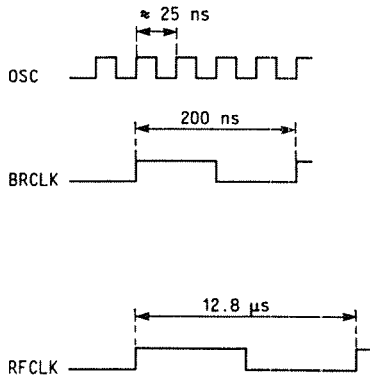


Fig 4.2: ND-110PCX hardware block diagram

4.3.2 Basic clocks



The basic clock for the ND-110PCX is derived from a 39.3216MHz crystal oscillator X101 (2/4/A1), giving basic clock OSC as a nominal 25ns period clock. For test purposes, this internal timing source may be inhibited by signal INOSCOFF, to enable an external timing source EXTOSC through to OSC

A clock divider (fig 4.3) provides a 200nsec clock BRCLK, and the 12.8μsec Refresh Clock for dynamic RAM.

Fig 4.3: Basic oscillator & clocks

4.3.3 Internal bus system

There are 4 bus systems (fig 4.2) in the ND-110PCX. These are highlighted in fig 4.4, and are as follows:

- ND110PCX internal bus:
 - Cache Data CD
 - Cache Address CA
 - Internal Data Bus IDB

If a cache memory were to be implemented as in the ND-110CX computer, the CD and CA buses of the ND-110PCX would correspond to those in the ND-110CX (RASK).

- ND-110PCX Local Bus:
 - Local Bus Data LBD
 - Local Bus Address LBA
- Personal Computer Bus:
 - PC Data PCD
 - PC Address PCA
 - PC Control
- HDLC Bus:
 - Input Output A IOA
 - Input Output B IOB

The data flow in ND-110PCX is essentially over these bus systems, which are connected through bi-directional gateways.

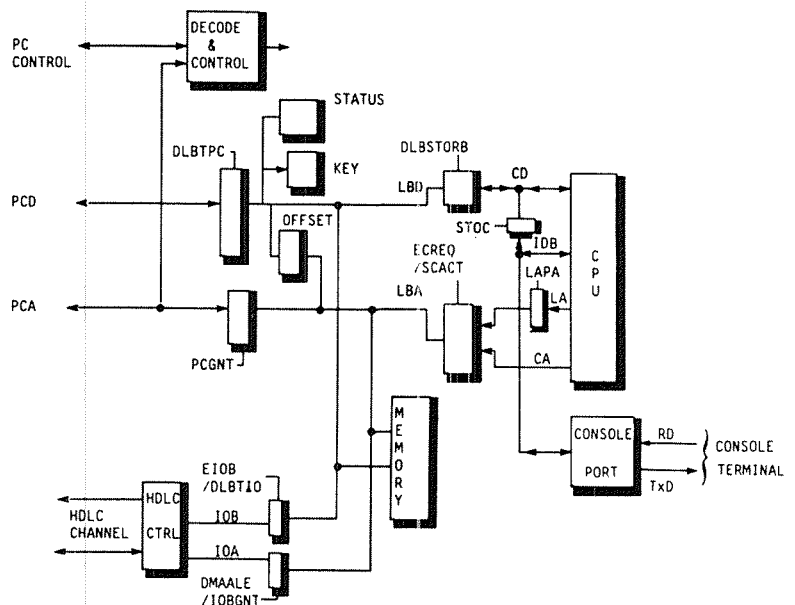


Fig 4.4: ND-110PCX bus structure

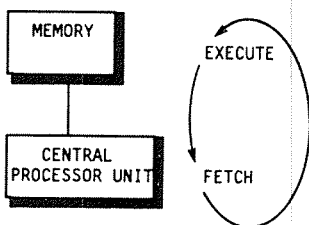


Fig 4.5: Instruction Cycle

Thus the process of program instruction fetch from memory and execution in the processor (fig 4.5) involves significant activity over the ND internal bus and local bus systems.

4.3.4 Processor

4.3.4.1 General

In microprogrammed CPU architecture, the processor hardware is controlled directly by the bits in each microinstruction. Several microinstructions may be needed to execute one macroinstruction (machine instruction).

In addition it performs other functions:

- operator communication (opcom) via the PC or Console port
- built-in test routines
- boot loaders
- HDLC communication.

The microprogram resides in the Control Store, which is made up of 8kx64-bit read-only memory (ROM).

Fig 4.6 presents the essential functional modules in the Processor. It is an extract from the overall block diagram of fig 4.2.

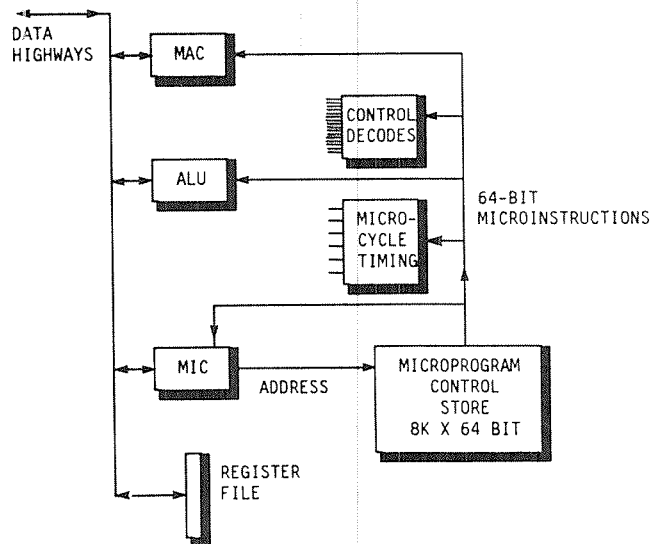


Fig 4.6: Basic functional modules in the processor

In principle, when an instruction is fetched from memory, the upper 10 bits of that 16-bit instruction are used in the MIC as an address, which defines the entry point in Control Store microprogram for this macroinstruction. The microinstruction from the Control Store, under control of the microcycle timing, controls the computational activity needed to execute the macroinstruction. This may involve decoding control signals from various fields within the microinstruction; performing various arithmetic/shift or logic functions via the ALU; testing selected conditions in the processor; and computing (in MAC) further memory addresses in order to read or write with memory.

Since several microinstructions are usually needed to execute one macroinstruction, the Sequencer part of the MIC also computes the address for the next microinstruction. As for macroinstructions, this may involve linking, using push/pop address stack (to branch or return), jump, increment to next, or repeat operations. The microcycle timing logic defines the end of processing for each microinstruction, and the start of the next. When the sequence of microinstructions for a macroinstruction is completed, a macroinstruction fetch is performed to get the next instruction from memory, the logical address being supplied from the MAC, for conversion by the MMS to physical memory addresses.

A major feature of the ND-100 series of computers is its 16 program levels, and associated 16-level Register File. The ND-110PCX supports these 16 levels of program operation. In addition, it creates conceptual levels "16" and "17" in its microcode; these are described in Chapters 3 and 6.

4.3.4.2 Control Store and pipeline

The 8kx64-bit Control Store (1/2/A,B) is made up of eight 8kx8-bit read-only memory chips.

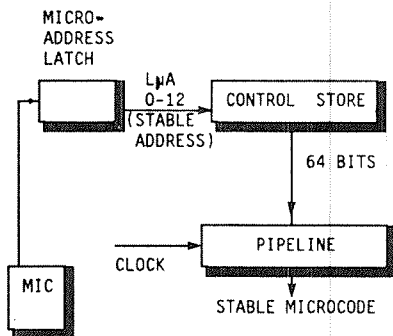


Fig 4.7: Control store address and microcode pipeline

The address lines LpA0-12 to Control Store are generated from the MIC (see section 4.3.4.3) and pipelined in the Micro-Address latch (1/1/D2). The 64-bit data output from Control Store is presented to a number of registers and PALs, which decode command signals and interpret the individual fields and combinations of fields contained within the microinstruction word. The significance of these fields, and resulting control decodes, are described in Chapter 5.

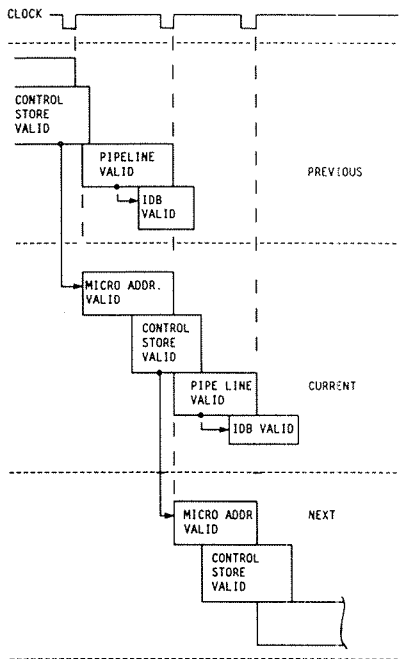


Fig 4.8: Control store pipeline timing

The "pipeline" system is used to stabilise the microinstruction from Control Store. By using a pipeline, the next address to Control Store may be applied during the current microinstruction. In this way, delays incurred in waiting for Control Store access time to complete are minimised. The principle is illustrated in fig 4.7.

In the ND-110PCX there is no single pipeline register block which latches the complete microinstruction word output from Control Store. Instead, various microinstruction bits, or their decodes, are latched as required.

The MIC (1/1/C2) supplies Control Store addresses. Clock MACLK latches these into the Micro-Address Latch, so presenting a stable address to Control Store. The memory access time of the Control Store chips is of the order of 150nsec, after which time the 64-bit microinstruction word is valid.

The next microcycle clock CLK (start of next microinstruction) then latches this microinstruction (as microcode bits or as a decoded signal) into the microcode pipeline. The timing sequences are shown in fig 4.8.

4.3.4.3 The MIC

General

The Microprogram Instruction Controller (MIC) is shown in 1/1/C2. It generates the correct sequence of addresses to the Control Store, so that the required sequence of microinstructions are fetched and executed. It also supports addressing of the 16 internal registers in the ALU, and the 16 register sets in the Register File.

The MIC is a custom-made integrated circuit, constructed from a 120-pin logic gate array. A detailed specification of this device is contained in a separate publication (ND-05.016). For convenience, a description of the main functional elements of the MIC is included here.

Fig 4.9 shows these elements, as a 1-sheet summary of the top level for the MIC.

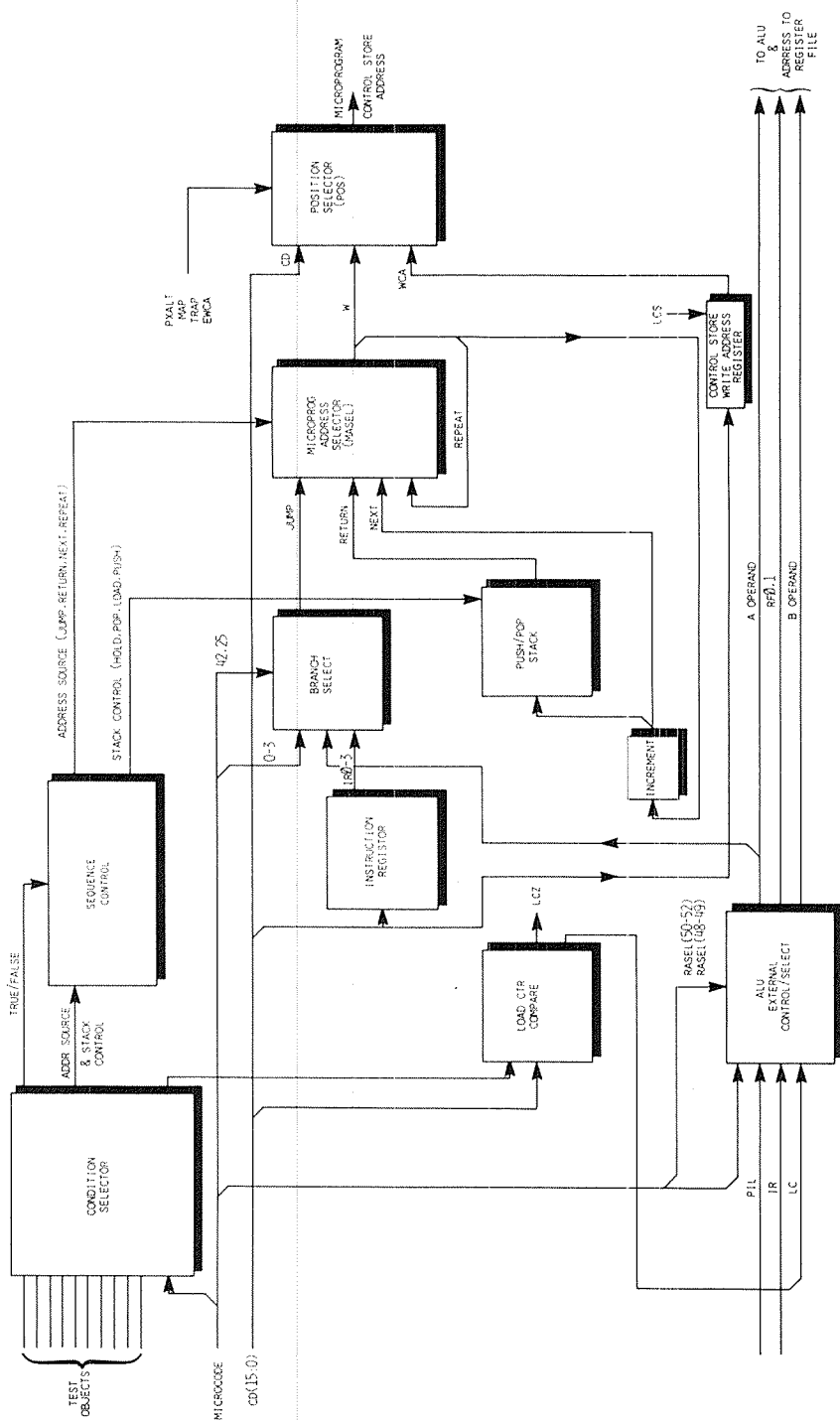


Fig 4.9: MIC functional block diagram

The microprogram address *A(0-12) from the MIC is latched by clock MACLK, into the Micro Address Latch, to stabilise the Control Store address on LpA(0-12).

Also output from the MIC are the A and B operands (AA0-3 and BA0-3) to the ALU. With bank select lines RFO-1, these provide the address to Register File (see section 4.3.4.5). If Register File extension address bit XRF3 is not active (low), this forces bank address 1 from the MIC (RFO high, RF1 low), so defining Register File Bank 1.

Signals LCZ and UP provide control inputs to the ALU.

Internal functions

The following description covers the basic functionality of the MIC, as represented in fig 4.9. For an explanation of the signal mnemonics on the MIC, refer to Appendix B.

Referring to fig 4.9, the Control Store address is selected from four possible sources, through the Position Selector.

- If the next microinstruction is to be the first for a new macroinstruction fetched from memory, signal RESTR checks if it is a privileged instruction, and acts accordingly. Signal MAP selects the CD15-6 lines, which contain the top 10 bits of the instruction op-code. These add 6000₈ to the op code value (internal to the POS), giving the Control Store vector entry point for the instruction as an address in the range 6000-7777₈. Thus the instruction vector area occupies this 1k address space.
- If the next microinstruction is to be in response to a TRAP condition (see section 4.3.4.10), then TRAP selects bits 6-9 of the CD bus as the Control Store address.
- Signal EWCA is active only when writing microprogram into the Control Store, during which time the address is generated from the CD lines, via the Control Store Write Address Register and the WCA lines. No such activity is required in ND-110PCX, and therefore signals EWCA and LWCA are tied to logic 1 (1/1/C3).
- If the next microinstruction is determined by the normal microprogram sequence itself, the W lines from the MASEL (Micro-Address SElector) are selected as the Control Store address.
- If signal PXALT is true, it forces a Control Store address above 4k, and the address within the 4-8k range is determined by the W lines.

The address source onto the W lines (fig 4.6) is decided in the Condition Selector and Sequence Control. Depending on the result of the specified test object, as

defined in the Sequencing Control bits of the current microinstruction (see Chapter 5), and whether the true or false condition is to be acted on, the address source will pass a JUMP, RETURN, NEXT or REPEAT address to the W lines. Test object inputs to the MIC are shown on 1/1/C2.

- **JUMP.** The jump address is derived from μ code bits 0-11,20 if it is a direct jump, or μ code bits 4-11,20 plus the A-operand (if it is a vectorised jump on the A-operand), or bits 0-3 of the macroinstruction if it is a vectorised jump on the current macroinstruction. If the jump involves a branch to a microcode subroutine, then the return address is saved by being "pushed" into the Stack (see RETURN below).

- RETURN

The NEXT (+1 on the current value) address is generated through a +1 incrementer, and presented to the Push/Pop Stack as well as to the MASEL. If the current microcode instruction demands a branch to another subroutine in microcode, the NEXT address is "pushed" into the Stack and JUMP provides the branch address.

Then, after the subroutine has been completed, the Sequence Control selects the RETURN address through the MASEL, and subsequently "pops" the Stack. Fig 4.10 illustrates this operation in terms of Control Store addressing.

The Stack (fig 4.11) provides 4 levels of nesting. Control operations to the Stack are traditionally referred to as Push to enter the latest return address into the Stack, and Pop to lift the top address from the Stack. In the MIC, these operations are performed by 4 Stack control lines:

- 1 HOLD: leaves the Stack unchanged
- 2 POP: pops the top entry out of the Stack, moving the entry below it up to the top of the stack.
- 3 LOAD: puts the next micro-address into the top entry in the Stack. Note that this overwrites the current top entry.
- 4 PUSH: pushes the previous top entry down one level, and places the NEXT address at the top of the Stack.

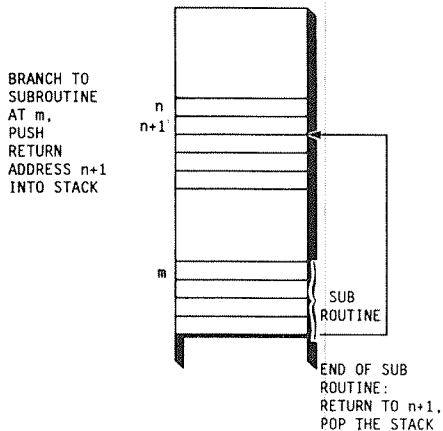


Fig 4.10: "Branch and return" addressing

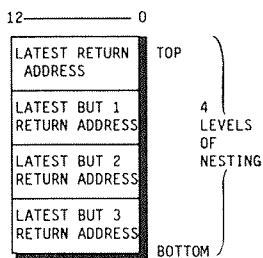


Fig 4.11: Push/Pop stack

- NEXT

As described in RETURN above, the Increment function adds 1 to the current address on the W lines, so defining the next address in Control Store.

- REPEAT

This presents the same address as that for the current microinstruction, so offering a repeat fetch of that same microinstruction.

In addition to the Control Store address, the MIC contains two further functions:

- ALU External Control/Select

The 4-bit A-operand and 4-bit B-operand are derived via this function, selected by the microcode fields RASEL and RBSEL respectively, from one of four sources:

- microcode bits 12-15 (A-operand) and 16-19 (B-operand)
- the program level (PIL0-3 - A-operand only)
- the instruction bits 3-6 (A-operand) or 0-2/3-5 (B-operand)
- the loop counter bits 0-3.

The A-operand and B-operand define the least significant 8 bits of the Register File address, with RF0/1 supplying the 2 additional bank select bits. The A and B operands also perform their traditional address/select control in the ALU.

- Loop Counter and Comparator

The Loop Counter is one input to the Condition Selector. This is presented with CD bits 0-3 to a comparator to determine the Loop Counter Zero condition (LCZ).

4.3.4.4 Cycle timing and control

Timing control in the ND-110PCX is effected by cycle decode logic. This is controlled by the microcode, and is modified by external events (e.g. interrupts, bus cycles, traps). The basic cycle timing is illustrated in fig 4.12.

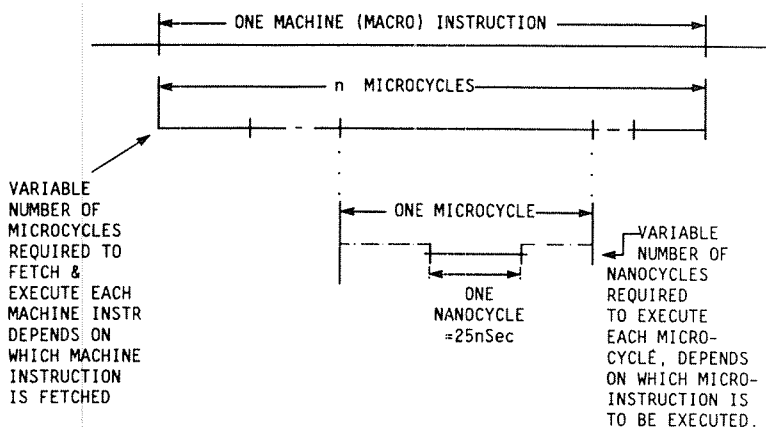


Fig 4.12: Basic cycle timing

Each macroinstruction cycle (the cycle in which a macroinstruction is fetched and executed) is divided into a number of microcycles (the period in which a microinstruction is executed).

The microcycle is in turn divided into a number of subcycles, termed nanocycles. The advantages of this are that microinstruction timings can be fine-tuned to what is happening within the microinstruction, and a finer resolution on the timing of external, sequenced events (such as bus accesses) can be achieved.

A number of different microinstruction lengths are possible, depending on the microinstruction being executed, and on external events. The nano-cycle clock frequency (40MHz) specifies these lengths in 25nsec increments.

It should be noted that the ND-110CX (RASK) and ND-110PCX differ from the ND-100, in that the microinstruction does not wait for a bus write cycle to complete before continuing to the next microinstruction.

Cycle Lengths Subcycle timing in the ND-110PCX microcycle is illustrated in fig 4.13.

The ND-110PCX microinstruction type can be:

- short
- slow
- break
- fetch
- read
- write.

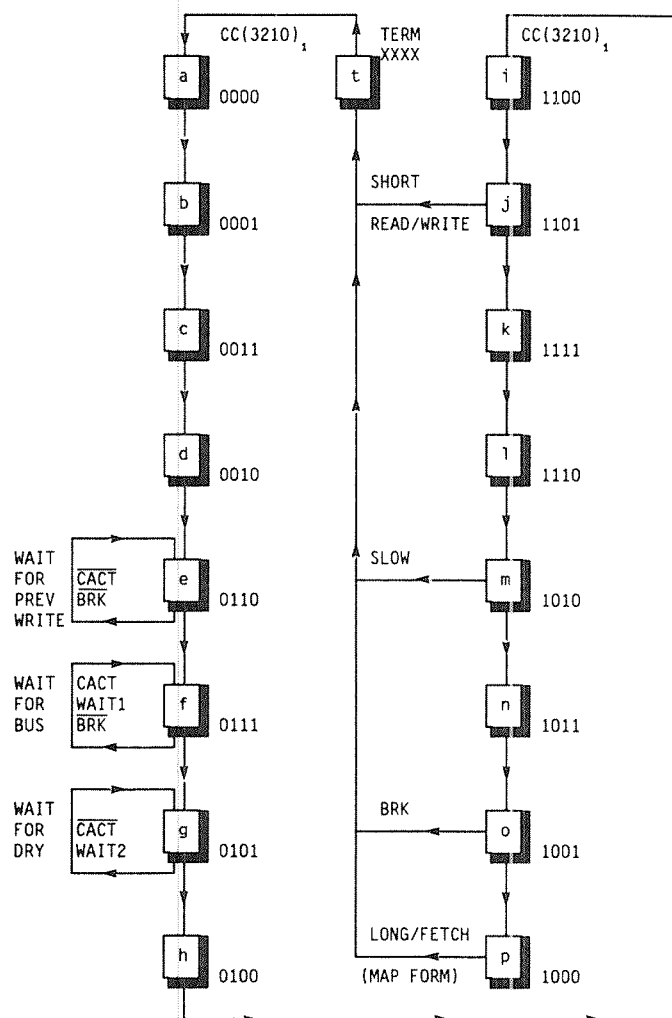
If a read or a write is in progress, then the nano-sequencer loops at cycles e, f and g (fig 4.13), as required, for external bus activity. A microcycle that only involves internal CPU activity or a read/write access, is a "short" cycle and terminates at sub-cycle j. This is the minimum time taken for a new micro-address to appear at the input to the pipeline registers.

Some internal CPU functions take some time to complete, and these slow cycles terminate at sub-cycle m. A cycle involving a trap is a break cycle, and finishes at o. This extra time is required because a different microinstruction has to be accessed from the Control Store. Fetch cycles, and some internal accesses, require a long cycle: the fetch cycle has to access the memory (finished at g) for a new macroinstruction and then present the instruction to the pipeline register.

The cycle times involving writes or reads, where loops on either e, f or g occur, can not be defined as they depend on external bus activity. For other cycles, the cycle time can be defined and is:

$$[\text{number of nano-cycles from a to last cycle} + 1 \{t\}] \times 25\text{nsec}$$

i.e. short is 275ns, slow 350ns, and long 425ns.



Note: the 4-digit code defines the value of the cycle counter for that state.

Fig 4.13: Sub-Cycles in ND-110PCX microcycle

Control Method The Nano-Cycle Sequencer PAL (1/2/D5) is controlled by signals decoded from the microinstruction word (DLY1-0, SHORT, COMSL, FORM, WAIT1-2), and external inputs CACT (bus activity by CPU) and BRK (trap or conditional break). Signals DLY0-1 from the Condition Control PAL (1/2/E3) are not used at present on the ND-110PCX. On the ND-110CX (RASK), they determine the length of the short cycle.

The decoded command-field signals (from microinstruction bits COMM4-0 and MIS1-0) are direct commands to the Sequencer: SHORT declares a short cycle, COMSL a slow cycle and FORM a fetch. WAIT1 forces a loop on f (fig 4.13) for a bus access grant (read and write), while WAIT2 forces a loop on g to wait for data ready (read). The ND-110CX (RASK) signals which denote "always slow" and "slow if not a hit in cache" have been removed.

Signal BRK is active if there is either a trap or a conditional break. A trap occurs if there is a protect fault or an interrupt. The SINTRAN IV exceptions have been removed. A conditional break is used during microprogram jumps. The conditions available are F=0 (ALU result = 0), F15 (ALU result bit 15), CRY (ALU carry out), WP (write to P register) and SGR (sign greater than).

**Cycle Lengths
for
µcode Commands**

The following tables give the cycle lengths for the microcode commands (microinstruction COMMO-4 and MIS0-1, see also Chapter 5 fig 5.2). In these tables the following notes apply:

- DLY and DELAY are microword controls for the cycle length and are not used in the ND-110PCX.
- SHORT indicates to the cycle control that this is a short cycle. Similarly COMSL indicates a slow cycle.
- FORM indicates that a fetch cycle is occurring.
- WAIT1 indicates that the cycle must wait for a bus cycle to start (i.e. a read, write or fetch), and WAIT2 requires the cycle to wait for bus cycle finish (i.e. read or fetch).
- LDIRV means that the instruction register is loaded, and IND denotes an indirect read.
- Command decodes 00-17 are internal CPU control
20-27 are sequencer control
30-37 are external or memory request.

Internal CPU Control

<u>µcode field</u>			D	S	W	W	C	F
			E	H	A	A	O	M
COMM	MIS	MNEM	D	L	I	I	M	O
			L	A	T	T	S	R
			Y	Y	T	1	2	L
00		NONE	.	.	*	.	.	.
01		LDPII	.	.	*	.	.	.
02		LDGPR	.	.	*	.	.	.
03		EWRF	.	.	*	.	.	.
04		CLIRQ	.	.	*	.	.	.
05	0	CLRHD1	.	.	*	.	.	.
05	1	CLRHD2	.	.	*	.	.	.
05	2	RPTON	.	.	*	.	.	.
05	3	SNTOP	.	.	*	.	.	.
06	0	SPTON	*
06	1	(WRIPT)	.	.	*	.	.	.
06	2	-	*
06	3	LDPCR	.	.	*	.	.	.
07		SIOC	.	.	*	.	.	.
10		SLOW (<RASK)	*
11	0	R,PCI,RX	.	.	*	.	.	.
11	1	R,PCI,ST	.	.	*	.	.	.
11	2	R,PCI,MOD	.	.	*	.	.	.
11	3	R,PCI,COM	.	.	*	.	.	.
12	0	W,PCI,TX	.	.	*	.	.	.
12	1	W,PCI,SYN	.	.	*	.	.	.
12	2	W,PCI,MOD	.	.	*	.	.	.
12	3	W,PCI,COM	.	.	*	.	.	.
13		START	.	.	*	.	.	.
14		STOP	.	.	*	.	.	.
15		CLRTC	.	.	*	.	.	.
16		CLFF	.	.	*	.	.	.
17		LDLC	.	.	*	.	.	.

Sequencer Control

<u>µcode fields</u>			D	S	W	W	C	L
			E	H	A	A	O	F
			D	L	O	I	M	O
COMM	MIS	MNEM	L	A	R	T	S	R
			Y	Y	T	1	2	L
								M
								V
								D
20	0	LDSEG	.	.	*	.	.	.
20	1	(LDDOMI,SDOM)	.	.	*	.	.	.
20	2	(LDPS)	.	.	*	.	.	.
20	3	LDIRV	.	.	*	.	.	*
21	0	(WCIHM)	.	.	*	.	.	.
21	1	SSEMA	.	.	*	.	.	.
21	2	CCLR	.	.	*	.	.	.
21	3	LDEXM	.	.	*	.	.	.
22	0	IREAD,PT	.	.	*	*	*	*
22	1	IREAD,APT (~1)	.	.	*	*	*	*
22	2	MAP	*	*
22	3	CNEXT,NWP	.	.	.	*	*	*
23	0	CJMP,F15	.	.	.	*	*	*
23	1	CJMP,NF15	.	.	.	*	*	*
23	2	CJMP,F=0	.	.	.	*	*	*
23	3	CJMP,NF=0	.	.	.	*	*	*
24	0	CNEXT,SGR	.	.	.	*	*	*
24	1	CNEXT,NSGR	.	.	.	*	*	*
24	2	CNEXT,CRY	.	.	.	*	*	*
24	3	CNEXT,NCRY	.	.	.	*	*	*
25	0	CNEXT,F15	.	.	.	*	*	*
25	1	CNEXT,NF15	.	.	.	*	*	*
25	2	CNEXT,F=0	.	.	.	*	*	*
25	3	CNEXT,NF=0	.	.	.	*	*	*
26	0	JMP,*	.	.	.	*	*	*
26	1	JMP,B	.	.	.	*	*	*
26	2	JMP,I	.	.	.	*	*	*
26	3	JMP,X	.	.	.	*	*	*
27	0	JMP,XB	.	.	.	*	*	*
27	1	JMP,XI (,B)	.	.	.	*	*	*
27	2	LBCONT	.	.	.	*	*	*
27	3	CONTINUE	.	.	.	*	*	*

(~1) IREAD,APT also executes a LDGPR command.

External or Memory Request

µcode field			D	S	W	W	C	F	L			
			E	H	A	A	O	O	D			
			L	O	I	I	M	R	I			
COMM	MIS	MNEM	Y	Y	T	1	2	L	M	R	N	D
30	0	AREAD,*	.	.	*	*	*
30	1	AREAD,B	.	.	*	*	*
30	2	AREAD,I (,B)	.	.	*	*	*
30	3	AREAD,X	.	.	*	*	*
31	0	AREAD,XB	.	.	*	*	*
31	1	AREAD,XI (,B)	.	.	*	*	*
31	2	IREAD2	.	.	*	*	*
31	3	AREAD,NEXT	.	.	*	*	*
32	0	AWRITE,*	.	.	*	*
32	1	AWRITE,B	.	.	*	*
32	2	AWRITE,I (,B)	.	.	*	*
32	3	AWRITE,X	.	.	*	*
33	0	AWRITE,XB	.	.	*	*
33	1	AWRITE,XI (,B)	.	.	*	*
33	2	AWRITE,HOLD	.	.	*	*
33	3	AWRITE,NEXT	.	.	*	*
34	0	READ,PT	.	.	*	*	*
34	1	READ,APT	.	.	*	*	*
34	2	READ,HOLD	.	.	*	*	*
34	3	EXAMINE	.	.	*	*	*
35	0	WRITE,PT	.	.	*	*
35	1	WRITE,APT	.	.	*	*
35	2	WRITE,HOLD	.	.	*	*
35	3	DEPOSIT	.	.	*	*
36	0	(ADCS)	.	.	*
36	1	(RWCS)	.	.	*
36	2	WLBR	.	.	*
36	3	WALBR	.	.	*
37	0	(PLBR)	.	.	*
37	1	SPARE
37	2	(IDENT)	*
37	3	(IOX)	*

4.3.4.5 Register File

For a description of the function of the Register File in the ND-110PCX, refer to the ND-110 Functional Description Manual (ND-06.026).

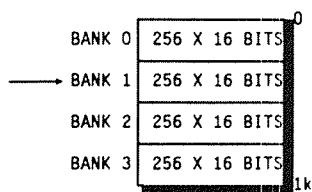


Fig 4.14: Register File
banks 0-3

The memory used in ND-110PCX for Register File storage comprises four 1kx4-bit RAM chips (1/1/F4). It is addressed by the MIC, using the A-operand and B-operand (8-bits), plus 2 bank-select bits RF0-1 (see section 4.3.4.3). The organisation of Register File is 16 sets of 16 registers per set, each register being 16-bits wide. This is shown in fig 4.14.

Hence, Register File uses only 256 of the available 1k registers in this memory space. It addresses bank 1 (fig 4.14).

Some of the remaining space in this area is used as fast memory, temporary storage (as in the ND-100CE and CX) - e.g. 16 locations are used to save PCR.

4.3.4.6 The ALU

Overview

The Arithmetic and Logic Unit (ALU) is shown on 1/1/A2. It is sometimes also referred to as the ND-BUFALU. It provides the same arithmetic and logic functionality as four AM2901 4-bit microprocessor bit slice chips (as used in the ND-100), so giving full 16-bit ALU facility. However, the control of this ALU is customised to respond to the particular format of the ND-110PCX microinstruction word. Also, the gate array incorporates key operating registers, which in the ND100 were discrete hardware.

The ALU is a custom-made integrated circuit, constructed from a 144-pin logic gate array. A detailed specification of this device is contained in the RASK Gate Arrays Technical Description (ND-05.016). For convenience, a description of its main functional elements is included here. Fig 4.15 is a reproduction of the top level schematic for the ALU gate array, obtained from the Daisy CAE system.

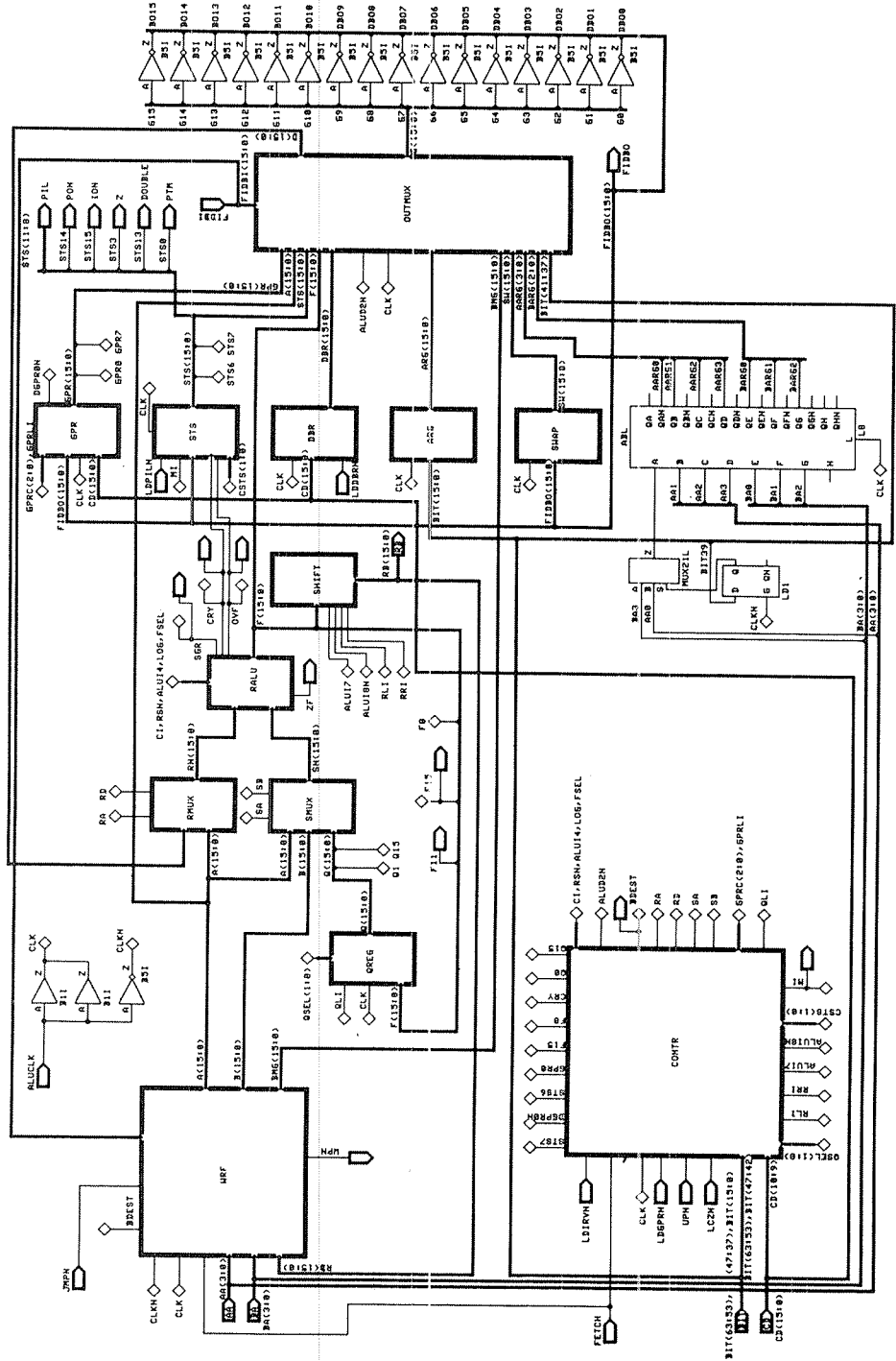


Fig 4.15: ALU functional block diagram

Functionality within the ALU

The ALU provides a large range of arithmetic and logic operations, on data presented from a variety of sources, with the computed result routed to a variety of destinations. The ALU source, computational operation required, and destination, are all controlled from the microinstruction word. A description of this control is given in Chapter 5, from which an appreciation of the ALU data inputs, internal data flow, data outputs, microcode control decodes and status condition controls, may be obtained. The following description identifies the main elements of the above, with reference to the ALU logic schematic in fig 4.15.

The Working Register File WRF (fig 4.15) consists of sixteen 16-bit registers. These contain the eight working registers on the current level, plus the eight scratch registers (see Register File, section 4.3.4.5).

When changing from one program level to another, the working registers (X,T,A,L,B,P,D,STS) in the WRF current register set are written into the current level registers in the Register File, and the registers for the new level copied from the Register File into the WRF. The eight scratch registers contain temporary information managed by the microprogram, such as addresses during memory reference instructions, and temporary results during floating point operations. These scratch registers are not saved or restored to the WRF under a level change operation, or conserved between macroinstructions.

Fig 4.16 shows the communication path between the Register File and the ALU WRF.

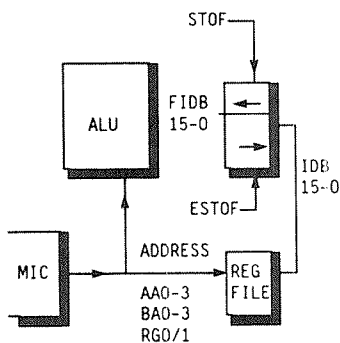


Fig 4.16: Data path between ALU and Register File

To transfer from Register File to ALU WRF, data from the addressed register in Register File is routed through the IDB transceivers, by STOF and ESTOF low (1/1/E1), onto FIDB 15-0. In the ALU (fig 4.15) it enters on the FIDBI 15-0 (internal FIDB), to the WRF block, where it is written to the working register correspondingly addressed by the BAO-3 lines.

To transfer data from ALU WRF to Register File, data from the addressed register is passed unmodified through the RMUX, RALU and OUTMUX onto the FIDBO 15-0 (external FIDB) lines, then through the bus controller by ESTOF/STOF (both high), to Register File.

The A-source and B-source select lines (AA0-3, BA0-3) from the MIC, each select output from one of the sixteen working registers in the WRF block, and the WRF register outputs are passed to the RMUX and SMUX selectors on A15-0 and B15-0. The alternative input to RMUX is D(15-0) from the OUTMUX, and input sources to OUTMUX that may be routed onto D(15-0) (selected by microcode IDB control bits 37-41) are:

- General Purpose Register (GPR15-0)
- Status Register (STS15-0)
- ALU Result Register (F15-0)
- Data Bus Register (DBR15-0)
- Microcode Argument Register (ARG15-0 - μ code bits 15-0)
- Bit Mask Generator (BMG15-0)
- Byte Swap Register (SW15-0)
- Microcode A-argument (AARG3-0)
- Microcode B-argument (BARG2-0)

Note that other sources may be externally sourced onto the IDB via the FIDB I/O port.

The SMUX selector inputs comprise both WRF outputs A15-0 and B15-0, together with a general purpose ALU process register (the Q-register) which holds the ALU result (F15-0), suitably manipulated and stabilised to become Q15-0. This holding register QREG can be shifted right or left, or linked to any of the working registers for double shifts. The RMUX selector lines RA/RD and SMUX selector lines SA/SB (controlled from microcode ALU instruction ALUI0-2 bits 55-57) determine which of these sources are passed on RN15-0 and SN15-0 respectively, to the ALU. The computed result is presented on F15-0, with output status lines SGR (sign greater than), CRY (carry), OVF (overflow) and ZF (zero flag).

The ALU result F15-0 is presented to the SHIFT function which, under control of the microcode ALUM0-1 bits 44-45, may perform a variety of manipulations (see Chapter 5). The resulting output RB15-0 is fed back to the working register file WRF, and output from the ALU gate array (1/1/B2).

The other output from OUTMUX, G15-0 (fig 4.15), drives the FIDB15-0 bus from the ALU, and is also an input source to the ALU's SWAP, STS and GPR registers. G(15-0) has all the inputs of D(15-0), plus A(15-0) from the WRF block, and this allows direct output from the working register file whilst a separate internal ALU operation is in progress. Note too that the ND-110PCX cache data bus CD15-0 is an input source for the GPR and the DBR.

The functions performed by other source data and control decodes to the GPR, STS, LBR, ARG, SWAP and A/B-source select registers, may be deduced from studying the ALU and internal data bus fields (bits 37-63) of the microinstruction word (see Chapter 5). Control decodes are supplied both externally to the ALU gate array, and internally via the CONTR block (fig 4.15).

For further information on the internal operation of the ALU gate array, refer to the MASK Gate Arrays Technical Description, ND-05.016.

To illustrate how the ALU operates in terms of data flow and manipulation, two examples are included here.

Example 1: ALU operation for the LDA instruction:

The operand to be loaded into the A register is enabled from the DBR onto the IDB. The microprogram selects the IDB input as R-operand, and passes it unmodified through the ALU. Destination is selected to be the current register block. The register number is then given by the B-source input, in this case 5. The operand is therefore loaded into the A register. A and B source-select may be used when reading from the Register File, but only the B source-select is used when writing to the current register set.

Example 2: ALU operation for the RAD1 SA DB instruction:

The A source-select reads the A register contents from the current register block, and the B source-select reads the B register contents. The microprogram selects the A register value as R-operand and the B register value as S-operand. The ALU function is selected to be $R + S$, with the destination of the result of the operation to be the current register block. In the current register set, the register to be written into is given by the B source-select, which is equal to the B register.

ALU Connections The functions of the signal line connections with the ALU are summarised below. Note that the ND-110PCX does not use all of the available signal connections to the ALU gate array; only those used in ND-110PCX are described here.

Referring to the CPU logic schematic (1/1/B2) and the ALU gate array schematic (fig.4.15):

- AAO-3 and BAO-3 provide the ALU Working Register File address. They also provide the AARG3-0 and BARG2-0 input to the ALU OUTMUX as sources onto the FIDB15-0 bus.

- CD15-0 is input to the ALU GPR. Bits 10 & 9 also provide for input of macroinstruction shift mode information to the ALU control block (see μ code bits 44-45, IR Shift, Chapter 5).
- FIDB15-0 is a bi-directional bus. Input is on FIDBI15-0 to the ALU OUTMUX and WRF; this is used when the MAC updates the P-register in WRF after a jump instruction. Output is via buffers from the OUTMUX onto FIDBO15-0, which also provides an input data source to the GPR, STS and SWAP registers. The bi-directional buffers (not shown on this level of the Daisy CAE schematic fig 4.15) are controlled by signal ED0.
- RB15-0 is the output of the ALU Shift register.
- JMP operates on the Working Register File, to control loading the P register during a jump instruction.
- WP indicates that a write operation has been made to the Working Register File P-Register, and therefore the MAC's copy of the P-Register must be updated, and a new fetch operation will be required before execution of the next macroinstruction.
- LDDBR controls loading of the DBR.
- LDGPR controls loading of the GPR.
- LDIRV strobes the CD10-9 bits into the ALU.
- LDPIL enables loading of the STS register bits 15-8.
- XFETCH controls source selection for the GPR during macroinstruction fetch operations. It also operates on the Working Register File.
- LCZ and UP are control inputs to the ALU CONTR block, affecting operation of the Q Register and Shift Register.
- F=0 (also known as ZF), F11 and F15 are status lines from the ALU Result Register, as are SGR, CRY and OVf.
- PIL3-0, PON1, Z, DOUBLE and PTM are bits output from the STS register (see Register File section 4.3.4.5, and ND-100 Functional Description of Status Register).
- The inputs from Control Store apply microcode control of the operations of the ALU. For details of specific control actions, refer to the microprogram description in Chapter 5.

4.3.4.7 The MAC

Overview

The Macroprogram Address Controller (MAC) generates the logical (virtual) addresses for all ND-110PCX accesses to main memory. It computes these 16-bit addresses under control of the microprogram, from a number of sources, including the Paging Control register, Segment and XPT registers, the microinstruction word itself, and of course the program counter. To speed calculation of new addresses, the MAC incorporates its own adder rather than calling on the ALU. To this end, it keeps a working copy of the ALU program counter, along with copies of the X and B registers.

The MAC is a custom-made integrated circuit, constructed from a 120-pin logic gate array. It is shown in schematic 1/3/A3.

A detailed specification of this device is contained in a separate publication (RASK Gate Arrays Technical Description, ND-05.016). For convenience, a description of the main functional elements of the MAC is included here.

Fig 4.17 is a reproduction of the top level schematic for the MAC gate array.

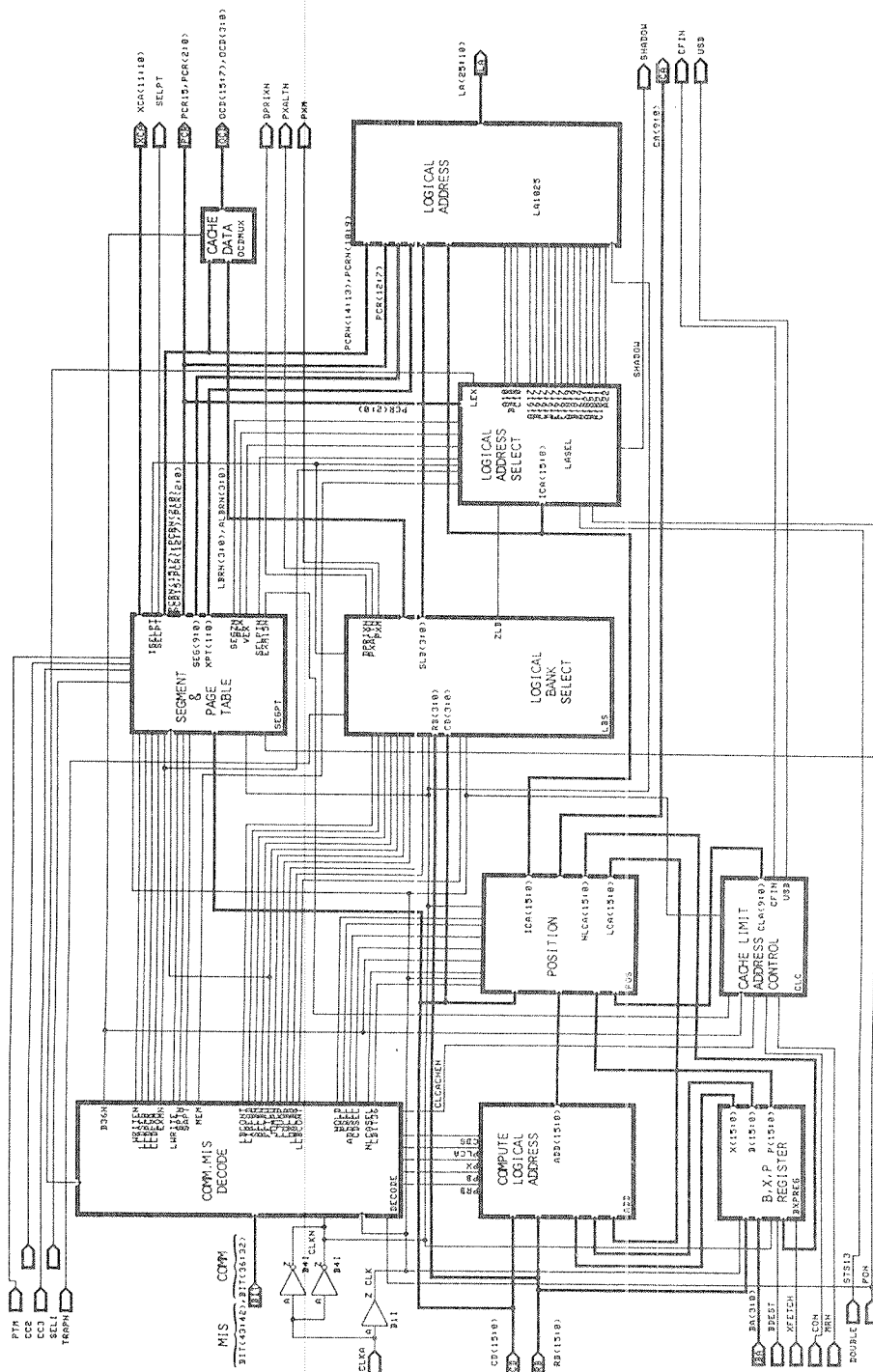


Fig 4.17: MAC functional block diagram

Memory management The ND-110 Functional Description (ND-06.026) provides an overview of the Memory Management System. It includes virtual addressing through page tables, and operation of ring protection, the paging control register, and shadow memory.

Memory address The microcode COMM:MIS fields define what memory operations (if any) are required. The MAC uses the unlatched signals *COMM(0-4) and *MIS(0-4) direct from the microcode (1/3/A2), latches them internally, and decodes them to identify the required memory addressing operation. The current values of the P, X and B registers are then used to compute this memory address, which defines a single location within the (legal - 16M) address range of the ND-110PCX. Status register bits 0 (PTM) and 13 (DOUBLE), and the paging-on indicator (PON), plus XFETCH (if an instruction fetch is involved), are all used in computing the 26-bit address output of the MAC.

This address comprises:

- *CA(0-9), which provides the address within the specified 1k page area.
- *LA(10-15), which provides the virtual page address to the page tables (see section 4.3.5.3).
- *LA(16-21), which selects the required page table (see section 4.3.5.3 and fig 4.19).
- Address bits 22-25 provide for future expansion - they are not used in the ND-110PCX's MMS system.

The PCR bits specify the memory ring protect status associated with the current register level (0-15) of the CPU, as defined by the PIL(0-3) lines.

If memory access is to Shadow area, the MAC outputs SHADOW active, which is latched and inputs LSHADOW active to the MMU Control PAL (1/3/A1). See section 4.3.5.3.

4.3.4.8 Control signal decodes/functions

Several PALs are used to decode the microcode command field and associated cycle timing parameters. These are shown in 1/2/C,D,E. The control functions implemented by these decodes are summarised in the description of the microinstruction word (Chapter 5). The associated cycle timing requirements are summarised in section 4.3.4.4.

Other control decodes are implemented in logic areas local to the function(s) in hand.

4.3.4.9 CPU I/O registers

I/O Control Register

This is shown in two parts, one in 1/1/E3, the other in 2/4/D3. It is loaded from the IDB, by the microcode command decode SIOC. The I/O Control Register bits are:

<u>Bit</u>		<u>Active</u>
15	Secret flag	high
14	X21 communications	high
13	IEPTON: enable PC to ND interrupt	high
10	Int Enable 10: power fail	high
9	Int Enable 9: memory out of range	high
8	Int Enable 8: parity error	high
5	Int Enable 5: zero	high
4	CPU request to I/O bus arbiter CRQ1	low
3	Reset real time interrupt RESRT	high
2	OPCOM LED L1	low
1	Run LED L2	low
0	Self test LED L3 + enable master clear EMCL	low

I/O Status Register

This register (2/4/D5) is enabled onto the IDB by the microcode IDB source decode EIOR. The I/O Status Register bits are:

<u>Bit</u>		<u>Active</u>
9	spare	
8	OPCOM request	low
7	PCFAIL: from the PC	high
6	TEST: from the console connector	low
5	PCPRES: PC present	low
4	CACK: CPU granted IO bus	low
3	BATNOK: battery not OK	high
2	NTOPINT: ND to PC interrupt set	low
1	PTONINT: PC to ND interrupt set	low
0	spare	

4.3.4.10 Traps

A trap is generated when it is necessary to break the microcode sequence in order to service a special condition. The trap control logic generates TRAP (1/2/F4), which selects the next microinstruction address (from the MIC) to be defined by a trap vector presented on the CD(6-9) lines. See also MIC section 4.3.4.3.

The trap vector generation logic is shown on 1/3/G. Outputs TRAP1, TRAP2, and TRAP3 are active if any input trap condition to their respective PALs is active (irrespective of the ETRP1/2/3 enable inputs to these PALs). Any one of TRAP1/2/3 active generates BRK (1/2/D4) to the Clock Generation PAL (1/2/E4), which generates TRAP at the next permitted time. This in turn activates the enable trap priority logic (1/3/G4).

- If TRAP1 (1/3/G1) is active, then at CC2 time ETRP1 enables the highest priority Trap PAL to output its selected trap vector onto the CD(6-9) lines.
- If TRAP2 is active and TRAP1 is not, then at CC2 time ETRP2 enables the second highest priority Trap PAL to output onto the CD(6-9) lines.
- If TRAP3 is active and neither of TRAPS 1 & 2 are active, ETRP3 similarly enables output from the third Trap PAL.

The input conditions to the Trap PALs are shown on 1/3/G. These give rise to the following Trap Vectors:

Vector Address	Description	Trap Enable (ETRPn)
00	master clear/power clear	3
01	page fault	2
02	protect violation	1
03	ring-down trap	1
04	page used trap	
05	write in page trap SINTRAN III	
06	spare	
07	spare	
10	spare	
11	spare	
12	spare	
13	spare	
14	spare	
15	spare	
16	panel interrupt	3
17	macro interrupt	3

4.3.4.11 Print status/installation number

Print Status	An 8-bit link station is provided (1/3/F5) in order that the modification status may be read by a MAC instruction. The status is read onto IDB4-11 by microcode IDB source decode ALD (TRAALD on the schematic). It is used in the VERSN instruction - see section 3.1.
Installation PROM	<p>An "installation descriptor", which plugs into connector J2 on the 3401 CPU board (1/1/D4), provides for implementation of a Butterfly-110 "characterisation" scheme which enables Norsk Data to track the configuration of each machine.</p> <p>The standard implementation is for a small Installation board to be plugged into connector J2. This piggy-back board (print number 3404) accommodates up to six 32x8-bit PROMs, designated A to F. It may be that positions B-F will be fitted with PALs instead of PROMs, to add further security against unauthorised copying.</p> <p>Each PROM position provides information describing the authorised configuration of the following Butterfly-110 features:</p> <ul style="list-style-type: none">• A: CPU number• B: SOFTWARE configuration• C: DISK C configuration• D: DISK D configuration• E: HDLC configuration• F: TERMINAL configuration <p>Only PROM A is necessary for the minimum configuration. The other descriptors (B-F) are added when upgrades are installed. For this reason, PROM A is soldered into the board; the other positions are fitted with header sockets. Chapter 7 section 7.3.10.2 includes a brief description of how Butterfly-110 uses this configuration information.</p> <p>The information encoded within the Installation board is read during system start-up, and at other points as required, to verify that the machine has an authorised configuration. The hardware mechanism to read the information is a 5-bit address comprising the PILO-3 lines plus BIT20, and data is read onto IDB0-7 by microcode source decode RINR.</p>

4.3.5 Memory

The memory system provides for up to 1Mwords of Dynamic RAM, 2kbytes of non-volatile RAM (NV-RAM), and up to 32kwords of PROM.

The current memory configuration is:

- DRAM : first 1Mbyte standard, second 1Mbyte optional
- NV-RAM: 2kbytes fitted
- PROM : 16kbytes fitted.

Memory is centred around the local buses LBA and LBD.

4.3.5.1 Memory management principles

See the ND-110 Functional Description (ND-06.026) for a description of the Memory Management system. In summary, this covers:

- Virtual/physical addressing.
- Paging and Ring-Protection - Page Tables, Page Control Registers.
- Address translation - virtual (16-bit) to physical (24-bit).
- Memory Protection: Page protection; Ring protection; Privileged instructions.
- Software control: PON & POF; SEX & REX; paging control; paging status.
- Shadow memory.

4.3.5.2 Memory map

Fig 4.18 shows the memory areas in the ND-110PCX. The "Address from Memory Management Unit" column shows the memory management address ranges for each area. The "physical memory" column shows the actual address values as applied to the memory devices themselves.

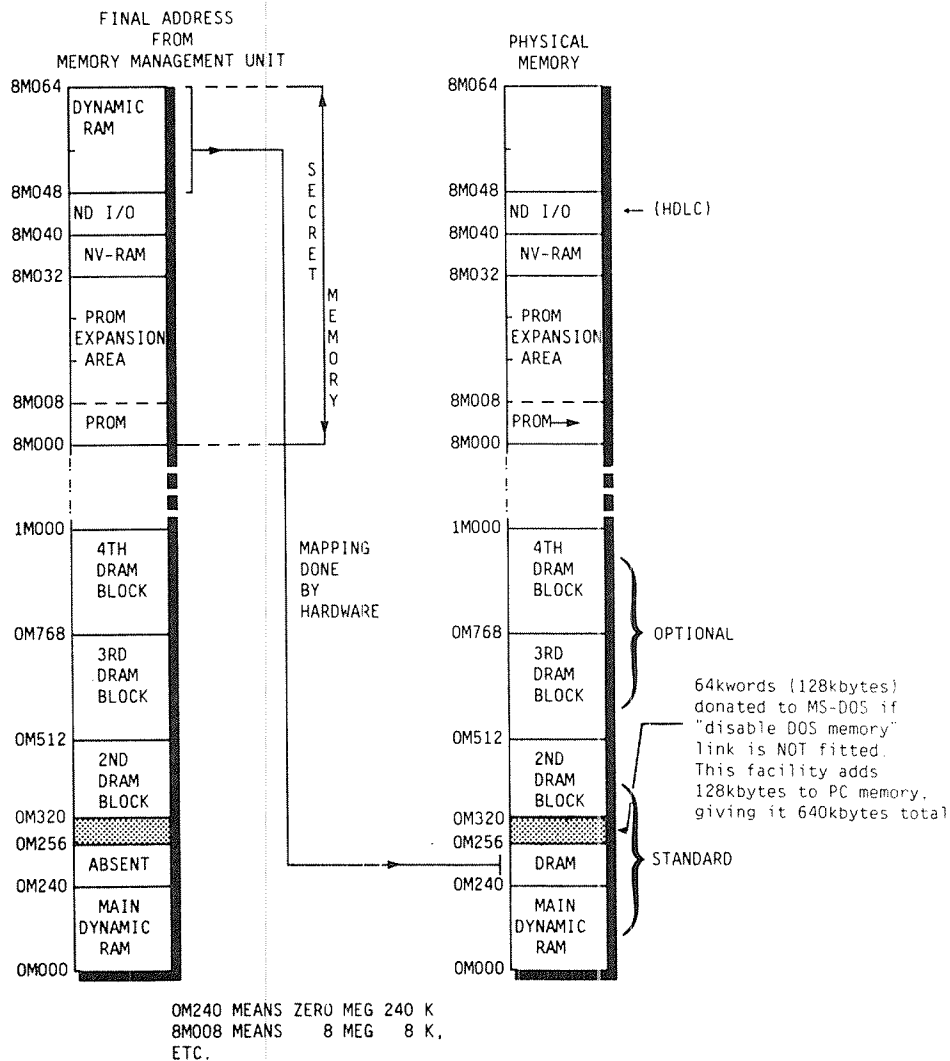


Fig 4.18: Memory map for ND-110PCX

Dynamic RAM

Fig 4.18 shows the addressing range of the DRAM, in 256k blocks. It ranges from 0M000 to 0M512 for the standard 1Mbyte memory. The optional extra 1Mbyte of memory extends this addressing range to 1M000.

The DRAM memory is shown on schematic 2/2/E1,2. Each block on the drawing represents 256kx9 bits of DRAM, providing for storage of 8 data bits plus 1 parity bit. A pin-out connection diagram for one such block is shown on 2/2/G3,4. Two blocks in one column therefore represent 512kx8 bits + parity (enabled by row/column address strobes RAS0,1/CAS0,1 respectively). Two blocks in two columns then provide 512kx16 bits + parity. The four blocks enabled by RAS2,3/CAS2,3 provide the optional extra 512kx16-bit memory, when fitted.

The top 16k of physical address area in the first 256k block of DRAM is reserved by the operating system, as part of the ND-110PCX "secret memory" (see below). It is mapped functionally to address area 8M048-8M064 (fig 4.18). This memory space is used as "mailbox" area for ND-PC intercommunication (IOX/IDENT simulation), by the HDLC program, and for various utility program purposes. It is not available to main memory.

Facility is included to "donate" 128kbytes of the ND's DRAM to the PC, adding this on top of the PC's own 512kbytes of RAM (see Chapter 3, fig 3.8). When this is done, the PC's memory is increased to 640kbytes. A "MS-DOS memory" link on the Local Bus Board (2/1/34) defines whether this facility is enabled (link FITTED = disabled). If the link is not fitted, then a 64kword area of ND DRAM (in address range 0M256-0M320) is prohibited to the ND, and can only be accessed by the PC.

PROM

The PROM address range is mapped into the secret memory addresses 8M000-8M032. It provides for up to 32kx16-bits of storage. The current requirement in ND-110PCX is for only 8k of PROM store, so only the first 8k is fitted at present. This is shown in the schematic 2/1/F3, where the two PROM chips each provide 8kx8-bits of storage.

PROM is used to store the macroprogram for the simulation of the IOX instructions, which are used to handle communication (read and write) between the ND and PC. See Chapters 3 and 6 for further information.

The software system configuration is also stored in PROM, in order to keep system upgrading under the control of Norsk Data. Space is available in PROM for test and other macroprograms.

NV-RAM	<p>The non-volatile random access memory retains its stored data when the machine is switched off. It is shown in 2/4/C1.</p> <p>The address range for NV-RAM (fig 4.18) is 8M032-8M040 in "secret memory" (see section 4.3.5.3). However only the first 2k is fitted at present.</p> <p>NV-RAM is used for parameter storage.</p>
ND-I/O	<p>The ND-I/O address space in secret memory (fig 4.18) is mapped to the HCLC Controller (also known as the Synchronous Comms Controller - SCC) and its associated DMA Controller (2/3/B2). Although 8k of address space is allocated to the ND-I/O function, only a small part of this is currently used in the ND-110PCX.</p>
Secret Memory	<p>The secret memory area (fig 4.18) is 64k long. It consists of 32kwords of FROM in 8k segments (only the first 8k are currently installed), 2kbytes of NVRAM, a 8kword memory-mapped area for ND-I/O, and 32kwords of DRAM.</p> <p>Secret memory is used for ND-PC intercommunication and for ND-I/O with the HDLC/DMA system. It is available for processes associated with the HDLC or IOX simulation functions. This DRAM is part of the 1M of ND-110PCX main memory, but is mapped by the secret flag; it is not available to main memory.</p>

CAUTION

Secret Memory is closely defined by the Butterfly system. No unauthorised user should attempt to use it.

The reserved addresses of the various parameters, programs and flags stored in secret memory are listed at the end of Chapter 6.

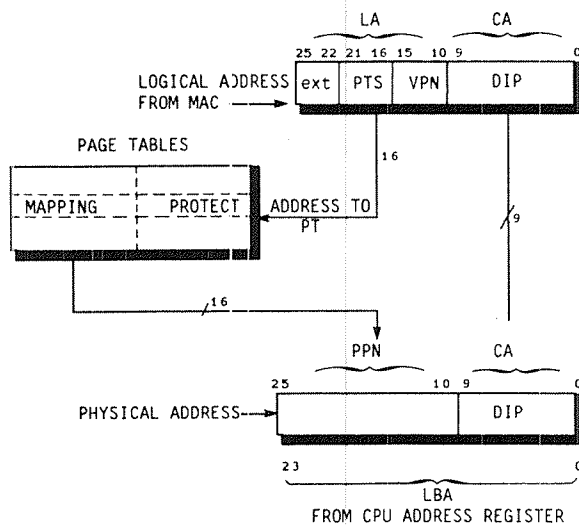
4.3.5.3 Memory addressing

Logical to
Physical Address

The MAC (1/3/A3) provides a 26-bit memory address, as shown in fig 4.19. This address comprises:

- *CA0-9, giving the displacement in a 1k page
- *LA10-15, giving the virtual page number (VPN) address
- *LA16-21, giving the PTS address for the selected page table
- *LA22-25, which provide extension (EXT) address bits, for future expansion.

The 16 bits of VPN, PTS and EXT are latched into the Logical Address Register (1/3/C1), while the *CA0-9 lines are latched into the Cache Address Register (1/3/B4).



DIP: displacement in 1k page
 VPN: virtual page number
 PTS: page table select
 ext: extension address bits
 (not used by MMS)

} page table address

Fig 4.19: Logical-to-physical address mapping

The Page Tables map virtual page addresses to physical page addresses, together with their associated page protect information.

Page Table memory, comprising Map and Protect areas (fig 4.20) is shown in 1/3/D1,E1,F1. For further information on the functional operation of the Memory Management System in relation to fig 4.20, refer to the ND-110 Functional Description, ND-06.026.

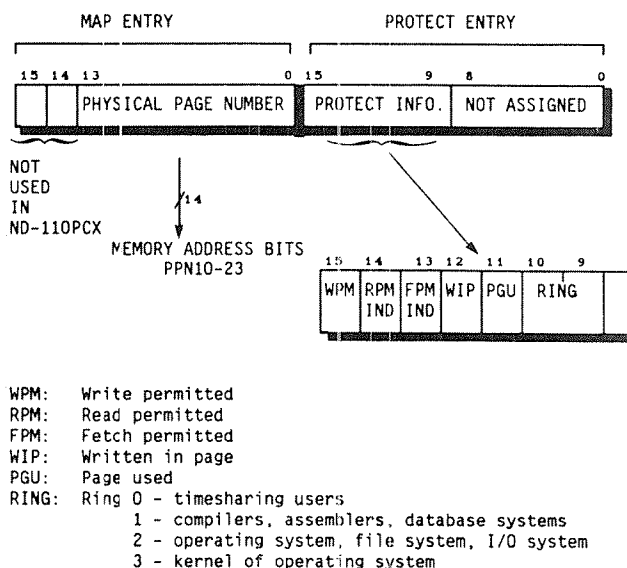


Fig 4.20: Layout of each entry in the Page Table

The ND-110PCX Page Tables provide the same levels of functionality as those in the ND-110. They use the same circuitry as the ND-110CX. On a memory access, the virtual page number address to the selected page table accesses the required 14-bit physical page number from the Map memory area. This is output on PPN10-23. The corresponding Protect information for that page is similarly output onto the PT0-15 lines. The MMU Control PAL (1/3/A1) provides the necessary enable signals (EPMAP, EPT) for Page Table memory.

The PPN 10-23 and CA0-9 lines are latched into the CPU Address Register (1/4/A4), giving the required 24-bit memory address, on LEA0-23.

During normal operation, The SINTRAN operating system sets up the required values in the Page Map and Protect Tables. The mechanism for reading and writing to the Page Table memory may be appreciated from the MMU Control PAL decodes (1/3/B2):

- For the whole Page Table:
 - EPT enables the memory devices
 - WMAP controls read/write with this memory.
- For the Map part, to enable data transfer between IDB and PPN10-25 lines:
 - EIPU enables the upper byte of data (1/3/E3), while EIPL enables the lower byte(1/3/E4)
 - ESTOF controls the direction of data flow, read being from PPN via IDB and FIDB to the ALU, and write being the opposite.
- For the Protect Table:
 - EPTI enables data transfer between the PT lines and the IDB (1/3/E4).
 - WRITE selects the direction of data flow.

For direct addressing (paging off), LAPA (1/3/D5) enables the logical address directly onto the physical address (PPN) lines.

Update ALU During Jump

When a jump instruction is encountered, the P register (in ALU) must be updated to the new value. This operation is performed through the Update-ALU-During-Jump logic (1/3/C3). Provided the address is not in Shadow memory area, microcode decode JMPA enables the jump address onto the FIDB lines, from which it is written into the P register in the Working Register File of the ALU.

Paging Status Register

Whenever the memory system reports errors (page fault, protect violations), the operating system is alerted through an internal interrupt (trap). The system then reads the Paging Status Register (1/3/E5) for further information. This information includes the virtual page address (LA10-25) at which the error occurred.

Address Decoder

The Address Decoder PAL (2/2/A2) examines all memory addresses presented on the LBA bus, in 8k address blocks (i.e. LBA13-23). If the address does not fall within the address ranges specified in the Memory Map (see fig 4.18), it is judged illegal, and INVALID is output from the PAL to the Invalid Address logic (see section 4.3.5.5). Note that this includes the situation where, if the "disable DOS memory"

link is not fitted, ENDOS active to the Decoder PAL causes INVAL to be generated if the ND attempts to address this area of DRAM, which has been reserved for exclusive use by the PC (see MS-DOS Offset Register, below).

Referring to the Memory Map in fig 4.18, the Address Decoder PAL generates active outputs as follows:

- If the address is in the ranges

0M000 - 0M240
0M256 - 1M000
8M048 - 8M064

then DRAM goes active, enabling the DRAM data buffers (2/2/E,F3). Because the address range 0M240 - 0M256 is reserved as part of secret memory, it is not available as legal addresses on the LBA bus.

The appropriate row address strobe for the addressed 256k segment is also generated:

0M000 - 0M240, 8M048 - 8M064:	RAS0
0M256 - 0M512	RAS1
0M512 - 0M768	RAS2
0M768 - 1M000	RAS3

- If the address is 8M000 - 8M032, then PROM goes active, enabling PROM (2/1/F,G3).
- If the address is 8M032 - 8M040, then STAT goes active, enabling NVRAM (2/4/C2).
- If the address is 8M040 - 8M048 and it is a CPU request, then IOMM goes active, enabling memory operations to read and write data with the HDLC/DMA control logic (2/3/B3).
- If the address is in the range 0M256-0M320, and ENDOS (from the "MS-DOS memory" link shown at 2/1/B4) is active, INVAL is generated from the Address Decoder PAL (2/2/A2), so inhibiting the ND from accessing this area of DRAM. See MS-DOS Offset Register, below.

Secret Memory

The secret memory (fig 4.13) starts at Memory Management address 8M000. Addresses 3M000 and above are specified by LBA23 (fig 4.21). This condition is asserted as follows:

- CPU Access:

It is accessed from the CPU with paging off and secret memory flag on. This is set and reset by bit 15 (SECRET) of the I/O Control Register (1/1/E2), loaded by microcode command decode SIOC. On the CPU Address Latch (1/4/A3), SECRET active forces address bit 23 active.

- PC Access:

From the PC it is accessed by setting bit-23 in the PC Offset Register (see below).

- ND-I/O Access:

The HDLC accesses it by specifying an address with LBA23 active.

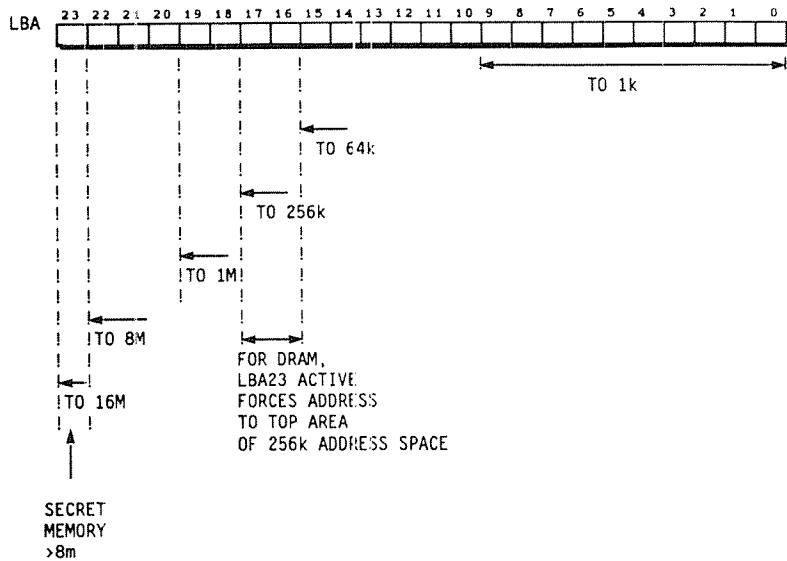


Fig 4.21: LBA bus addressing for secret memory

For PROM, NVRAM and ND-I/O Bus operations with memory, detecting LBA23 active is an essential part of the control exercised by the Address Decode PAL (2/2/A2) in generating PROM, STAT and IOMM respectively.

For DRAM, LBA23 forces address lines 16 and 17 active (2/2/A1 and fig 4.21) through the Memory Address Mux, to the DRAM chips. In this way, the hardware maps the address to the top part of the first 256k segment.

Autoload
Descriptor

The autoload descriptor in ND-110PCX is not a switch, as in other ND-100 derivatives. Instead it is held as a parameter in NV-RAM. Its address is 8M032 (ND address 40100001, PC address D0002H with offset 81H). The ALD is loaded and changed by the Butterfly Supervisor program, System Parameter Definition menu (see Chapter 7). It is accessed by microcode as required.

PC Offset
Register

The PC may access any part of ND-memory. It addresses the ND-memory in 64k byte blocks, by mapping through a 64kbyte PC address window, in the range D0000-DFFFFH. This is decoded by the PC I/F Control PAL (2/1/A4), which generates DISWHI/DISWLO and EPCHI/EPCL0, for high/low byte control of memory operations. It also sends ALEX active, which generates PCREQ (2/1/C4), to request an ND memory cycle.

PC addressing of ND-memory is shown in fig 4.22. The PC specifies the required 64k block, by loading the starting address of the block into a General Offset Register (2/1/C1). This register is loaded by EGOFF, in a PC I/O write operation (see section 4.3.7.3). As indicated in fig 4.22, 64k blocks are addressed on LBA15-21,23, with LBA23 equating to the "secret memory" bit. PCA1-15 are translated to LBA0-14 (2/1/A1), to specify an address within the 64k offset. The least significant PC address bit PCA0 specifies the high or low byte of the addressed location, through the PC I/F Control PAL.

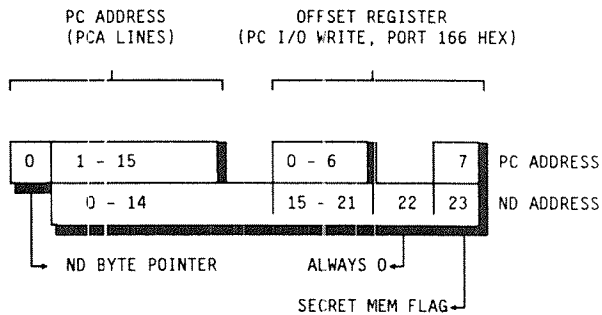


Fig 4.22: PC addressing of ND memory area

MS-DOS Offset Register

If the "disable DOS memory" link is not fitted, ENDOS active reserves 128kbytes of ND-DRAM (in the range 0M256-0M320) as two extra 64kbyte PC RAM blocks 8 and 9 (i.e. PC addresses 512-640k - see fig 4.23) for exclusive use by the PC.

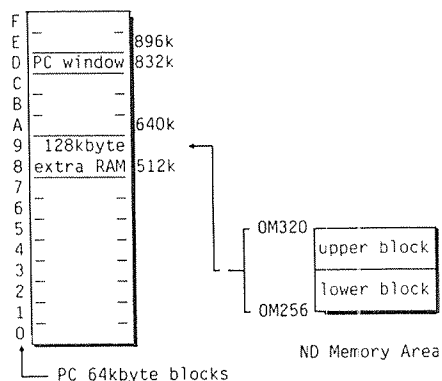


Fig 4.23: Donation of 128kbytes of ND-DRAM to the PC

In a similar way to the PC Offset Register, the MS-DOS Offset Register (2/1/C1) is loaded with the required 64kbyte block address, by ED0FF in a PC I/O write operation see section 4.3.7.3). Address bit PCA16 to the MS-DCS Offset Register specifies whether the PC requires access to the lower 64kbyte block or the upper 64kbyte block (fig 4.23).

Note that all timings in fig 4.23 are in nsec, and do not take account of the nominal 10nsec propagation delay for DELOUT through the Memory Start PAL.

4.3.5.4 Read/write/refresh cycles

Memory Access

The CPU, PC, and HDLC/DMA make requests for access to the memory system. In addition, DRAM memory requires a refresh cycle at regular intervals, in order to re-write the data which is dynamically stored within the DRAM devices, and therefore the Refresh control logic makes regular requests for memory access.

These four sources of request for memory access are presented to the Bus Arbiter PAL (2/2/B4).

- ND CPU

MCRQ goes active (2/2/D5) from the LB Control PAL.

- PC

PCRQ goes active. This is OSC-synchronised MPCREQ (2/2/D5) from PCREQ (2/1/C4). PCRQ is sent active by ALEX (from the PC I/F Control PAL (2/1/B4), which is the decode of a PC request to access the ND's address space. ALEX also sends IOCHRCY inactive to the PC; then, at the end of the PC access, PCGNT and MDRY both active, return IOCHRDY active.

- ND-I/O

IORQ goes active. This is OSC-synchronised MIOREQ (2/2/D5) from IOREQ (2/3/C5), which is sent active by DMAACK, when the DMA Controller sends its BREQ (pin 3) output active (input to CPU) CRQI (2/3/D3) from the I/O Control Register (2/4/E3 - see section 4.3.4.9) is used by the CPU to gain access to the IOB, and disable I/O requests to the Local Bus after termination of the current I/O access.

- Refresh

RRQ goes active. This is OSC-synchronised REFRQ. It is generated by RFCLK (2/4/B2) every 12.8µsec - see section 4.3.2 - which generates delayed refresh clock MRFCLK (2/2/D5), then RFC (2/2/A4), which sets REFRQ active. REFRQ remains active until the next Refresh cycle, when REF clears it.

If a simultaneous request (from 2 or more sources) for memory access occurs, the Bus Arbitor PAL selects the basic priority as (highest to lowest):

CPU
PC
ND-I/O
Refresh.

In addition, it operates a "round-robin" arbitration, so that no single requesting source may monopolise memory to the exclusion of other sources. Further, if a Refresh request is delayed for more than half of a RFCLK period, the Refresh becomes the highest priority.

Memory Grant

If instant memory access is not possible, a wait state is asserted from the Bus Arbiter (2/2/B4) to the requesting source. When the access starts, the wait is removed and grant is asserted. Memory available is indicated by DDRY active to the Bus Arbiter, from the Request Synchroniser (2/2/A4). This in turn is generated from MDRY, which is a synchronised BDRY (2/2/A5), which is in turn sent active at the end of each memory cycle, by BUSRDY (2/2/C3). These synchronising clock stages are introduced to meet the timing requirements of the system. Provided the memory is available, the Bus Arbitor PAL sends MSTART active (2/2/B4) to initiate a memory cycle. Signal MSTART is OSC-synchronised to give DSTART (2/2/A4), which in turn gives DDSTART (2/2/D5), in turn generating TSTART. Signals MSTART, DSTART and TSTART are applied to the Memory Start PAL (2/2/B4), which initiates a memory timing sequence - see Memory Cycle, below.

- For a ND CPU Request:

SCACT is asserted. This enables CPU memory address from the Address Latch (1/4/34,B5) onto the LBA lines, and also connects the LBD lines (memory data) with the cache data (CD) lines (1/4/A1,A2).

- For a PC Request:

PCGNT is asserted. This enables PC Address lines PCA(0-14) onto the LBA lines (2/1/A1,2). PCGNT also enables the PC General Offset (EGOFF, 2/1/C1) or MS-DOS Offset (2/1/D1) address onto LBA15-23, according to whether PC address bit PCA18 (2/1/D4) specifies access through the PC address window or to the extra 128kbytes of MS-DOS memory that may be donated from ND-DRAM (see fig 4.23 and section 4.3.7.3).

- For a ND-I/O (HDLC) Request:

IOBGNT is asserted. This enables the address lines on IOA and IOB buses from DMA Controller, onto the LBA lines (2/3/F3).

- For a Refresh Request:

REF is asserted. This sends all the CAS0-3 lines active (2/2/D4), and selects the "D0" inputs to the RAS Driver (2/2/B2), giving RAS0-3 all active when TIMEPULS enables the Driver. REF also resets the REFRQ latch (2/2/A4), so clearing the currently pending request for a refresh cycle. Via the Request Synchroniser, this removes the RRQ active request to the Bus Arbitor PAL.

Memory Cycle

The memory cycle has been designed to be the same for PC, CPU and Refresh cycles. The ND-I/O (HDLC) cycle has a delayed start, generated by CACK (2/3/E4). This allows for the timing requirements of the DMA Controller.

Signal TIMEPULS from the Memory Start PAL (2/2/B4) initiates the memory cycle timing sequence.

If PROM is being addressed, signal PROM from the Address Decoder (2/2/B1) enables the PROM chips (2/1/F3,G3), giving 16-bit data out from the address specified by LBA0-12, onto the LBD0-15 lines.

If NV-RAM is being accessed, signal STAT from the Address Decoder (2/2/B1) enables the Static RAM (2/4/C2), for a byte read or write operation.

Timing for memory cycles is shown in fig 4.24. Signal TIMEPULS is propagated through the delay line:

- For DRAM, EHADR high enables the low order address bits through the Memory Low Address multiplexer (2/2/C2), onto MADR0-9, and RAS0-3 going low when TIMEPULS goes active (fig 4.24) latches this into DRAM. After 15ns, EHADR goes low, enabling the high order address bits through the Memory High Address multiplexer (2/2/C1), onto MADR0-9, and CAS going low (fig 4.24) then latches this into DRAM.
- For a write operation to NV-RAM, after 60ns, T60 active (2/2/C3) with DWEL0 active generates SWEL0 (2/2/D3), and so writes LBDO-7 into NVRAM (2/4/C2).

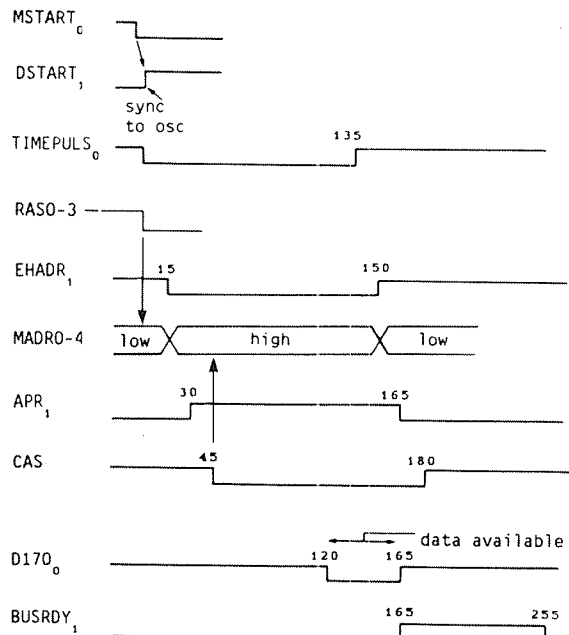


Fig 4.24: Memory timing

- After 120ns, D170 is generated (fig 4.24). For I/O operations, this generates the LIOA strobe (2/3/E5) to the I/O Data Transceivers (2/3/G1).

NO-I/O

For a memory-write operation, DLBTIO not active from the I/O Control PAL (2/3/A5) transfers the IOB bus onto the LBD bus, and LBW active from the I/O Address Latch (2/3/F1) defines a memory write.

For an I/O read from memory, memory data is latched by LIOA going high, into the Data Transceivers, and EIOBLO active enables this data onto the IOB bus.

CPU

For a CPU write to memory operation, DLBTORB (2/2/E5) is inactive, so that write data latched from CDO-15 into the CPU Data Transceivers (1/4/A2) is transferred onto the LBDO-15 lines. Also LBW active from the CPU Address Latch (1/4/B3) defines a memory write. Thus, with DISWHI and DISWLO both non-active from the PC I/F Control PAL (2/1/B4), this generates memory write signals WEHI and WELO high (2/2/D3). These signals pass LBDO-15 through the DRAM buffers (2/2/E3) to the memory devices, and effect a DRAM write operation.

For a CPU read from memory operation, DLBTORB is active, so that DRAM data on the LBDO-15 lines (2/2/E2) is latched by LRIDB into the CPU Data Transceivers (1/4/A2) and enabled through to the CDO-15 lines.

PC

For a PC read from memory operation, LPCBD (2/1/E2) latches the read byte from the LDB bus into the appropriate PC Data Transceiver (2/1/F1,G1). The selection of transceiver is determined by EPCL0 or EPCH1 from the PC I/F Control PAL (2/1/B4), which in turn is defined by the PC address bit 0:

odd = high byte; even = low byte.

For a PC write to memory operation, SLBTPC non-active (2/1/E3) transfers the PCDO-7 lines (2/1/E3) through the appropriate (high byte or low byte) PC Data Transceiver (2/1/E1) onto the LBD bus, and LBW active from the PC Address Buffer (2/1/B2) defines a memory write. Then, for writing to DRAM, depending on whether the high or low byte is to be written, DISWLO

or DISWHI low from the PC I/F Control PAL (2/1/B4) masks write pulse DWEL0 or DWEHI in the Write Pulses gating (2/2/D3). In this way, only the selected byte is written into the appropriate half of the 16-bit DRAM address.

For all write operations to DRAM, WEHI and WEL0 both non-active transfers data from the LBD bus onto the bi-directional DRAM data lines (2/2/E3). Conversely, on read operations, data from DRAM is passed onto the LBD bus by both these signals being active. The signals WEHI and WEL0 control writing of high or low bytes to DRAM, for byte transfers with ND-I/O (HDLC) and with the PC.

- After 135ns, DELOUT applied to the Memory Start PAL returns TIMEPULS high. This transition is then propagated through the delay line.

TIMEPULS going high also sends BUSRDY high (fig 4.24). This sends BRDY active (2/2/A5), which via the Request Synchroniser sends MDRY active, signifying the end of the memory cycle. As a result:

- for CPU-initiated requests, when timing constraints have been satisfied, the LB Control PAL (2/2/E4) returns CRQ non-active.
- for PC-initiated requests, MDRY returns PCREQ non-active (2/1/C4). This also sets IOCHRDY to the PC to signify end of cycle. While IOCHRDY is not set, the PC's 80286 inserts wait states.
- for ND-I/O-initiated requests, BDRY returns IOREQ non-active (2/3/C4).
- Refresh-initiated requests were reset by REF (2/2/A4) as soon as the Bus Arbiter granted the Refresh cycle.
- After 165ns, D170 returns non-active.
- After 225ns, BUSRDY returns low.

Ready Status of NV-RAM

The NV-RAM has a long write time, though the read cycle is standard. Data is latched into it by a normal memory write, but the entire write operation internally to the chip takes typically 4ms (max 10ms).

When a write operation occurs, the READY line from the chip (2/4/C1) goes low, and stays low until the end of the internal activity. This signal must be read prior to a write, and checked on a read. This is accomplished by reading the device, then checking LBD bit 15, which is the ready status. The data is in the lower byte (LB00-7).

Thus on a read cycle the data is valid if bit 15 is high. For a write operation, LBD bit 15 must first be checked using a read cycle to establish that the device is ready.

Single strike
write to NV-RAM

In addition to the write checking noted above, a "write gate" must be opened each time a write to NV-RAM is required from the PC. This exists to help prevent unauthorised write operations to this "special" area of the memory, which has a limited life (nominally 10000 write cycles to any one location). There is no similar gate from the CPU.

The gate is opened by a PC write operation to location NVOPEN (port 162H), which is decoded by the PC I/O Address Decode (2/1/A3), and generates STRIKE active to the PC I/F Control PAL (2/1/B4). It is automatically closed after the next PC access, by MDRY generating CLOSE (2/1/C4).

4.3.5.5 Parity and error checking

Parity in DRAM

The DRAM memory system has 1 bit of parity for each byte. On writing to DRAM, the LBD bus inputs to the parity generation & checking logic (2/2/G1,G2). Parity bits PARHI AND PARLO are generated for each byte, and the EVEN parity bit for each byte is stored in DRAM as bit D8.

On reading from DRAM, the parity bits are output on bit Q8, as PHI and PL0. These are applied to the Parity chips. If the parity of a byte (including its even parity bit) contains no error, then the logical opposite (odd parity) of the regenerated parity bit, will be high. Conversely, if a parity error is detected, it will be low.

For a read from DRAM, if a parity error is detected, ERR (2/2/G2) goes active.

For the ND CPU system, ERR active when read data is ready sends PARERR active from the LB Control PAL (2/2/E4). PARERR is input to the Interrupt Detect logic (2/4/D2) - see section 4.3.6. Also, SPEA and SPES are generated from the LB Control PAL (2/2/E4). These strobe the memory address and relevant parity status bits into respective registers (2/4/C3 and 1/4/C3), for subsequent examination by the CPU (using microcode "IDB source" decodes EPEA and EPES).

For the PC system, if ERR goes active while a PC memory read is in progress, the LB Control PAL sends PCERP active (2/2/E4). Then, when the PC de-asserts its read command line, T60 returning high after a nominal 195ns into the memory cycle, sets the PC Parity Error flip-flop (2/1/C3). This returns IOCHCHK active to the PC, to indicate a PC-read parity error is detected. The Error flip-flop is reset by MEMW on the next write operation to ND memory.

Error Correction Control Register An ECC system is implemented on the ND-110 (RASK) memory system. This is not implemented in the ND-110PCX.

Invalid Address If an address value is not within the ranges specified for ND-110PCX memory (see Memory Map fig 4.18 and section 4.3.5.2), the Address Decoder PAL (2/2/A2) sends INVAL active, so that when T60 goes low 60ns into the memory cycle (see section 4.3.5.4), INVALID goes active (2/2/B3). This sends DMOR active on the next basic clock (2/2/D5). If this is not a refresh cycle, this generates interrupt ID9 (memory out of range) to the Interrupt Detect logic (2/4/D2 - see section 4.3.6).

Signal INVALID (2/2/B3) is returned inactive when BUSRDY from the Memory Timing logic (see section 4.3.5.4) goes active for this memory cycle.

4.3.6 Interrupt system

4.3.6.1 Interrupt mechanism

Program levels At any one time, the ND-110 is operating on a given program level between 0 and 15; the priority increases as the program level increases, i.e. level 15 is highest priority. In Butterfly-110, an additional software level 16 (highest priority) also exists, as described in Chapter 3.

- Level 16 is handled exclusively by Butterfly-110 microcode and macrocode.
- "External" interrupt sources are assigned program levels as follows:
 - 10 - terminal output, to async devices
 - 11 - DMA channel, for mass storage devices
 - 12 - terminal input, from async devices
 - 13 - real time clock
- "Internal" interrupts are handled on program level 14.
- Interrupts on levels 0-9 are handled by microcode.

Interrupt control mechanism The ND-110 interrupt control system is illustrated in fig 4.25. It handles all interrupts arriving on program levels 10-15.

The current program level is held in the PIL register.

Each source of interrupt is assigned a bit in a PID register. When a source generates an interrupt, it sets its PID bit on the appropriate CPU program level. If the corresponding PIE bit is also set (= interrupt enabled), this interrupt is input to a priority encoder, which arbitrates between all the enabled interrupt sources, to generate the program level assigned to the highest priority active interrupt. On processing the "interrupt", the microcode "level change check" routine compares the interrupting level with the current level:

- if the interrupting level is higher than the current level, the microcode executes a level change

- otherwise, the CPU continues at its current program level. Then, on completion of the task on the current level, program issues a WAIT, which returns to the level handler, for it to find the next highest level PID/PIE active pair to be serviced.

Assuming the level is enabled and the level change is made, the level handler issues an IDENT instruction to find the source of the interrupt. The action of the IDENT disables the device interrupt enable. Having received the IDENT code, the level handler decodes it and calls the appropriate device driver, which will do some work and (usually) issue some IOX instructions. These disable the source of the interrupt and re-enable the device interrupt enable. The device driver then returns to the level handler and executes a WAIT. This resets the PID bit for that level.

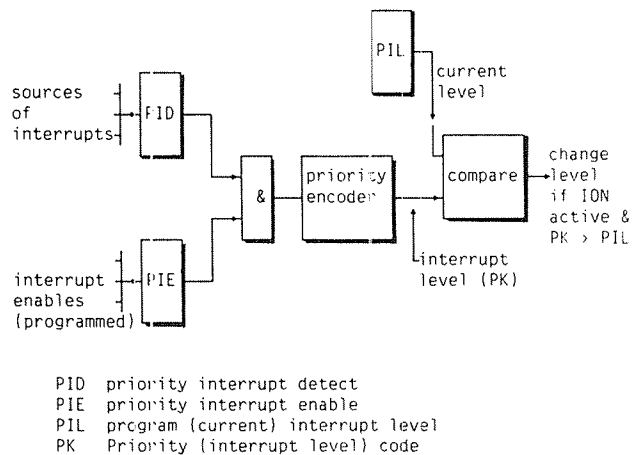


Fig 4.25: Interrupt control system

Butterfly-110 handling of mailbox I/O operations is structured to comply with this 3-stage system: Interrupt handling, then IDENT, then IOX(T).

4.3.6.2 Interrupt controller

The various sources of interrupt to the ND-110PCX are applied to the Interrupt Detect logic (2/4/E1).

In the ND-110CX series, there are two kinds of interrupt, though in ND-110PCX the true difference is lost. One kind is termed "Panel", while the other is "Interrupt". The start of the vector areas for these events are microcode labels PANVC: and ITSRV: respectively. In the following list (which can be read from the schematic 2/4/F1,2), the lowest priority is first, the highest last.

<u>Address</u>	<u>Signal</u>	<u>Description</u>
ITSRV:+0	COMINT	console interrupt (section 4.3.8)
+1	PTONINT	PC interrupt to ND (mailbox)
+2	HDINT2	ND-I/O HDLC (SCC) controller interrupt
+3	HDINT1	ND-I/O DMA controller interrupt
+4	ID5	internal int 5 - ALU "Z" indicator
+5	ID8	internal int 8 - parity error (PARERR)
+6	ID9	internal int 9 - mem out of range (DMOR)
+7	ID10	internal int 10 - power fail
PANVC:+0	RUN	gives "stop" interrupt when false
+1	spare	
+2	RTC	real time clock (20ms)
+3	spare	
+4	spare	
+5	spare	
+6	spare	
+7	MCL	master clear switch

Any active interrupt source generates either IRQ (from the ITSRV PAL) or PAN (from the PANVC PAL). These signals are applied to the Interrupt Request logic (1/2/E1). The PAN interrupt generates PANPAN active provided the panel is not in Stop mode. Either IRQ or PANPAN active set the INTRQ flip flop.

Signals PANPAN and INTRQ are also applied to the Trap Vector Generation logic (1/3/G3), which outputs an absolute vector address to the microcode, for servicing that interrupt - see Traps, section 4.3.4.10. The microcode then generates the appropriate IDB source decode EPICV or EPAN (1/2/F2) from microcode bits 37-41 (IDBS0-4, see Chapter 5), to obtain from the Interrupt PALs (2/4/F1,F2) the identity of the highest priority interrupting source, and process it accordingly.

The functions of the PID, PIE and priority encoder are all performed by the microcode. All interrupts except those from HDLC/DMA and PC-to-ND, are handled solely by the "CPU" microcode. The HDLC/DMA interrupts are handled by the "HDLC" microcode, while the PC-to-ND interrupt is handled by the "mailbox" software on "level 16".

4.3.6.3 Signal box interrupts

There are two interrupts between the PC and the ND, one in each direction. Both interrupts are set by one end and reset by the opposite end. This is illustrated in fig 4.26.

Interrupts between the PC and the ND are handled using a "signal box" system. The functions performed by this facility are described in Chapter 6 (section 6.2.4).

From the hardware viewpoint, a PC interrupt request to the ND (PTONINT) originates from the PC, by the PC sending PSPTON active (fig 4.26). The ND processes the interrupt, then resets the interrupt line (by sending NRPTON active). However, a further interrupt may have occurred between the original interrupt and the point when the ND's interrupt handler resets it. To avoid missing this possibility, after resetting the interrupt line, the signal box handler checks the interrupt set of flags again. If it finds any active, it generates NSPTON active, to re-establish the interrupt condition, and process it.

A similar function is performed for ND to PC interrupt requests, as shown in fig 4.26.

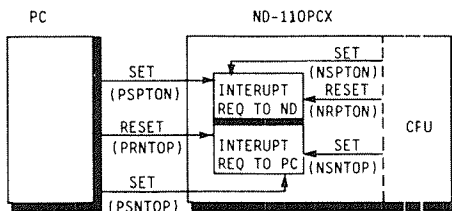


Fig 4.26: ND-PC interrupt request control

The PC exercises control by writing the relevant bit to the Keys Register (2/1/E1), to generate PRNTOP, PSNTOP and PSPTON. Command LKEY from the PC-I/O logic (2/1/D3) strobes this into the Keys Register. This decoder is described in section 4.3.7.3.

The ND exercises similar control, by issuing microcode command decodes NRPTON, NSPTON and NSNTOP (1/2/C1).

For a PC-to-ND interrupt, PSPTON generates PTONINT (2/1/D3), to the Interrupt Detect logic (2/4/E1), and to the ND Status Register (2/4/D5) and PC Status Buffer (2/1/E1). On servicing this interrupt, the microcode generates command decode NRPTON (1/2/C1), to return PTONINT non-active (2/1/C1). If necessary, NSPTON re-establishes the interrupt, as described above.

For a ND-to-PC interrupt, the microcode generates NSNTOP (1/2/C1). This generates NTOPINT to the ND Status Register (2/4/D5) and PC Status Buffer (2/1/E1), and sends IRQ5 active to the PC (2/1/D2). When the PC services this interrupt, it generates PRNTOP (2/1/C2), which returns NTOPINT and IRQ5 non-active. If necessary, PSNTOP re-establishes the interrupt, as described above.

4.3.6.4 Internal interrupts

Internal interrupts are a mechanism whereby microcode and internal hardware events can be reported to the macrocode (via IIC interrupting on level 14), for service.

Control mechanism The functions of the ND-110's internal interrupt detect (IID) and internal interrupt enable (IIE) registers and associated processing (see fig 4.27) are all performed in the Butterfly-110 microcode.

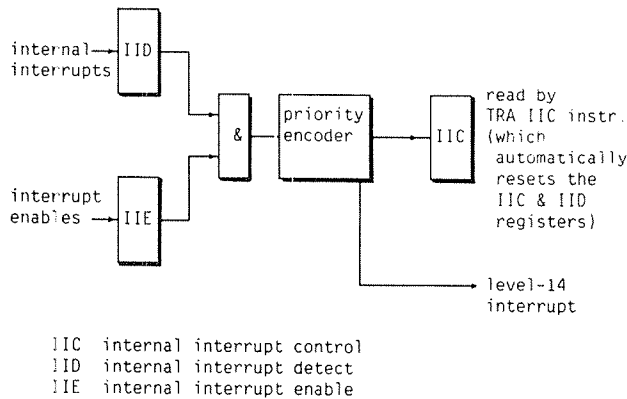


Fig 4.27: Interrupt control system

Interrupt conditions

The main additions to the internal interrupt sources are:

- in the detection of IOX errors, to cater for the presence of Butterfly-110's IOX simulators (mailbox devices) and to check with the Installation PROM that the Butterfly machine is authorised to operate with specific devices
- to the Power Fail interrupt, which is also used to respond to a PC power fail (PCFAIL), and to a microcode watchdog time-out which senses if the PC fails to respond to an interrupt from the ND.

The internal interrupt conditions and their associated vectors are listed below:

Interrupt	Code (octal)	Source
-		
MC	1	monitor call
MPV	2	memory protect violation
PF	3	page fault
II	4	illegal instruction
Z ¹	5	error flag
PI	6	priveleged instruction
IOX	7	IOX error
PTY ²	10	parity error
MOR ³	11	memory out of range
POW ⁴	12	power fail interrupt

¹ ALU "Z" indicator :	ID5	} see also section 4.3.6.2 and description below.
² PTY =parity error PARERR :	ID8	
³ MOR =mem out of range DMOR:	ID9	
⁴ POW =power fail :	ID10	

MC Monitor call

The MON instruction may have up to 255 different codes (bits 0-7 of the MON instruction). The T-register on level 14 is loaded with this code, for use by the interrupt handler.

MPV Memory protect violation

Two types of violation are possible:

- memory protect - an illegal reference (read, write, fetch or indirect) has been attempted,
- ring violation - a program attempted to access an area having a higher ring status.

The interrupt is generated by the Trap mechanism (see section 4.3.4.10). It may interrupt within a macroinstruction. The paging system must be turned on (PON instruction) to receive this interrupt. Information regarding the cause of this interrupt is contained in the paging status register.

PF Page fault

This is generated if program attempts to reference a page that is not in memory. The interrupt is generated by the Trap mechanism (see section 4.3.4.10). It may interrupt within a macroinstruction. The paging system must be turned on (PON instruction) to receive this interrupt. Information regarding the cause of this interrupt is contained in the paging status register.

II Illegal instruction

This is caused by an attempt to execute an instruction that is not implemented. It is detected by the microcode.

Z Error flag

This is generated if the Z flag in the STS register is set. This may be caused by several conditions:

- floating point divide by zero
 - attempt to EXR an EXR instruction
 - DNZ overflow
 - RDIV overflow
 - Z-flag set by program (BSET, MST, TRR).
- } Z-flag set by microcode

Note: the level-14 interrupt routine must always reset the Z-flag on the interrupting level, otherwise a new interrupt will occur when that level is re-entered.

PI Privileged instruction

This arises when there is an attempt to execute a privileged instruction from ring 1 or 0. It is detected by the microcode. The paging system must be turned on (PON instruction) to receive this interrupt.

IOX IOX error

This is caused if an IOX instruction addresses an input/output device that does not exist, or is not configured as an authorised device in the Installation PROM. It is also generated if no device answers to an IDENT instruction. These conditions are detected by either the IOX-simulator code on level-16, or the microcode.

PTY Memory parity error

This occurs if an ND DRAM parity error has been detected (PARERR), in which case the PEA and PES registers contain information about the failure condition (see section 4.3.5.5). The contents of these two registers are locked in until the PEA register has been read (TRA PEA instruction), so to protect the error information, the PES register should be read first (TRA PES).

MOR Memory out of range

This arises if program attempts to access a memory address that does not exist - see "invalid address", section 4.3.5.5. As for the MOR interrupt, the PEA and PES registers contain information about the failure condition (see section 4.3.5.5). The contents of these two registers are locked in until the PEA register has been read (TRA PEA instruction), so to protect the error information, the PES register should be read first (TRA PES).

POW Power fail

This interrupt is triggered by any one of three sources:

- the power sense circuit has detected an ND power fail condition (see section 4.3.10)
- the PC has issued PCFAIL (see section 4.3.10)
- the microcode watchdog timer for the signal box (see Chapter 3 section 3.7.1.2) has timed-out. The microcode monitors the PC activity in response to ND-to-PC interrupts, and generates a power fail interrupt if the PC does not respond within the time-out period - this may be because the PC has stopped accepting interrupts, or has had a soft or hard reset, or is in a "hung" condition.

4.3.7 PC-ND I/O system

4.3.7.1 PC Expansion Bus

The PC Expansion Bus is a 20-bit address and 8-bit data bus. It has the capability for I/O (PCIOR, PCIOW) or memory (PCMEMR, PCMEMW) transfers. The interface provided in the ND-110PCX is such that the PC can write to all of the ND's address space, and has automatic 8-16 multiplexing and de-multiplexing between the 16-bit ND data bus and 8-bit PC data bus. PC bus error reporting is also provided.

A summary specification for the PC Expansion Bus is provided in Appendix C.

4.3.7.2 PC access to ND memory

The PC can directly access a 64kbyte block of ND address space. This area is pointed to by the PC General Offset Register (2/1/C1), which is loaded by the PC using a program I/O operation to the PC's I/O port 165H. See section 4.3.5.3.

Read and write operations with ND memory are described in section 4.3.5.4.

Error reporting for PC accesses to ND memory is described in section 4.3.4.5.

4.3.7.3 PC program I/O

In the ND-110PCX, a number of registers are accessible, and PC-related control signals generated, from the PC. These are located in the PC's I/O address space, in the range 160-167H. This range is decoded by the PC I/O PAL (2/1/B3), from PCA0-9 and the PC control lines PCIOW (write), PCIOR (read), and PCAEN (DMA). If a valid I/O operation is decoded, output ENIO active enables the I/O Address Decoder (2/1/A3), which demultiplexes the PCA0-2 lines.

As a result, the following PC I/O operations are decoded:

<u>Address (Hex)</u>	<u>Description</u>	<u>write /read</u>
160	reserved	
161	control: simulate power fail (PCSIMF)	W
162	control: open write gate to NV-RAM (NVOPEN)	W
163	control: ND master clear (PCMCL)	W
164	reserved	
165	data: load general offset (GOOFF)	R/W
166	data: read status/write keys (GOST)	R/W
167	control: release ND (PCLETGO)	

A control signal requires no data. The other locations effect a read or write operation over the PC I/O bus.

Control decodes	Either of PCSIMF or RSTDRV (PC reset) send PCFAIL active (2/1/B2), which is used by the Power-Fail logic (see section 4.3.10). The function of NVOPEN is described at the end of section 4.3.5.4. PCMCL is applied to the Clear logic (see section 4.3.11).
Load General Offset address	Decode GOOFF, gated with PCIOR (2/1/D3), generates LOFF or EOFF, to write to or read from the General Offset Register. For PC I/O write operations, this stores the 64k offset value of the address the PC wishes to access in ND memory. Read operations are necessary to cater for situations when the PC is interrupted, and needs to save the current offset, so that it can return later.
Load MS-DOS Offset register	Whenever the PC is granted a memory access, either through the PC window or to the extra MS-DOS memory, ALEX from the PC I/F Control PAL (2/1/B4) goes active; this loads the MS-DOS Offset register (2/1/C1), which defines whether the upper or lower half of this extra 128kbytes is accessed. See also fig 4.23.
Selecting General or MS-DOS Offset	Address line PCA18 specifies whether the access is through the PC window or to the extra 128kbytes of MS-DOS memory that may be donated from ND DRAM (see section 4.3.5.3: PC Offset and MS-DOS Offset). If a PC memory request is granted (PCGNT active from 2/2/C4), PCA18=1 specifies addressing through the PC window, so EGOFF is generated (2/1/D3) to enable the PC General Offset value onto LBA15-23 (2/1/C1). Alternatively, PCA18=0 identifies an access to the 128kbytes of extra MS-DOS memory donated from ND-DRAM, so EDOFF is generated (2/1/D3), to enable the MS-DOS Offset value on to LBA15-23.

Decode GOST is similarly gated with PCIOR and PCIOR, to generate LKEY for writing Keys data to the Keys Register (2/1/E1), or ENDST for reading the Status Register (2/1/E1). These are described in section 4.3.7.4.

4.3.7.4 ND-PC status and control registers

Status The Status Register (2/1/E1), is enabled onto BPCDO-7 by PC I/O decode ENDST (see section 4.3.7.3). The bits are:

- 0 PC to ND interrupt set (PTONINT)
- 1 ND to PC interrupt set (NTOPIINT)
- 2 not used
- 3 test - microcode internal test facility
- 4 system request (SYSREQ) to change environment (ND/PC)
- 5 ND in stop mode (STOP)
- 6 ND in master clear (MCL)
- 7 ND in power fail (POWFAIL)

Keys The Keys register (2/1/E1) latches control input data from the PC, into the ND. The bits are:

- bit 0 Set PC to ND interrupt (PSPTON)
- bit 1 Set ND to PC interrupt (PSNTOP)
- bit 2 not used
- bit 3 Reset ND to PC interrupt (PRNTOP)
- bit 4 Enable PC write to ND memory (ENWRT)
- bits 5,6 & 7 are not used.

PC I/O decode GOST generates LKEY, to load data from the PC into the Keys Register (see section 4.3.7.3). This register must be set to FFh when no activity is required. Signal ENWRT (Keys register bit 4) is used to gain access to the ND's memory: it must be active for ND/PC communication to be able to work.

4.3.8 ND-I/O system

4.3.8.1 HDLC & DMA

"ND-I/O"

Part of the ND-110PCX secret memory area is mapped to the ND I/O system for HDLC/DMA operations (see section 4.3.5.2 and fig 4.18). The memory operations to read (input) and write (output) with this I/O sub-system are described in sections 4.3.5.3 and 4.3.5.4.

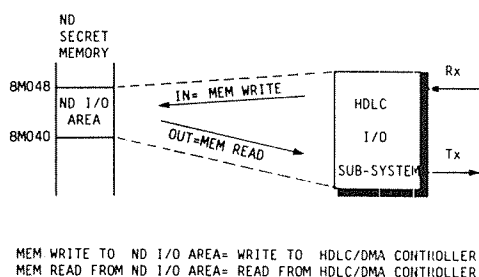


Fig 4.28 illustrates the memory operations involved for programmed I/O (PIO). Essentially, to perform PIO output to the HDLC/DMA sub-system, the ND-110PCX reads from its secret memory area 8M040-8M048 (see section 4.3.5.2). Similarly, to perform PIO input from the HDLC/DMA sub-system, it writes to this ND I/O memory-mapped secret memory area.

Fig 4.28: ND-I/O subsystem

HDLC I/O

The Butterfly HDLC facility offers the following types of communications connections:

- RS232 (V.24)
- RS422 (X.21 / X.27).

This may be used as follows:

- ND-computerlink, up to 153.6kbaud
- circuit switched network connection
- packet switched network connection
- remote job entry (RJE) and emulators.

HDLC controller The HDLC controller chip (2/3/B2) has two separate I/O ports, designated "A" and "B". The A-port is used as the standard I/O port for ND-Computerlink, and the B-port is used for auxilliary V24 signalling and X21 data change detection.

The HDLC controller is set up by the ND writing to secret memory addresses which map to registers inside the HDLC controller chip. Refer to the HDLC Controller specification for an explanation of the internal operations and set-up requirements of this device. For the purposes of this description it is desirable to note that it may be set up to provide:

- the types of communication listed above
- interrupt requests to the ND,
 - for program I/O transfer, using its INT output,
 - for DMA transfer, by sending its W/REQA output active for Rx requests, and DTR/REQA active for Tx requests.

DMA Controller The ND-110PCX sets up the DMA Controller device (2/3/E2) in a similar way to the HDLC Controller. It has two DMA channels, designated 1 and 2 respectively. Channel 1 is used to service HDLC requests for transmit data, and channel 2 is used to service HDLC requests for receive data. In any contention, channel 1 has highest priority. Refer to the DMA Controller specification for a description of the internal operations and set-up requirements of this device.

For the purposes of the description here, it is necessary to appreciate the basic functions involved in performing any DMA operation (see fig 4.29).

- 1 A DMA channel to ND memory is set up in the DMA Controller, defining a starting address in ND memory, together with the number of bytes (byte count) to be transferred, and the direction of transfer:

IN to ND = write to ND memory
OUT from ND = read from ND memory.

- 2 The ND sets up the Controller through its ND-I/O area in secret memory. The actual DMA transfers with ND memory then commence from whatever starting address is specified in this DMA channel (fig 4.29).

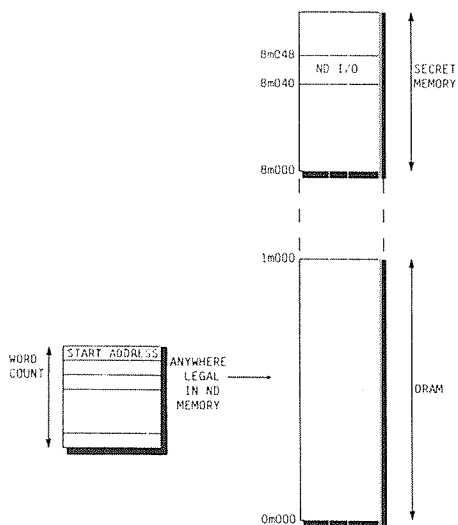


Fig 4.29: ND-I/O - DMA Buffer

- 3 Once a DMA channel is set up, the DMA Controller outputs BREQ when it requires service to transfer a word/byte of data, and is given BACK when the request is acknowledged.
- 4 This continues until the number of transfers specified in the word count have been performed, at which point the DMA Controller sends its INT output active, to signal the end of that DMA channel operation.

Programmed I/O Transfers with HDLC

The HDLC Controller may communicate directly with ND CPU, bypassing the DMA Controller. In this situation, it is set up to interrupt the CPU each time it requires service. Then, each time it requires service, its INT output active generates HDINT2 ((2/3/D1) to the Interrupt Detect logic (2/4/E3), so raising an interrupt request - see section 4.3.6.

The CPU services the HDLC interrupt by making a memory access to the ND-I/O area in secret memory. This sends IOMM active (2/3/B3), so enabling direct I/O operation with the HDLC or DMA controller devices. Signal IOMM is gated with address bit LBA4, which effects alternate "chip enable" selection between the HDLC controller (SCCCE) and DMA controller (DMACS), on every eighth ND-I/O address: ND-I/O address space is segmented accordingly. Also, when M/I/O from the DMA controller goes low, it sends SCCCE active, to enable the HDLC controller for data transmit/receive operation with the DMA controller. The necessary control decodes are derived from the IOB Control PAL (2/3/A5) and associated gating.

DMA Transfers with HDLC

If a DMA channel is set up for an HDLC transfer of a block of data, then the HDLC Controller indicates it requires service by sending a request to the DMA Controller device.

- For output from ND-110PCX (i.e. transmit), the HDLC Controller sends its DTR/REQA line active. This latches a DREQ1 condition to the DMA Controller, so requesting data. The DMA Controller services this request, by performing a DMA read from ND-memory, and transferring the next byte to the HDLC device, for transmission.
- For input to ND-110PCX (i.e. receive), the HDLC Controller sends its W/REQA line active, which sends DREQ2 active to the lower priority of the two DMA Controller channels. The DMA Controller services this request, by taking the received byte from the HDLC Controller and performing a DMA write to ND memory.

When the specified number of words have been transferred, the DMA Controller sends its INT output active. On the next XCLK, this generates HDINT1, which requests an interrupt to the ND-110PCX, so indicating completion of this operation.

4.3.8.2 Console port

The ND-110PCX Console port - also known as the Peripheral Communications control Interface (PCI) - is shown on 1/1/C4 (the controller) and 2/4/G2,G3,G4 (the line Tx and Rx elements).

The microcode commands RPCI and WPCI control ND activity with the Console port. The command operations available are described in Chapter 5. They give rise to command decodes EPCI and RPCI (1/2/C2), which with the microcode bits MISO-1, control the internal operations of the Console port controller.

The clock input to the controller is BRCLK. Console interrupt COMINT from the controller goes active when it is ready to take data for transmission (TXRDY), or to present a byte of received data (RXRDY). Signal COMINT goes active to the Interrupt Detect logic (2/4/E1), to request service from the ND.

For output from the ND-110PCX, 8-bit parallel data on the IDB is transmitted serially from the controller's TXD line, and out (2/4/G3) onto the Console connector, in the prescribed RS232 configuration.

For input, serial data is received via the Console connector (2/4/G2), and converted into 8-bit bytes by the controller. COMINT is generated by the controller when it has a byte ready to be read in by the ND.

4.3.8.3 Panel processor

The panel processor of other ND-100 systems is not available in the ND-110PCX. Calendar information for the ND is obtained from the PC side of Butterfly.

There is no message facility, and related commands are ignored.

The Panel Control Register operates as normal. The Panel Status Register is standard apart from bit 15, which is permanently high. Bit 12 is set after a read, and reset by TRA,PANS.

Internal registers LMP and OPR are standard.

4.3.9 Real-time clock

The Real-Time Clock (2/4/E5) is built around a National MM5368 device. Provided the RTC device is allowed to operate (HOLDRT non-active), the 32kHz crystal clock CLK32 generates a RTC interrupt request to the Interrupt Detect logic every 20ms. The microcode acknowledges this interrupt by issuing command CLRTI.

The HOLDRT control (from the Console connection) is used to stop the RTC, for test purposes. Restart control signal RESRT is generated by microcode command SIOC with bit 3, from the I/O Control Register.

4.3.10 Power fail and stop

The microcode controls start and stop of the ND-110PCX, by issuing microcode commands SSTART and SSTOP respectively. These set/reset the Run flip-flop (2/4/B3).

The Console connector input POWSENSE detects if the ND power falls below the required levels, and PCFAIL non-active indicates that the PC allows the ND to run. If either of these go active, the Power Fail flip-flop (2/4/B3) is set. Signal POWFAIL active sends MINH active to inhibit ND memory after the microprogram has reset the Run flip-flop. Also, if enabled by IE10, its output ID10 generates an interrupt request via the Interrupt Detect logic (2/4/E2).

4.3.11 Clear

A master clear for the ND-110PCX is implemented when MCL (2/4/E4) is sent active.

The ND-110PCX microcode issues a master clear, using command SIOC with bit 0 (2/4/D4), which generates EMCL active, in turn sending MCL active. Also, if either PCMCL (from the PC-AT) or SWMCL (from the Console connector) go active (2/4/A5) they send CLEAR active. This:

- sends STOP active from the Run flip-flop (section 4.3.10)
- sends MCL active (2/4/E4)
- clears the I/O Control Register (2/4/D4).
- generates master reset MR, which resets the MIC, PCI, etc.

4.4 HARDWARE OPTIONS

4.4.1 Serial Communications Board plus Connector Box

The 4-port Serial Communications Controller board plugs into the expansion card cage of the PC-AT. The 1-to-4 serial port connector box mounts on the rear of the PC, and connects to this interface.

For a technical description of these two components, refer to the supplier's technical literature.

4.4.2 Tape streamer

The tape streamer sub-system comprises a QIC-02 interface board and a tape streamer unit. The QIC-02 interface plugs into the expansion card cage of the PC-AT, with a cable connection to the external tape streamer unit.

For a technical description of these two components, refer to the supplier's technical literature.

4.4.3 Additional 1Mbyte RAM

The option to increase the ND-110PCX memory size from 1Mbyte to 2Mbyte, involves fitting the additional DRAM to the Butterfly-110 Local Bus board 3402, and installing the correct configuration of the Installation Number PROM on the Butterfly-110 CPU board 3401. See also section 4.3.4.11.

The technical description given in section 4.3.5 for the ND-110PCX memory applies for both 1Mbyte and 2Mbyte memory sizes.

4.4.4 HDLC communications

The HDLC chip set comprises the HDLC Controller (SCC) and the DMA Controller devices. These must be fitted on the Butterfly Local Bus Board 3402. In addition the correct Installation PROM must be installed, on the Butterfly-110 CPU board 3401. See also section 4.3.4.11.

A technical description for the ND-110PCX HDLC Communications sub-system is given in section 4.3.8.

4.4.5 Plug-in modem

A variety of plug-in modem cards are available, for insertion into the PC-AT expansion bus. Refer to the supplier's literature for descriptions of these modules.

4.4.6 External Winchester disk

The Winchester disk sub-system comprises an interface board and a Winchester disk unit. The interface plugs into the expansion card cage of the PC-AT, with a cable connection to the external Winchester unit.

For a technical description of these two components, refer to the supplier's technical literature.

4.5 ND-110PCX TERMINAL DISPLAY UNITS

For a technical description of these two units, refer to the supplier's technical literature.

4.6 ND-110PCX TERMINAL KEYBOARD

The technical specification for this unit is provided in Appendix F.

4.7 MOUSE

For a technical description of this unit, refer to the supplier's technical literature.

4.8 BAR CODE READER

For a technical description of this unit, refer to the supplier's technical literature.

CHAPTER 5 MICROPROGRAM DESCRIPTION

TABLE OF CONTENTS

Section		Page
5.1	INTRODUCTION	5-5
5.2	HARDWARE ON THE MICROCODE BUS	5-5
5.3	ND-110PCX MICROPROGRAM WORD FORMAT	5-9
5.3.1	Summary of changes from ND-100	5-9
5.3.2	Microinstruction word: field definitions	5-10
5.4	RETURN FROM SIMULATION INSTRUCTION: RTNSIM	5-21
5.4.1	Interface to the simulation routines	5-21
5.4.2	Entry point interface (UPSIM routine)	5-21
5.4.3	Return from simulation (RTNSIM instruction)	5-22
5.5	WRITING MICROPROGRAM	5-23
5.5.1	Summary	5-23
5.5.2	Design of microcode	5-24
5.5.3	Assembly language considerations	5-24
5.5.4	Generating control store PROM	5-26
5.6	MNEMONICS LIST	5-27

LIST OF ILLUSTRATIONS

Figure		Page
5.1	ND-110PCX CPU block diagram	5-6
5.2	ND-110PCX microinstruction word format & functions summary ..	5-10

5.1 INTRODUCTION

This Chapter describes how the microcode controls the operation of the ND-110PCX. It provides:

- a description of the structure of the ND-110PCX microinstruction word format
- an overview of each field in the microinstruction, at a level which is sufficient to understand the range of control operation(s) effected by that field
- an appreciation of the activities involved in writing microcode.

5.2 HARDWARE ON THE MICROCODE BUS

The CPU consists of elements built around the internal data buses IDB, FIDB, CA and CD. These are shown in fig 5.1.

The following description provides a summary of the functional interface between these elements and the microinstruction word.

ALU

- This performs arithmetic and logic functions. Two 4-bit operand pointers, the A and B operands, select sources for operations to be performed. Nine ALU function bits select operations within the ALU. The data input and output ports are directly connected to the FIDB, with the CD bus providing control input data. Floating Arithmetic Control performs the necessary functions to optimise multiply and divide operations.
- It includes Shift Linkage, which controls the shift mode as well as the length of the shift operands.
- It includes the STS register.
- It includes the DBR, which latches data coming from memory via the CD bus, for eventual use on the IDB.
- It includes the IDB byte swapper.

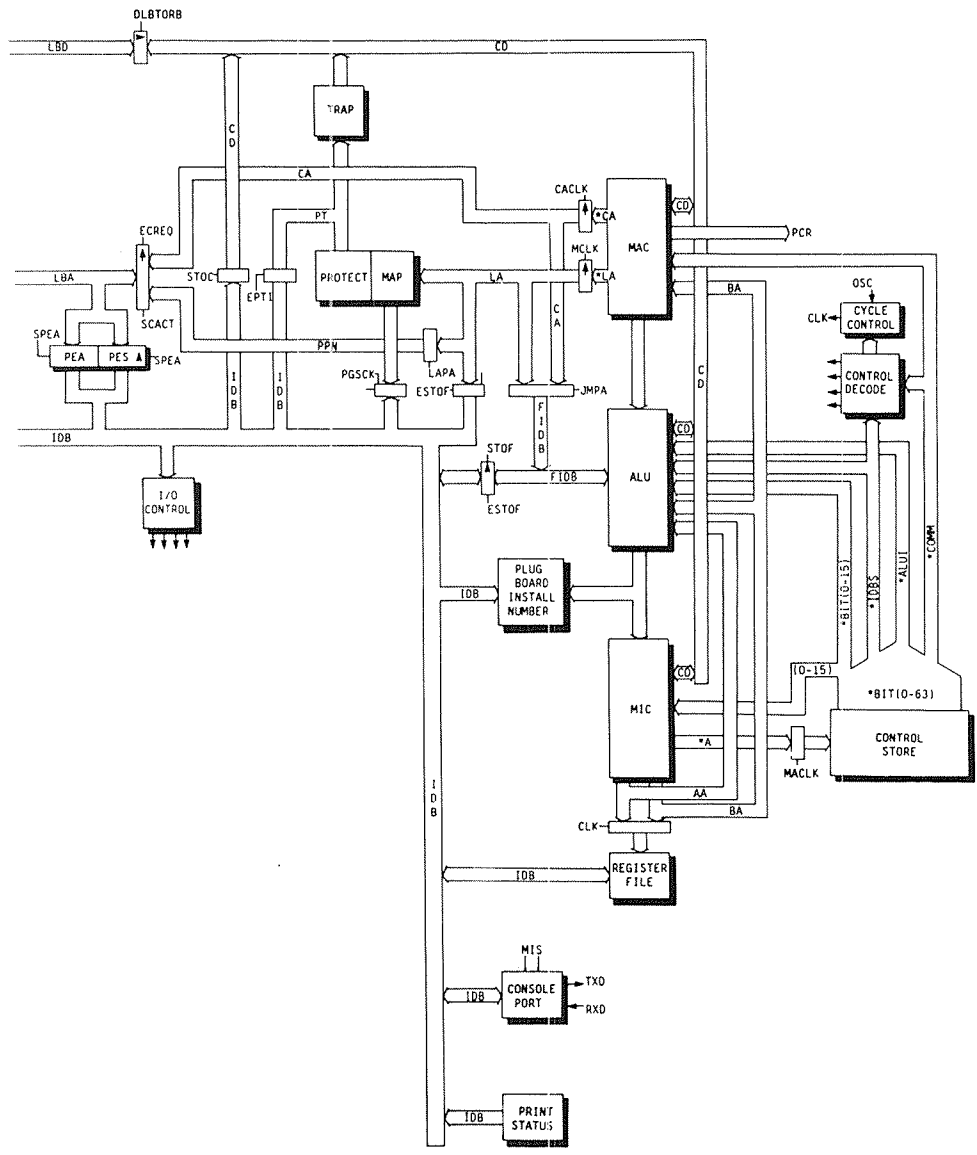


Fig 5.1: ND-110PCX block diagram

- It includes the General Purpose register GPR. This is always shifted the same way as an ALU register if shift is specified in the ALU control. The GPR is especially valuable during multiply and divide operations: in multiply operations it holds one of the operands, and in divide operations it receives the result as a shift input from the right.
- It includes the BMG (Bit Mask Generator), which generates a 1 bit among zeros. The position of the 1 bit is controlled by the A-operand.
- It contains the Register File, which stores the register blocks for the sixteen CPU program levels. Each register block comprises sixteen registers. Eight are used to hold the registers P, B, X, L, T, D, A and STS. The other eight are used by the microprogram as scratch registers, for MOPC (Microprogram Operators Panel Control) and the interrupt system. The Register File is addressed using the A operand to indicate the level, and the B-operand to indicate the register within that level.

MAC

This provides the logical (virtual) 16-bit memory address, to the Memory Management System (MMS). It includes arithmetic operations to speed the computation of memory addresses, avoiding reference to the ALU for such purposes.

PES and PEA registers

These store information on the memory system. They are used to determine status and address data, following detection of an error condition.

Control Store

This is read-only memory. It stores the microcode.

MIC

- This generates the address of the next microinstruction. The address may specify a jump condition, or the current address + 1 (continue), or a return condition (from an internal 4-deep push/pop stack which can nest up to four microprogram interrupt levels).
- It generates the MAP (Microprogram Address Pointer) entry points for all macroinstructions, by adding 6000 to the op code (i.e. the most significant 10 bits) of the instruction itself. This defines the starting point in microprogram for every macroinstruction fetched into the CPU.
- It controls conditional branching in the microcode sequence. A code indicating a test object is asserted from the microprogram, together with a code indicating what to do if a later test gives a false result. When a test is enabled at a later time, the result of the former ALU operation is tested, and may result in conditional branching.

- It controls the source of the lower four bits, in a vectorised jump sequencing microinstruction. The upper eight address bits are specified in the microinstruction (in the microinstruction, the four lower bits must be 0). In this manner a 16-way branch can be implemented.
- It incorporates the Loop Counter. This is a 6-bit register, with the most significant bit as a sign bit. It has an auto-increment/decrement facility, which enables it to be counted towards zero from either direction. On reaching zero, further counting is inhibited. The Loop Counter is used to repeat one (or several) microinstructions a predetermined number of times, or to count (for example) the number of shifts needed to normalise a floating point number.

Control Decode This generates control signals from the microinstruction word. These variously route data around the ND-110PCX, and define CPU modes of operation.

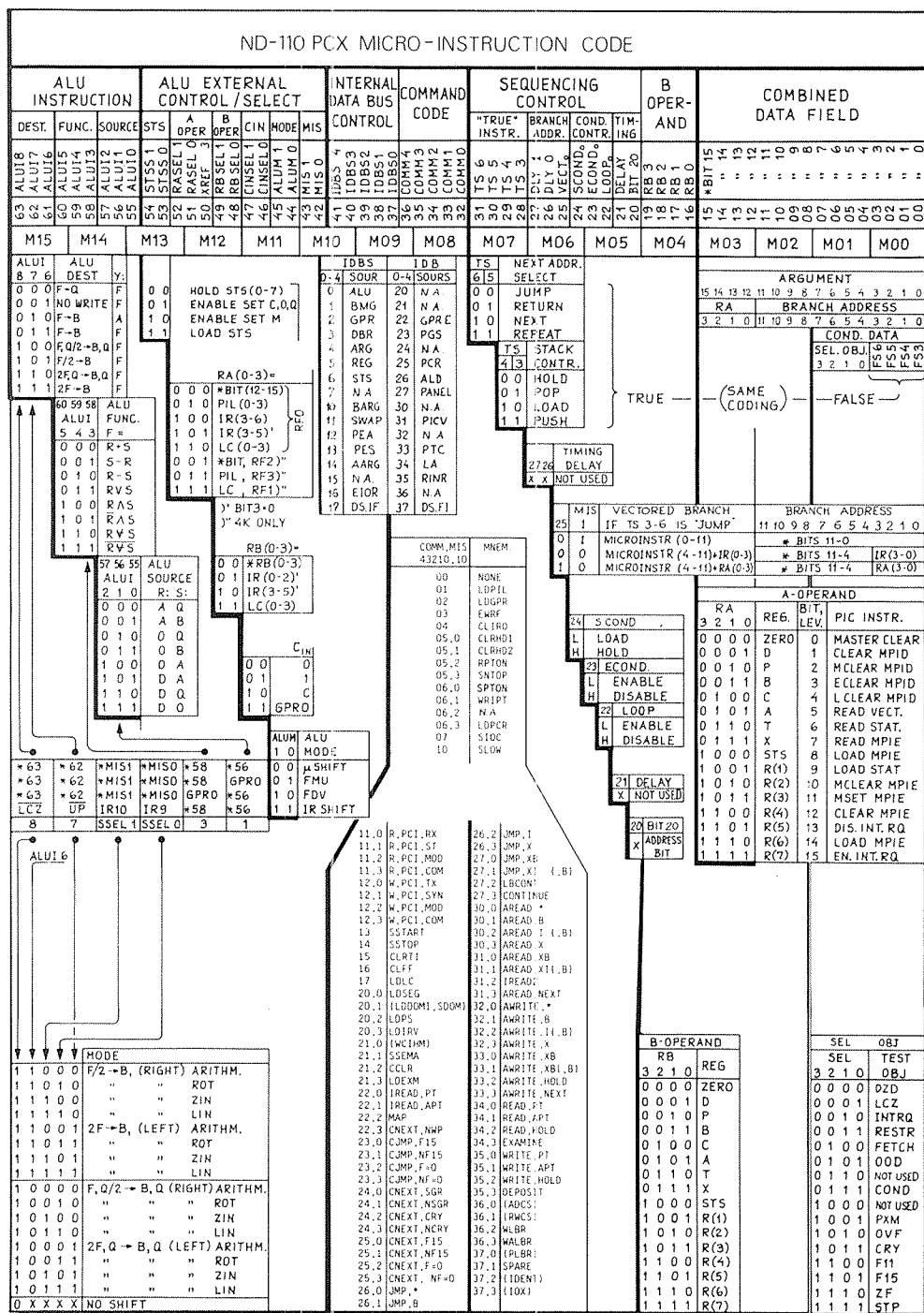
5.3 BUTTERFLY MICROPROGRAM WORD FORMAT

Control of the ND-110PCX CPU is effected by the microcode. Sequencing through successive microcode instructions is determined by a combination of external events (e.g. interrupts, bus cycles), and the microcode itself. Fetch operations in the microcode obtain the next macroinstruction, which then maps the CPU to the appropriate address in Control Store, to implement that macroinstruction by microcode. Each microcycle is subdivided into nanocycles for efficient use of time. External events affect the CPU through the trap and interrupt systems and the nanocycle controller.

The microword definition of fields and their functions is summarised in fig 5.2.

5.3.1 Summary of changes from ND-100

Bit 50 XRF3	- changed name, same function (Ext Reg File)
Bits 42-43 ¹	- added function to address the console chip
Bit 41 IDBS4	- IDB source entirely decoded
Bits 37-41 ¹	- extended list of IDB source selections
Bits 32-36	- extended list of command decodes
Bit 27 DLY1	- nano-cycle length control
Bit 26 DLY0 ⁰	- nano-cycle length control
Bit 21 DELAY ⁰	- nano-cycle length control
Bit 20 BIT20 ¹	- various functions
Bits 4-7 ¹	- test objects 0, 6, 8, 9 are changed



5.3.2 Microinstruction word: field definitions

Bit 0-3 : This selects the false sequence to be latched by SCOND (bit 24) to the hold mechanism. It specifies the sequencer instruction (jump, return, next, repeat) and stack control (hold, pop, load push), in case a later test ECOND (bit 23) gives a false result.

Bit 4-7 : This selects the test object to be output by SCOND (bit 24) to the hold mechanism. If a test object changes state, one microinstruction will be executed before the change is detected. The cause of this is the pipelining of ALU and sequencing calculations.

The 16 test objects are:

0	DZD	double zero detect
1	LCZ	loop counter is 0. Used to go through a loop a predetermined number of times
2	IRQ	Used to test the interrupt request flip-flop, to see if interrupts are pending.
3	RESTR	Used to test if the current program is running in restricted mode. Instructions that are privileged should not execute when this condition is true.
4	FETCH	Indicates whether the last memory access was a fetch of new macroinstruction. Used to generate the correct P value in case of page fault.
5	OOD	The one out detect flip-flop is used to set TG correctly in FAD, FSB and FMU instructions.
6	-	Not used (incorrect DZD)
7	COND	This condition latches the true/false result of the last microinstruction, so that it can be tested later if that is more convenient than testing it now. It will postpone the test as long as desired.
8	-	Not used (always false). It was used to detect if a single or multiple store instruction has affected *+1. If true, the prefetched instruction must be discarded; however ND-110PCX does not use prefetch.
9	PXM	PXM prefix mode, for future use in SINTRAN IV.

10	OVF	overflow from ALU. Indicates overflow during arithmetic operations.
11	CRY	carry out from ALU.
12	F11	Bit 11 from ALU.
13	F15	Bit 15 (sign bit) from ALU.
14	ZF	Result = 0 from ALU.
15	STP	Indicates the Stop flip-flop is set.

Bit 0-11 : This field contains the branch address in jump microinstructions. The jump addresses may also be generated by the MIC, and partly by the vectorized jump mechanism. When these sources generate jump addresses, the corresponding microword bits must be 0.

Bit 0-15 : These are enabled as an Argument, onto the IDB if IDB source (bits 37-41) is ARG (4).

Bit 12-15: These control the A-operand if RASEL (bits 51-52) are 0. The A-operand controls the BMG, as well as being a Register File and ALU operator.

Bit 16-19: These control the B-operand if RBSEL (bits 48-49) are 0.

Bit 20 : Firstly, it issued as the 13th bit, for microinstruction branches along with bits 0-11. This gives the 8k address range of the ND-110PCX Control Store.

Its secondary function is as the 5th address bit to the installation number PROM.

Bit 21 : This is the delay bit TC1 to the sub-cycle control area. It is not used in ND-110PCX.

Bit 22-24: These are condition control bits:

Bit 22 is active low. It increments or decrements the loop counter (always towards 0). If the selected condition is false, RETURN/HOLD is executed by the MIC. If it is true, TS3-6 (bits 28-31) are executed. The new value of the loop counter will not be available as an A-operand or test object, until after the next microinstruction.

Bit 23 is active low. It enables testing (against the previously set test condition) of the result of the previous microinstruction. If the result is true, the MIC sequencer executes TS3-6 (bits 28-31) to find the next microinstruction. If the result is false, the MIC sequencer executes the false sequence instruction which was previously output to the hold register by SCOND (bit 24).

Bit 24 is active low. It sets a condition to be tested on a later occasion. A 4-bit code indicating test condition, and a 4-bit code indicating the sequence instructions in case of a false test, are output to the Hold register.

Bit 25 : This specifies a vectored jump if the true sequence is "jump". It is an active low signal (inverted by assembler). Its source is determined by MIS 1 (bit 42):

Bit42	Bit25	
0	x	Jump
1	0	Jump lower 4 bits are IR(0-3)
1	1	Jump lower 4 bits are A-operand

Bit 26-27: These delay bits specify the length of a short microcycle. They are not used in the ND-110PCX.

Bit 28-31: These control the microprogram sequence "next address source" and "push-pop stack operation" if the condition result is true. They will only be in control as long as LOOP (bit 22) and ECOND (bit 23) are high.

Bit 32-36: This is the Command Field.
Command decodes: 00-17 are internal CPU control
20-27 are sequencer control
30-37 are external or memory request.

COMM		
<u>,MIS</u>	<u>MNEMONIC</u>	<u>DESCRIPTION</u>
00	NONE	No command executed.
01	LDPIL	Load the upper byte of the STS register with the upper byte of IDB. The new PIL will not be available as A-operand before the next but one microinstruction.
02	LDGPR	Load IDB into GPR. If a shift is specified in the same microinstruction, GPR will not be shifted, only loaded.
03	EWRF	Write IDB into the Register File specified by the A and B operands.
04	CLIRQ	Inhibit the ability of interrupts to modify the microinstruction sequence at the time of a mapped jump. Used to avoid interrupts when the MAPJ bit is used at other times than FETCH. CLIRQ also blocks PANEL interrupts from being included in the IRQ condition.

05,0	CLRHD1	Clear DMA controller interrupt
05,1	CLRHD2	Clear HDLC controller interrupt
05,2	NRPTON	Reset PC to ND-110PCX interrupt
05,3	NSNTOP	Set ND-110PCX to PC interrupt.
06,0	NSPTON	Set PC to ND-110PCX interrupt
06,1	WRIPT	Write in page table
06,2	-	Not used
06,3	LDPCR	Load the page control register.
07	SI0C	Set I/O control. The I/O Control Register is loaded from the IDB, by the microcode command decode SI0C. The I/O Control Register bits are:
	<u>Bit</u>	<u>Description</u>
	15	Secret flag
	14	X21 communications
	13	IEPTON: enable PC to ND interrupt
	10	Int Enable 10: power fail
	9	Int Enable 9: memory out of range
	8	Int Enable 8: parity error
	5	Int Enable 5: zero
	4	CPU request to I/O bus arbiter CRQI
	3	Reset real time interrupt RESRT
	2	OPCOM LED L1
	1	Run LED L2
	0	Self test LED L3 + enable master clear EMCL.
10	SLOW	Command a "slow" microcycle (approximately 475ns)
11,0	R,PCI,RX	See MISO-1 bits 42-43
11,1	R,PCI,ST	..
11,2	R,PCI,MOD	..
11,3	R,PCI,COM	..

12,0	W,PCI,TX	See MISO-1 bits 42-43
12,1	W,PCI,SYN	..
12,2	W,PCI,MOD	..
12,3	W,PCI,COM	..
13	START	Reset the Stop flip-flop. The CPU will start executing a main memory program.
14	STOP	Set the Stop flip-flop. Whenever a FETCH has been performed, the microprogram is forced to microaddress 3760 by the interrupt hardware, to execute the stop routine.
15	CLRTI	Reset the clock flip-flop. Every 20ms the microprogram is forced to microaddress 3762, thereby entering the clock routine. This routine sends out the CLRTI command.
16	CLFF	Clear all flip-flops having special functions regarding the floating point rounding indicator (TG). The DZD (double zero detect used in FDV) and the OOD (one out detect used in FAD, FSB and FMU) flip-flops are cleared by CLFF.
17	LDLC	Load the loop counter with the 6 lower bits of the IDB. The modified loop counter will not be available as an A-operand before the next but one microinstruction.
20,0	LDSEG	Load segment register. This selects a physical 64k memory address area (bits 16-23), in the MAC.
20,1	(LDDOMI,SDOM)	Load SINTRAN IV domain register
20,2	(LDPS)	Load SINTRAN IV PS register
20,3	LDIRV	Load instruction register (MIC).
21,0	(WCIHM)	Write cache inhibit map (no cache in ND-110PCX)
21,1	SSEMA	Set semaphore: shared memory handshake
21,2	(CCLR)	Cache clear (no cache in ND-110PCX)
21,3	LDEXM	Load examine: OPCOM physical address, forced through page table.
22,0	IREAD,PT	Indirect address read, relative to page table
22,1	IREAD,APT	Indirect address read, relative to alternative page table
22,2	MAP	Use IDB as an instruction
22,3	CNEXT,NWP	Conditional "next", if P is not changed in the last cycle.

23,0	CJMP,F15	Conditional jump if F15 true
23,1	CJMP,NF15	.. F15 false
23,2	CJMP,F=0	.. F0 true
23,3	CJMP,NF=0	.. F0 false.
24,0	CNEXT,SGR	Conditional next if SGR true
24,1	CNEXT,NSGR	.. SGR false
24,2	CNEXT,CRY	.. carry true
24,3	CNEXT,NCRY	.. carry false.
25,0	CNEXT,F15	Conditional next if F15 true
25,1	CNEXT,NF15	.. F15 false
25,2	CNEXT,F=0	.. F0 true
25,3	CNEXT,NF=0	.. F0 false.
26,0	JMP,*	Jump, P-relative
26,1	JMP,B	Jump, B-indexed
26,2	JMP,I	Jump, indirect, may be B-indexed
26,3	JMP,X	Jump, X-indexed.
27,0	JMP,XB	Jump, X- and B-indexed
27,1	JMP,XI (,B)	Jump, indirect, X-indexed or X- & B-indexed
27,2	(LBCONT)	Logical bank continue (PREX instr, SINTRAN IV)
27,3	CONTINUE	Fetch new instruction relative to P.
30,0	AREAD,*	Read, P-relative
30,1	AREAD,B	Read, B-indexed
30,2	AREAD,I (,B)	Read, indirect, may be B-indexed
30,3	AREAD,X	Read, X-indexed.
31,0	AREAD,XB	Read, X- & B-indexed
31,1	AREAD,XI (,B)	Read, indirect, X-indexed or X- & B-indexed
31,2	IREAD2	Read 1st part of 32-bit indirect address
31,3	AREAD,NEXT	Read from next address.
32,0	AWRITE,*	Write, P-relative
32,1	AWRITE,B	Write, B-indexed
32,2	AWRITE,I (,B)	Write, indirect, may be B-indexed
32,3	AWRITE,X	Write, X-indexed.
33,0	AWRITE,XB	Write, X- & B-indexed
33,1	AWRITE,XI(,B)	Write, indirect, X-indexed or X- & B-indexed
33,2	AWRITE,HOLD	Write in last address
33,3	AWRITE,NEXT	Write in next address.
34,0	READ,PT	Read, page table-relative, addr fr IDB
34,1	READ,APT	Read, alt PT-relative, address from IDB
34,2	READ,HOLD	Read from last-used PT, address from IDB
34,3	EXAMINE	Read, physical addressing using segment register.

35,0	WRITE,PT	Write, page table-relative, addr fr IDB
35,1	WRITE,APT	Write, alt PT-relative, address from IDB
35,2	WRITE,HOLD	Write to last-used PT, address from IDB
35,3	DEPOSIT	Write, physical addressing using seg reg
36,0	(ADCS)	Set address to control store: WCS only
36,1	(RWCS)	Read/write to CS: WCS only
36,2	WLBR	Write logical bank register (SINTRAN IV)
36,3	WALBR	Write alt logical bank reg (SINTRAN IV)
37,0	(PLBR)	?
37,1	SPARE	-
37,2	(IDENT)	Ident request
37,3	(IOX)	IOX request

replaced in ND-110PCX
by mailbox simulation

Bit 37-41:

This controls which source is selected onto the IDB.

<u>IDBS</u> <u>0-4</u>	<u>SOURCE</u>	<u>DESCRIPTION</u>	STOF ENCD EDO
0	ALU	ALUF OR A-OPR	* . .
1	BMG	bit-mask-generator: a 1 among 0s	* . .
2	GPR	general purpose register	* . .
3	DBR	DB/cache read register	* . .
4	ARG	microcode argument register	* . .
5	REG	register-file selected by A & B op	. . *
6	STS	macro status register	* . .
7	N.A.	(old MMR)	. . *
10	BARG	B-operand as argument (0-3)	* . .
11	SWAP	byte swap register	* . .
12	PEA	parity error address	. . *
13	PES	parity error status & address	. . *
14	AARG	A-operand as argument on bits 3-6	* . .
15	N.A.	(old PICS)	. . *
16	IOR	I/O register (see bit 32-36: 07)	. . *
17	DSABL,IF	dummy (IDB to FIDB)	. . *
20	N.A.	(old MIPANS)	. . *
21	N.A.	(old MAPANS)	. . *
22	GPR,SEXT	GPR, sign extended	* . .
23	PGS	paging status extended	. . *
24	N.A.	(old CSR)	. . *
25	PCR	paging control register	. * *
26	ALD	AL descriptor and print-status	. . *
27	PANEL	panel IDB interrupt vector	. . *
30	N.A.	(old RCS)	. . *
31	PICV	PIC interrupt vector	. . *
32	N.A.	(old LBR)	. . *
33	PTC	read PT content	. . *
34	LA	LA(10-15), CA(0-9) update ALU jump	. .
35	RINR	read installation number	. . *
36	N.A.	-	. . *
37	DSABL,FI	dummy (FIDB to IDB)	. . .

NOTES:

- 1 ENCD enables the CD out of the MAC gate array.
- 2 EDO enables the FIDB out of the ALU gate array.
- 3 STOF enables the IDB onto the FIDB.
- 4 PCR is an implied source with LBR. ENCD enables CD from the MAC onto the CD bus, and then via CD/IDB (STOC) to the IDB. CD in MAC is a multiplex of LBR/PCR, and is selected by COMM4 (COMM,PLBR for LBR). If COMM4=0, then PCR is selected.

Bit 42-43: Miscellaneous bits. These bits are used to control the shift linkage in the ALU. They are also used as an extension to the command field (see bits 32-36).

A new function for bits 42-43 (exclusive to ND-110PCX) is that they are used to address the console, in conjunction with the COMM,R,PCI (11) and COMM,W,PCI (12) commands:

MIS 00	read	Rx holding register	,RX
00	write	Tx holding register	,TX
01	read	status register	,ST
01	write	SYN1/SYN2/DLE register	,SYN
10	read/write	mode registers	,MOD
11	read/write	command register	,COM

Bit 44-45: These modify bits 55-63, before they reach the ALU. See bits 55-63.

Bit 46-47: These control the carry-in line to the ALU:

00	Carry is 0
01	Carry is 1
10	Carry is the C flip-flop of the STS register
11	Carry is the GPR bit 0 (used in division).

Bits 48-49: These control the source of the B-operand to the CPU:

00	The B-operand is bits 16-19 of the microinstruction. It is used to address a register in Register File or ALU WRF.
01	The B-operand is bits 0-2 of the current macroinstruction. This is used to address the destination register, in macroinstructions.
10	The B-operand is bits 3-5 of the current macroinstruction. This is used to address the source register, in macroinstructions.
11	The B-operand is bits 0-3 of the loop counter. This is used when performing a level change, to provide fast scanning of addresses in Register File and ALU WRF. It is also used to read the loop counter from MIC onto BAO-3, using BARG in the ALU to steer it onto the IDB

Bit 50-52: These control the source of the A-operand to the ALU.

- 000 The A-operand is bits 12-15 of the microinstruction.
- 010 The A-operand is the PIL register (used to address the current level register block in the Register File).
- 100 The A-operand is bits 3-6 of the current macroinstruction. This is used in bit operation instructions, and in instructions specifying a level in the Register File).
- 101 The A-operand is bits 3-5 of the current macroinstruction. This is used to address the source register, in macroinstructions.
- 110 The A-operand is bits 0-3 of the loop counter.

Bit 53-54: These control the behaviour of the 8 lower bits of the STS register:

- 00 does not affect the STS register.
- 01 updates the C, O and Q flip-flops, to the result of the current ALU instruction.
- 10 updates the M flip-flop to the disappearing bit in a shift micro-instruction. If there is ALUM IRSHIFT, M will be updated, regardless of the setting of bits 53-54.
- 11 loads the 8 lower bits of the STS register, from the IDB.

Bit 55-63: Together with bits 44-45, these control the behaviour of the ALU. Bits 44-45 modify bits 55-63 before they reach the ALU.

ALUM0 = 0(bit 44)
 ALUM1 = 0(bit 45): Micro Controlled Mode

The ALU behaviour is controlled entirely from the microprogram.

ALUM0 = 1(bit 44)
 ALUM1 = 0(bit 45): Multiply Mode

The ALU behaviour is controlled from the microprogram, except that GPR bit 0 controls the addition of the A-operand register or 0. This is only used in multiplication instructions.

ALUM0 = 0(bit 44)

ALUM1 = 1(bit 45): Divide Mode

The ALU behaviour is controlled from the microprogram, except GPR bit 0 controls whether or not the A-operand register should be added or subtracted. This is only used in divide instructions.

ALUM0 = 1(bit 44)

ALUM1 = 1(bit 45): Shift Instruction

The shift direction and shift mode is controlled from the loop counter together with bits 9-10 of the macroinstruction. The extra microinstruction executed after the loop counter reaches 0 does not shift the ALU result.

5.4 RETURN FROM SIMULATION INSTRUCTION: RTNSIM

5.4.1 Interface to the simulation routines

The interface between the calling microcode and the three level-16 macrocoded simulation routines which handle the mailbox (IOX, IDENT and Interrupt):

- has separate entry points for each of these routines
- has a common return instruction - called RTNSIM. In the software it is also called SECRE. This new instruction is the level-16 equivalent to WAIT. It is privileged.

The other part to this interface is four secret memory locations which contain the queue pointers for the IDENT queues. These are set up when an interrupt comes through the mailbox, and are reset when an IDENT is executed.

5.4.2 Entry point interface (UPSIM routine)

In general, the Working Register File is left as it was at the time of the calling macroinstruction. The first difference is in the IDENT routine call, where the A register contains the lower six bits of the calling IDENT instruction (i.e. the level information). The second difference is that if the calling instruction is IOX, then the T register contains the device number.

The microprogram, having saved the current register set and status, loads the A and T registers, switches off paging, and sets the P register to the first address of the simulation routine (see Chapter 6 section 6.9).

The parameters for the new T, A and P registers are passed in registers R5, R6 and R7 respectively. The level-16 scratch register is set non-zero. The secret flag is set, which allows fetch operations from the secret memory. The Paging-On indicator is reset. The scratch register BPFLG is tested, and if set (indicating breakpoint on) then the TEST indicator is checked. If TEST is set, START is not asserted, which allows breakpointing through secret code. Otherwise, if stop mode is set then START is asserted and scratch register STOPST bit 15 is set. This re-asserts STOP on RTNSIM. Finally a macro-jump is made.

The calling routines must take appropriate action. The PC-to-ND interrupt line must be disabled. They should also check if the PC is present, and if not call UPSIM. This prevents any interrupts which might call level 16 routines whilst level 16 is running. If the call was from OPCOM or the bootstrap loaders to the IOX routine, then they must load the RTNADR scratch register with the return pointer into the vector table which is used in RTNSIM.

The current PIL is not changed during level-16 routines, since it is used to point to the calling level on return from simulation.

5.4.3 Return from simulation (RTNSIM instruction - opcode 150403)

This instruction is the level-16 version of the WAIT instruction. Four of the working registers are used to provide data transfer from level-16 to the microcode, as follows:

A	data if appropriate (IDENT and some IOX)
D	non-zero if the A register contains return data
X	PID bits, set in PID format
T	IID (internal interrupts) bits, set in IIE format.

The L register is used to signal to the instruction the source of the level-16 call:

L bit 15 set	ND-to-PC interrupt; this calls the signal box interrupt-setting routine.
L bit 11 reset	PC-to-ND interrupt (0 if IOX or IDENT call); this calls the signal box check routine.

The microcode first resets the level-16 scratch register to zero. It then OR's the X register into the PID, then the T register into the IID (Internal Interrupt Detect register). If any IID bits were set in the T register, the routine jumps to the "check IID against IIE routine". If ION is active, the highest PID/PIE pair of bits set is found; if this is higher than the current level, the PVL, PCR and PIL registers are updated. This is done unconditionally, since a micro-interrupt may have been serviced whilst the level-16 routine was running. The L register is saved in one of the microcode R registers. The D register is checked, and if zero the old level (found from the PIL register) of the A register is restored. The other working registers (D,P,B,L,T,X and STS) are then restored. Paging indicator is thereby automatically set. The secret flag is reset.

If scratch register STOPST bit 15 is set, STOP is asserted and the bit is cleared. If bit 15 of the saved L register is set, the signal box "set PC interrupt" routine is called, for ATTN1. If bit 11 of the saved L register is set, the signal box "check flags" routine is called, as the level-16 routine must have been called by an interrupt. If bit 11 is not set (IOX or IDENT call), the PC-to-ND interrupt is re-enabled. The routine then loads the RTNADR scratch register into the loop counter and clears it down. The routine vectors on the loop counter via a return address table, to return to the calling routine. Location 0 in the table is the default, and contains a COMM,CONTINUE (27,0) command.

5.5 WRITING MICROPROGRAM

5.5.1 Summary

The ND-110PCX has 8k of 64-bit Control Store. From the software viewpoint, addresses to Control Store are normally expressed in octal notation; they are therefore represented here as address range 0-17777₈.

Generating microcode involves the following operations:

- 1 Specify the complete functionality of the macroinstruction that is to be implemented by the microprogram.
- 2 Design the structure of each microinstruction, using the symbolic expressions for the ND-110PCX "microinstruction set" (see section 5.6).
- 3 Assemble the symbolic microprogram into binary format, using the ND-Butterfly microcode assembler program.
- 4 Process the resulting output files through the PROM-Program series of files. Then use the PROM-BURN program to produce Control Store PROM.

5.5.2 Design of microcode

The process of designing microcode to implement a machine instruction, requires detailed understanding of the structure of the microinstruction word, and how the various fields and combinations of fields in this word stimulate the hardware, to perform the required operations.

It also requires a detailed knowledge of all the ND-110PCX microinstruction mnemonics, which provide the tools to code the microprogram. A list of these is given in section 5.6.

5.5.3 Assembly language considerations

The ND-Butterfly assembler creates a binary file from the symbolic microcode. In doing so:

- if "/" is found in a character group, the current location counter (CLC) is updated to the octal number in front of /.
- if ":" is found in a character group, the rest of the group is regarded as a label.
- the character ";" terminates each microinstruction, and CLC is then incremented. Each microinstruction may contain an unspecified number of lines.
- the termination to a macroinstruction is the label CONT: this sets the Z register to zero, and executes a COMM,CONTINUE T,JMP which fetches the next macroinstruction relative to P. This command can be executed elsewhere.
- the characters "space" and "tab" are accepted by the ND-Butterfly assembler as delimiters between mnemonics, labels and numbers.
- the character "%" is used to indicate that the rest of the line is a comment.

Any other characters are used to form character groups, which may be mnemonics, labels and commands. If a number between 0-7 is found in a character group, it is regarded as an octal number. Each mnemonic in the list is translated by the assembler into its corresponding octal value. All the octal values of the mnemonics in one microinstruction are "ORed" together. If an octal number is encountered, it is placed in

bits 0-15 of the microinstruction word and considered as an argument. If a symbolic name is encountered which is not found in the mnemonic list, it is regarded as a label, which is or will be defined.

Macroinstruction branching is achieved by the following series of commands to the MAC gate array.

```
COMM,JMP,xxx
COMM,CJMP,xxx
COMM,CNEXT,xxx
```

Microinstruction branching uses the standard format, whereby the condition and false sequence must be set up at least one microinstruction before CONDENABLE (condition enable) is executed. The test is on the result of the set condition in the previous microinstruction.

When there is no more input to the assembler, all labels are replaced by their values, which are filled into bits 0-11 and 20, in the correct microaddresses. Note that bits 22-27 of the microinstruction word are inverted after the word is assembled.

Changes from the ND-100-Rask assembler include the following:

- the mode files require changes because the file and user names are changed. These are now named BFLT-F48-GEN:MODE and BFLY-F32-GEN:MODE, for the 48-bit and 32-bit floating point maths sets. These mode files call MICRO-RASK and then BFLY-PROM-PROGRAM which takes the assembler output and creates the Data-I/O prom burner files.
- the following is a basic dialogue with the old NODAL assembler:

```
@(rask)nodal
NODAL - TSS 3.5
>OLD BFLY-MNE PROG
>LOAD BFLY-MNE-COMMENTS
>LOAD BFLY-MNEMONICS
>D05
FIRST:380                                new/old mnem number
LAST:380
MNEMONIC NO.380:COMM,MEGA                mnemonic
VALUE NO.380:0,2005,0,0                  octal words
COMMENT:DOES EVERYTHING CORRECTLY
>D030                                    finish
@
```

- The program RASK-MICRO is now used for mnemonic updates, and this produces the mnemonic files in the correct format for the program.

- The files used are
RASK-MNE-VALUES:DATA
RASK-MNE-SYMBOLS:SYMB
RASK-MNE-COMMENT:SYMB.
Mnemonics can be modified using the RASK-MICRO program.
The listing file is called BFLY-MNE-LIST:SYMB.

5.5.4 Generating control store PROM

The control store is generated using the RASK-MICRO assembler. This is part of the ND-570 assembler package ND-108588. The ND-570 Microprogram Guide (ND-05.014) is a useful reference.

There are five output files from the assembler, with the name BFLY-F32 or BFLY-F48. The extensions are:

```
:LIST    listing file - complete
:LABEL   labels and their addresses
:ERR     error listing
:UDEF    undefined entries
:DATA    the assembled output for BFLY-PROM-PROGRAM.
```

The BFLY-PROM-PROGRAM runs on the ND-500 under Fortran, and generates eight output files of the form BFLY-Fnn-xx-yy:PROM where nn is 32 or 48, and xx-yy is the microcode bit range (e.g.0-7, 8-15, etc.).

These files can then be used by the PROM-BURN program. They also need new names.

The source files are MICRO-BFLY-INSTR:SYMB, MOPC, -VECT, -EXTRA, -ENTRY, -BCD, -HDLC, -BOOT, -SIGNL, -HIOX, and either -F32 for 32-bit or -F48 for 48-bit.

5.6 MNEMONICS LIST

The following seven pages present the list of mnemonics for the ND-110PCX microinstruction set. These are numbered 1 - 495.

The list includes the op-code for the microinstruction (in octal), and a one-line description of it. The description is truncated in this printout of the list, to include only the first 60 characters.

Note that the assembler inverts bits 22-27.

[illegible]

[illegible][illegible]

[illegible]

[illegible][illegible]

[illegible][illegible]

[illegible]

CHAPTER 6 SOFTWARE DESCRIPTION

TABLE OF CONTENTS

Section	Page
6.1 INTRODUCTION	6-7
6.2 MAILBOX	6-10
6.2.1 Mailbox devices	6-10
6.2.2 Standard mailbox locations	6-12
6.2.3 Disk mailboxes	6-12
6.2.3.1 Floppy disk mailbox format	6-13
6.2.3.2 Hard disk mailbox format	6-14
6.2.4 Asynchronous (serial port) mailbox format	6-15
6.2.5 Butterfly DMA devices mailbox format	6-17
6.3 IOX SIMULATION	6-19
6.3.1 Operation	6-19
6.3.2 IOX simulator modules	6-21
6.3.2.1 Tables and queues	6-21
6.3.2.2 Signal box	6-23
6.3.2.3 IOX simulator switch (IOXSWITCH)	6-26
6.3.2.4 IDENT handler (IDIOX)	6-27
6.3.2.5 PC interrupt (PCINT)	6-27
6.3.2.6 Timer (TMINT)	6-28
6.3.3 IOX instruction simulators	6-29
6.3.3.1 Terminal IOXs	6-29
6.3.3.2 Hard disk IOXs	6-31
6.3.3.3 Floppy disk IOXs	6-32
6.3.3.4 Standard disk status and control words	6-32
6.3.4 IOX 30x: the PC console as Terminal-1	6-35
6.3.5 IOX 10-13: Real time clock	6-36
6.3.6 Login display following logout	6-36
6.3.7 Other Butterfly-110 IOXs	6-37
6.3.8 Effects of no cache	6-37
6.4 BUTTERFLY-110 ASYNC DRIVER	6-38
6.4.1 Operation	6-38
6.4.2 MON 373: special INSTR	6-40
6.4.3 Terminal datafield extension	6-40
6.5 PC-NDI/O	6-41
6.5.1 Basic structure	6-41
6.5.2 Interrupts - PC signal box	6-44

6.5.3	Calendar handling	6-45
6.5.3.1	Get and Set commands	6-45
6.5.3.2	Operation through signal box	6-46
6.5.3.3	Get time	6-46
6.5.3.4	Set time	6-47
6.5.4	Terminal-1 operation	6-48
6.5.4.1	Output	6-48
6.5.4.2	Input	6-49
6.5.5	PC Peripheral device handling	6-49
6.5.5.1	Request queuing and device reservation	6-49
6.5.5.2	Device switch	6-50
6.5.5.3	Floppy disk handling	6-51
6.5.5.4	Hard disk handling	6-53
6.5.5.5	PC console display handling	6-56
6.5.5.6	PC console keyboard handling	6-56
6.5.5.7	Serial port handling	6-56
6.5.5.8	Parallel port handling	6-59
6.5.6	System initialisation	6-59
6.5.7	ND-kernel	6-59
6.5.8	System communications area	6-60
6.5.9	PC interrupts used	6-61
6.6	MULTITASKING PC SOFTWARE	6-62
6.6.1	MS-DOS and BIOS	6-62
6.6.2	Additional requirements for Butterfly-110	6-63
6.6.3	Prerequisite MS-DOS information	6-64
6.6.3.1	MS-DOS structure	6-64
6.6.3.2	PC interrupt system	6-65
6.6.4	Multitasking	6-66
6.6.4.1	The sharing requirement	6-66
6.6.4.2	PC-NDI/O multitasking control	6-66
6.6.4.3	BSM to PC-NDI/O multitasking control	6-68
6.6.4.4	Terminal manager multitasking	6-69
6.7	SYSTEM MANAGER	6-70
6.7.1	Overview	6-70
6.7.2	System modules	6-70
6.7.3	Keyboard compatibility	6-72
6.7.3.1	PC	6-72
6.7.3.2	ND	6-73
6.7.4	New keyboard features	6-74
6.7.5	System manager	6-76
6.7.6	Keyboard driver	6-77
6.7.7	Character format in input buffer	6-80
6.8	ND SERVERS	6-81
6.8.1	ND file services available to the PC	6-81
6.8.2	Communication between ND and PC	6-82
6.8.2.1	Extensions to MON 144 (MAGTP)	6-82
6.8.2.2	Status registers	6-85
6.8.3	File access requests from PC to ND-servers	6-86

6.9	PC SERVICE PROGRAMS	6-90
6.9.1	File access	6-90
6.9.1.1	Facilities	6-90
6.9.1.2	PC trap handler	6-92
6.9.1.3	Inter-CPU buffer messaging	6-94
6.9.3	File transfer program (DUPLI)	6-96
6.9.4	Print spooler service program (PSSP)	6-99
6.9.5	PC servers	6-99
6.10	SECRET MEMORY LOCATIONS	6-100
6.10.1	PROM addresses	6-100
6.10.2	NV-RAM addresses	6-100
6.10.3	DRAM addresses	6-100

LIST OF ILLUSTRATIONS

Figure		Page
6.1	Butterfly-110 software structure	6-8
6.2	Butterfly-110 mailbox devices	6-11
6.3	ND - PC communication	6-19
6.4	IOX simulator system modules	6-22
6.5	IOX simulator switch	6-26
6.6	IDENT handler routine	6-27
6.7	PC interrupt handler routine	6-28
6.8	Standard status word settings	6-33
6.9	Standard control word settings	6-34
6.10	Async data transfers on ND side of mailbox	6-39
6.11	PC-NDI/O basic block structure	6-42
6.12	PC-NDI/O segments	6-43
6.13	Butterfly-110 serial ports	6-58
6.14	System communication area	6-60
6.15	System area - device reservation flags	6-61
6.16	Typical MS-DOS memory map	6-64
6.17	PC interrupt system	6-65
6.18	MS-DOS free - timer sequence	6-66
6.19	PC-NDI/O multitasking control	6-67
6.20	BSM multitasking control	6-68
6.21	BSM modules	6-69
6.22	System manager structure	6-70
6.23	System manager display window	6-76
6.24	Keyboard driver table	6-78
6.25	ND servers for the PC	6-81
6.26	ND servers - allocation of ND memory space	6-83
6.27	Secret memory: PROM address allocations	6-101
6.28	Secret memory: NV-RAM address allocations	6-101
6.29	Secret memory: DRAM address allocations	6-102

6.1 INTRODUCTION

This Chapter describes program modifications and additions to SINTRAN and MS-DOS domains, for the Butterfly-110 machine. It assumes familiarity with the software functional description in Chapter 3, extending this to consider each of the program areas identified in the system software block diagram of fig 3.1. For convenience, this diagram is repeated here (fig 6.1).

The Reader should be familiar with the basic programming environment and related terminology of ND's SINTRAN and microcode, and of Microsoft's MS-DOS.

ND additions

On the ND side of mailbox, additions and changes include the:

- IOX Simulators

This term is used to cover the changes and additions to the microcode, and the addition of the IOX Simulator macrocode. Microcode changes particularly affect areas relating to interrupt handling, level changing, the IDENT instruction, and IOX(T) instructions, as well as the new RTNSIM instruction. The microcode accesses the IOX Simulator macrocode for level-16 interrupt handling and timer, level-16 IDENT handling, and level-16 IOX(T) handling. Additional macrocode provides dedicated IOX instructions for performing specific tasks related to Butterfly-110 functions.

- SINTRAN servers

A Butterfly Device Driver has been created, together with five corresponding Butterfly DMA devices, to handle file access operations with the PC side of the machine. One Fileserver (a real-time program called PCFSERV) calls the device driver (using MON 144) to perform whatever file access function on the ND-filessystem that the PC has requested. See also Chapter 3 section 3.9.4.

- Special async driver

A special driver has been created to handle data transfers between the ND and Butterfly-110's asynchronous mailbox devices. To SINTRAN, these are terminal devices, and so modifications include extension of the Terminal datafield to enable efficient byte transfers directly with async mailbox ring buffers. This significantly improves the speed of async data transfers between the ND and terminal-type mailboxes.

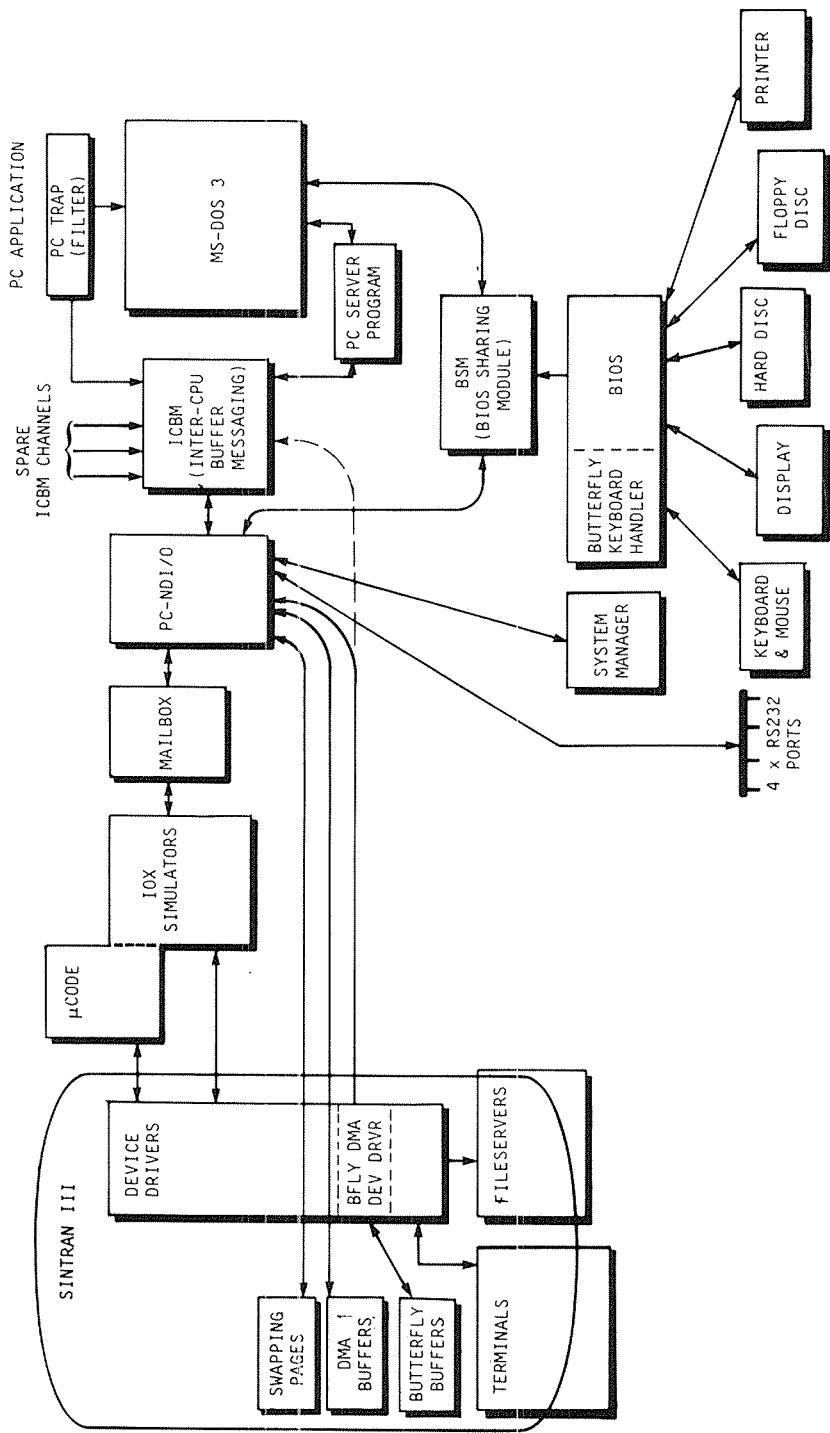


Fig 6.1: Butterfly-110 software structure

PC additions

On the PC side, additions include:

- PC-NDI/O, controlling PC communication with the ND
- a system manager, controlling the operating environment for the keyboard and display, to respond in PC-mode as a PC terminal, or in ND-mode as an ND terminal.
- a new keyboard handler, which replaces the standard keyboard handler in PC-BIOS.
- a BIOS sharing module, which allows multitasking use of the BIOS.
- PC Trap Handler and Inter-CPU Buffer Messaging module, to trap and perform file function (INT21h) calls coming from DOS application programs, when those calls are directed to a file in the ND-filesystem.

6.2 MAILBOX

Communication requirement Communication between the PC and ND processors in Butterfly-110 is via interrupts (one in each direction) and ND shared memory. The lowest level of communication is through the "signal box". Each side of the machine has its own signal box code: the PC's is in PC-NDI/O, and the ND's is incorporated in the ND-110PCX microcode.

At the signal box level, four interrupt sources are "multiplexed" onto the one interrupt that is available in each direction (ND → PC, PC → ND). These are:

- mailbox input/output
- terminal-1 output
- terminal-1 input
- calendar input/ output.

Terminal-1 data Terminal-1 data is passed between the PC and ND through a 4-word area in secret memory, called "st-mail" - see Chapter 7 section 7.6.2.5.

Calendar data Calendar information is passed through a 2-word area in secret memory, called PANS and PANC respectively - see section 6.3.3.

Mailbox transfers Mailboxes allow more complex messages to be sent than is possible through the "signal box" mechanism. They provide access to PC peripherals from the ND, and also pass a number of housekeeping jobs between the ND and the PC.

The MAILBOX interrupt is used by one side of Butterfly-110 to notify the other side that it has sent a mailbox. In the ND-110PCX, mailbox interrupts are handled by the IOX-simulator code. In the PC, they are handled by PC-NDI/O.

6.2.1 Mailbox devices

A mailbox is an area of ND secret memory which contains information describing the type of input or output operation required. Mailboxes vary in size, from 16 words upwards.

Chapter 3 section 3.3.4 gives a brief description of the three basic types of mailbox, and identifies the current list of mailbox "devices". Fig 6.2 provides more detailed information on mailbox devices in the Butterfly-110 system.

Mailbox	Use	ND term	ND LDN	device number (oct)	ND ident (NDID)	PC ident (PCID)	PC pointer to mailbox (hex)
floppy disk 1	DOS drive A					1	
hard disk 1	DOS drive C					2	
floppy disk 2	DOS drive B (not yet implemented)					3	
hard disk 2	DOS drive D (not yet implemented)					4	
terminal 5 O/P	COSMOS async port output	5	36	340	44	20	8036
terminal 6 O/P	printer port output	6	37	350	45	21	803A
terminal 7 O/P	emulator for SINTRAN login	7	38	360	46	22	803E
terminal 8 O/P	emulator for UE login	8	39	370	47	23	8042
terminal 9 O/P		9	48	1300	50	24	8046
terminal 10 O/P	QuadPort card output	10	49	1310	51	25	804A
terminal 11 O/P		11	50	1320	52	26	804E
terminal 12 O/P		12	51	1330	53	27	8052
terminal 13 O/P	provision for 2nd QuadPort card (included in ND side only)	13	52	1340	54	28	
terminal 14 O/P		14	53	1350	55	28	
terminal 15 O/P		15	54	1360	56	30	
terminal 16 O/P		16	55	1370	57	31	
terminal 5 I/P	COSMOS async port input	5	36	340	44	36	8034
terminal 6 I/P	printer port input	6	37	350	45	37	8038
terminal 7 I/P	Bfly terminal ND-mode SINTRAN	7	38	360	46	38	803C
terminal 8 I/P	Bfly terminal ND-mode UE	8	39	370	47	39	8040
terminal 9 I/P		9	48	1300	50	40	8044
terminal 10 I/P	QuadPort card input	10	49	1310	51	41	8048
terminal 11 I/P		11	50	1320	52	42	804C
terminal 12 I/P		12	51	1330	53	43	8050
terminal 13 I/P	provision for 2nd QuadPort card (included in ND side only)	13	52	1340	54	44	
terminal 14 I/P		14	53	1350	55	45	
terminal 15 I/P		15	54	1360	56	46	
terminal 16 I/P		16	55	1370	57	47	
Bfly buffer 0	ICBM channels & Bfly DMA devices					64	8054
Bfly buffer 1						65	8058
Bfly buffer 2						66	805C
Bfly buffer 3						67	8060
Bfly buffer 4						68	8064

- Notes: 1 terminals 2,3,4 do not exist in Butterfly-110 (& terminal-1 is the Console)
- 2 The ND (IOX simulator) has terminals 13-16 defined, which could be used to handle a second QuadPort card. No corresponding handlers currently exist in the PC (PC-NDI/O).
- 3 The "addresses" in this table are pointers to the addresses where these mailboxes are located. They are ND word addresses, so multiply by 2 to give the PC byte address.

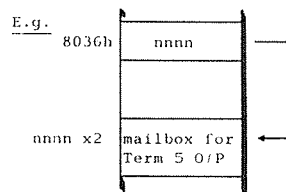


Fig 6.2: Butterfly-110 mailbox devices

6.2.2 Standard mailbox locations

Common entries The first 6 words are common to all mailboxes. These are located at word offsets 0 through 5 of every mailbox.

Word offset	Name	Function
0	QLINK	used by ND, to link mailboxes in the IDENT chains
1	PCQLINK	used by PC-NDI/O, as link to next mailbox in queue
2	INTHAN	pointer to the ND interrupt handler
3	ND LEVEL	ND program level (10-13) for handling this mailbox
4	NDID	ident code returned for a ND IDENT instruction
5	PCID	mailbox identifier, used by PC-NDI/O

ND LEVEL: Mailboxes, like all other I/O devices on the ND, are assigned appropriate ND program levels, on which an interrupt associated with that mailbox should be handled:

- 10 for output devices
- 11 for mass storage devices
- 12 for input devices.
- 13 for various other devices

PCID: This is a unique mailbox identifier to PC-NDI/O, which it uses to switch to the correct handler within the PC.

Interrupt timer Additionally the interrupt Timer uses mailbox displacements -1 to -3. The interrupt timer (TMINT), and its use of these mailbox locations for async mailboxes, is described in section 6.2.6.4.

Word offset	Name	Function
-3	TMSUB	timeout counter start value
-2	TMRD	timeout counter
-1	TTMR	timeout subroutine (if any)

6.2.3 Disk mailboxes

To the ND the floppy and hard disks are separate devices which can be used in parallel. On the PC, they are accessed via the same controller and therefore have to be used serially. Therefore, if one disk is being accessed when a request for the other is received, the second request is queued - see section 6.3.9 for further details.

6.2.3.1 Floppy disk mailbox format

Word offset	Name	Function
-16	FLTEST	test location
-11	SININ	fixed bits in SINSTAT
-10	BOOTING	boot flags
-9	SINCOM	control word from SINTRAN
-8	SINSTAT	status to SINTRAN (on IOX)
-3	TMSUB	} interrupt timer locations
-2	TMR	
-1	TTMRS	
0	QLINK	} standard mailbox locations
1	PCQLINK	
2	INTHAN	
3	ND LEVEL	
4	NDID	
5	PCID	
6	DATD1	high part of control block address (for IOX1565)
7	DATD2	low part of control block address (for IOX1567)
8	STWD1	status word 1: response bits from PC
9	STWD2	status word 2: status/ND err ret code } IOX1562/4
10	STWD3	status word 3: floppy format type
11	STWD4	reserved for future expansion
12	CNWD1	control word 1: ret mbx/cmnd
13	CNWD2	control word 2: autoloading
14	CNWD3	control word 3: test mode } IOX1563
15	CNWD4	control word 4: booting
18	COMMAND) (floppy command
19	DEV-ADDR) (sector address
20	DEV-MEM-TOP) (top 8 bits address
21	MEM-ADDR) floppy (low 16 bits address
22	OPT-WC-TOP) command block (options/word cnt
23	WORD-CNT) (low 16 bits word cnt
24	FL-STAT1) for further (status 1
25	FL-STAT2) explanation (status 2
26	TOP-LAST-ADDR) see ND-11.015 (top 8 bits last addr
27	LAST-ADDR) (floppy disk (low 16 bits last addr
28	REM-WDS-TOP) controller (top 8 bits words left
29	REM-WDS) manual) (low 16 bits wrds left

Words 18-29 are not part of the floppy mailbox in secret memory. Words 6 and 7 point to a command block in normal ND memory which PC-NDI/O has to read to find out what to do. Words 18 - 29 are that command block.

6.2.3.2 Hard disk mailbox format

Word offset	Name	Function	
-16	FLTEST	test location	
-15	IMADH	indicator for MADHI	} used in interface between IOX simulators and ND drivers
-14	IMADL	indicator for MADLO	
-13	IWDCN	indicator for WDCNT	
-12	IBLAD	indicator for BLADD	
-11	SININ	fixed bits in SINSTAT	
-10	BOOTING	boot flags	} values SINTRAN assumes
-9	SINCOM	control word from SINTRAN	
-8	SINSTAT	status to SINTRAN (on IOX)	
-7	MAXCYL	maximum cylinder address	
-6	HEADS	heads/tracks per cylinder	
-5	SECTRAK	sectors per track	
-4	WORDSEC	words per sector	
-3	TMSUB	} interrupt timer locations (not used for disk mailbox)	
-2	TMR		
-1	TTMRS		
0	QLINK	} standard mailbox locations	
1	PCQLINK		
2	INTHAN		
3	ND LEVEL		
4	NDID		
5	PCID		
8	STWD1	status word 1	} returned status, used by the ND (IOX504)
9	STWD2	status word 2	
10	STWD3	status word 3	
11	STWD4	reserved for future expansion	
12	CNWD1	control word 1	} control information to the PC (IOX505)
13	CNWD2	control word 2	
14	CNWD3	control word 3	} reserved for future expansion
15	CNWD4	control word 4	
16	MADHI	memory high address (upper 8 bits)	} IOX501/506
17	MADLO	memory low address (lower 16 bits)	
18	WDCNT	word count for transfer (IOX507)	
19	BLADD	block (= disk) address (IOX503)	

6.2.4 Asynchronous (serial port) mailbox format

In Butterfly-110, "asynchronous" (byte-by-byte transfers) mailboxes are required for the serial ports, parallel port, and display/keyboard (terminals 38 & 39) on the PC. These ports include those on the QuadPort+extension expansion card assembly.

There are two mailboxes per port, one for input and one for output. They are designed so that no word location is written to by both processors, which avoids problems of synchronising write operations.

Word offset	Name	Function
-9	SINCOM	control word from SINTRAN
-8	SINSTAT	status to SINTRAN (on IOX)
-4	MINBH	minimum number of bytes required in mailbox
-3	TMSUB] interrupt timer
-2	TMR	
-1	TTMRS	
0	QLINK] standard mailbox locations
1	PCQLINK	
2	INTHAN	
3	ND LEVEL	
4	NDID	
5	PCID	
6	DRATE	data rate code from SINTRAN (serial ports only)
7	OPPR	pointer for ND to mailbox in other direction
8	IOXSTAT	status command from IOX simulator to PC-NDI/O
9	PCSTAT	status command from PC-PC-NDI/O to IOX simulator
12	MAX BUFF LEN	max length -1, of ring buffer for this mailbox
13	PUT	ring buffer PUT pointer (for entry into mailbox)
14	GET	ring buffer GET pointer (for fetch from mailbox)
15	BCOUNT	byte count in ring buffer ¹
16	SPBYTE	test mode, and IOX304 location
18	MAX CHAR	greatest number of characters found (for debug use)
20	data	ring buffer for this mailbox
	↓	

¹ BCOUNT is a variable to the IOX simulator; PC-NDI/O does not attempt to update it when bytes are put in to or taken out of the FIFO buffer. BCOUNT is therefore only correct at the moment when IOX simulator calculates it - most of the time it does not reflect the true byte count.

DRA TE Serial port data rate, specified in the same format as the ND-100 data rate select word. Only the bottom 4 bits are valid.

The port characteristics are set to defaults when PC-NDI/O is installed: 9600 baud, 7 data bits, 2 stop bits, even parity, X-on/X-off disabled. These may be changed by ND programs, using the usual IOX instructions, but the baud rate for the input and output part is always the same, and is set by the lower 4 bits (input part) of the data rate selection control word.

The defaults are restored whenever PC-NDI/O is reset via INT 65h, i.e. after a master clear, cold start or warm start.

IOXSTAT command from IOX simulator to PC-NDI/O. This is a bit-significant word:

- bit 15 - A new command word has been sent.
- bit 14 - 1 = even parity
0 = no parity
- bit 13 - 1 = 1 stop bit
0 = 2 stop bits
- bits 11 & 12 select the data length:

<u>bit 12</u>	<u>bit 11</u>	
0	0	8 bits
0	1	7 bits
1	0	6 bits
1	1	5 bits

- bit 10 - enable X-on/X-off handling
- bit 9 - clear device.

PCSTAT Status from PC-NDI/O to IOX-simulator. This is a bit-significant word:

- bit 7 - set if an overrun error has occurred
- bit 6 - set if a parity error has occurred
- bit 5 - set if a framing error has occurred.

6.2.5 Butterfly DMA devices mailbox format

Butterfly DMA devices are used in file access operations between the PC and ND - see Chapter 3 section 3.3.10. A DMA-type mailbox provides control and status information for these operations, in word offsets 12, 13 and 8, 9 of the mailbox.

Word offset	Name	Function
-3	TMSUB] interrupt timer locations (not used in DMA mailboxes)
-2	TMR	
-1	TTMRS	
0	QLINK] standard mailbox locations
1	PCOLINK	
2	INTHAN	
3	ND LEVEL	
4	NDID	
5	PCID	
8	STATS	status set by PC part of Butterfly DMA device
9	RTSTA	status returned to PC part of Butterfly DMA device
12	INTST	interrupt status from PC for Butterfly DMA device
13	CTRWD	control word to PC part of Butterfly DMA device

Words 8, 9, 12, 13 are used for signalling and returning status between the ND and PC - see also section 6.7.3.2.

STATS The status bits, set by the PC, are:

- 0 interrupt on device RFT enabled
- 2 device busy - already dealing with a write operation from the ND
- 3 device RFT (not active)
- 4 error
- 5 message written by PC-NDI/O is awaiting collection by ND
- 6 no server started on the PC
- 7 clear the device sent by the PC
- 9 illegal command

RTSTA The status bits returned to the PC by the driver are:

- 2 device busy - already dealing with a write operation from the PC
- 5 message written by the ND is awaiting collection by the PC
- 6 no server started on the ND
- 7 clear the device sent by the ND
- 9 illegal command

INTST PC-to-ND interrupt control word bits:

- 8 interrupt, set by PC-NDI/O alone when a PC-to-ND interrupt is generated; cleared when ND driver reads status
- 10-15 operation code: 0 = read from ND (never used)
 - 1 = write to ND
 - 2 = clear device unit
 - 3 = transfer status to device unit

CTRWD ND-to-PC interrupt control word bits:

- 0 interrupt on device RFT (device finished)
- 2 activate device
- 10-15 operation code: 0 = read from PC (never used)
 - 1 = write to PC
 - 2 = clear device unit
 - 3 = transfer status to device unit.

6.3 IOX SIMULATION

6.3.1 Operation

Interrupts

Communication between the ND and the PC is performed via an area of shared ND-110PCX memory (fig 6.3). It uses two interrupt lines - one to the ND from the PC, and one from the ND to the PC.

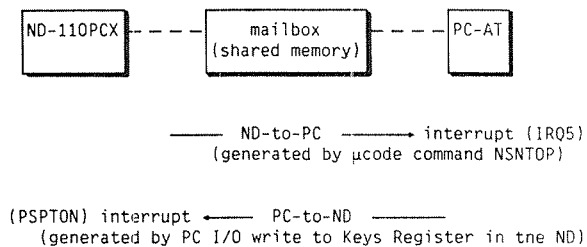


Fig 6.3: ND - PC communication

IOX simulators

To minimise the changes to SINTRAN (except the additions for shared file access with the PC, which are described separately), communication is arranged through changes to the IOX(T) instruction microcode. Instead of causing activity in the I/O part of the ND, IOX(T) instructions have been recoded to check if this IOX is to an "on-board" ND device (Real-time clock, HDLC, Console port). If not, operation moves to microcode program level 16, having first set the P Register on this level to point to the code handling IOX(T) commands for mailbox devices.

Code on level 16 consists of control routines and a set of IOX simulators. Each simulator mimics the action of a specific IOX(T) instruction, translating it into a read or write operation involving a mailbox "device".

The shared ND memory area includes the mailboxes. Each shared Butterfly-110 device (e.g. floppy disk, Winchester disk, printer, etc.) has a dedicated mailbox for each direction of transfer (ND-to-PC, PC-to-ND). The mailbox formats for each type of "shared device" in Butterfly-110 are given in section 6.2.

ND-to-PC transfer An ND output (write) instruction to a device causes the code simulating the IOX(T) instruction to enter relevant parameters which specify the task, into the mailbox for that device, then to interrupt the PC.

As soon as the interrupt to the PC has been sent, the IOX simulator returns to the ND user. No "handshake" to acknowledge acceptance of mailboxes is involved - once the interrupt has been sent, the ND assumes that the PC will accept the interrupt and proceed to perform the specified task. Therefore, while the PC processes the request, the ND may work on other tasks, so no processing time is lost. Note that on return from an IOX simulation, the A and T Registers are set up exactly as if the standard IOX instruction had been performed, so that the simulation is transparent to the rest of the ND system.

If, after performing the requested task, the PC wishes to pass a result back to the ND, it places the result into the appropriate mailbox, then sends an interrupt to the ND.

PC-to-ND transfer Tasks originating from the PC are handled in a similar way: the PC sets up a mailbox specifying the required task, that mailbox is entered into the (PCSOURCE) mailbox queue, and an interrupt then generated to notify the ND that the PC requires service (in this instance to process a mailbox).

When the ND receives an interrupt from the PC, then depending on the particular mailbox involved, an interrupt Timer routine may intervene to introduce a short delay before the ND services the interrupt. Otherwise, the ND-110PCX interrupt handler enters the interrupt onto the appropriate "real" interrupt level (i.e. one of program levels 10-13, see fig 3.9). Subsequently, an IDENT instruction is executed to determine which driver should be started to handle the interrupt.

The IDENT instruction has been recoded in a similar way to the IOX(T) instruction. The code executed on level 16 mimics the standard IDENT instruction, and returns the true IDENT code taken from the appropriate mailbox.

Once an interrupt has been handled in this way, the ND enters the appropriate device driver.

Status read by ND A status read causes the IOX(T) simulation code to read the status information from the mailbox.

Level 16
entry points There are four separate entry points on level 16:

- IOX(T) entry, IOXSWITCH
- IDENT entry, IDIOX
- PC interrupt entry, PCINT
- Timer entry, TMINT

6.3.2 IOX simulator modules

The IOX simulators comprise a set of discrete modules of code, executing on level 16. Fig 6.4 shows these modules and their interrelationships.

6.3.2.1 Tables and queues

IOX Table This table is used by the IOX switch (fig 6.4). It maps the IOX number to the relevant device table.

Each device table contains two words for each entry. The first points to the IOX simulation code, the second to the relevant mailbox field.

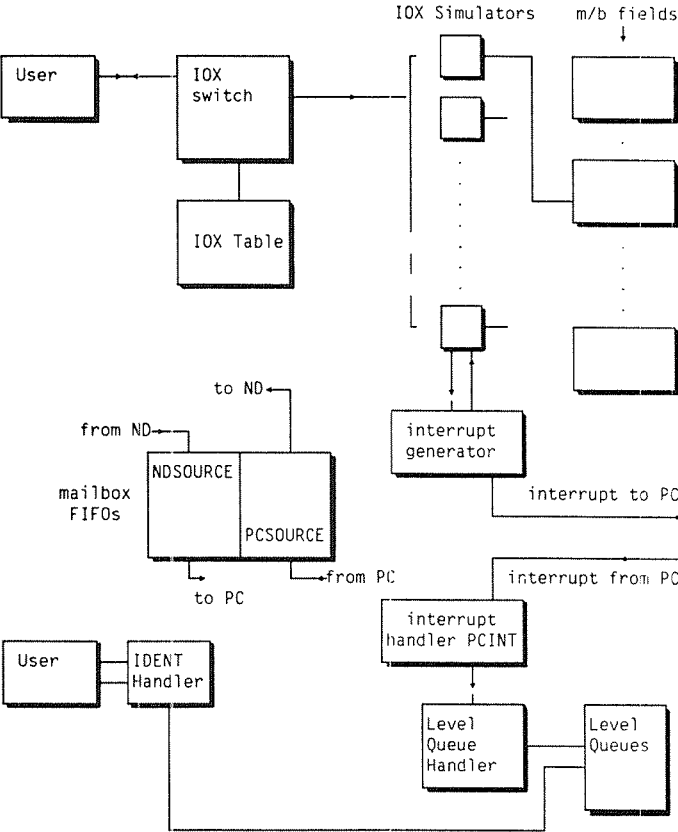


Fig 6.4: IOX simulator modules

Level queues	There is one level queue (fig 6.4) for each level that has an I/O interrupt capability. Each queue holds the mailbox fields currently interrupting on this level. They are the software equivalent of the "queue" of ND-100 interface boards, which can simultaneously issue an interrupt on a particular level.
Mailbox FIFO queue	<p>When the ND has a mailbox to send to the PC, it enters it into the NDSOURCE FIFO. On processing a mailbox interrupt from the ND, PC-NDI/O obtains from this FIFO the address where the mailbox is to be found. Then, when it has completed processing that mailbox, it returns to the NDSOURCE FIFO, to process the next mailbox in the queue. This sequence continues until the FIFO is empty (i.e. no more mailboxes queued for the PC to process).</p> <p>In the same way, the PC uses the PCSOURCE FIFO to queue mailboxes for the attention of the ND.</p>

6.3.2.2 Signal box

Utilities The concept of the interrupt handler signal box is described in Chapter 3. It multiplexes the four possible PC interrupt sources onto the one available interrupt line.

The signal box consists of four utilities:

- 1 PROSIGINT. The "process incoming interrupt" utility calls the clear signal box flag utility CLRSIGFLG.
- 2 CLRSIGFLG clears the signal box flags.

After the called higher routine has terminated, the:

- 3 SETSIGINT utility may be called, to set a responding interrupt.
- 4 CHKSIGBOX utility is called, to process any pending interrupts.

A higher level routine calls the SETSIGINT utility if an interrupt to the other processor is required. These utilities exist in both the ND and the PC. The names above relate to the ND microcoded implementation. The PC implementation is part of the PC-NDI/O program.

The following descriptions of these utilities are written in as non-processor specific terms as possible, so that they can be applied at a functional level to both the ND and the PC sides of the mailbox.

Signal Box Flags Globally, there are three processes that use the signal box. These are the IOX simulator, the terminal-1 interface (output and input), and the calendar. Various parts of these processes are called by the signal box, whilst other parts use the signal box to generate interrupts. There are four flags for each interrupt direction. Each is called by, and interrupts to, a different routine.

The flags are called either ATTE n or CONF n .

- ATTE flags call for attention
- CONF flags provide confirmation of a completed process.

The n can be 1, 2 or 3 and indicates the process:

- 1 being the IOX simulator
- 2 being terminal-1
- 3 being the calendar.

PC-to-ND flags These are four flags called ATTE1, ATTE2, CONF2 and CONF3. The PC sets the flags under the following circumstances:

ATTE1 The PC mailbox requests attention from the ND.

ATTE2 There is terminal-1 output to the ND (having filled £ST-MAIL).

CONF2 The terminal-1 input from the ND has been processed, and the receive buffer is empty.

CONF3 The calendar routine requested by the ND has been completed, and returned data (if appropriate) may be in £SM-PANS.

On receipt of these flags, the ND initiates the appropriate response:

ATTE1 The mailbox interrupt handling routine (level 16).

ATTE2 The terminal-1 PC input interrupt handling routine (BURTOP:).

CONF2 The terminal-1 PC output interrupt handling routine (BURTOP:).

CONF3 Calendar status register (PANS) update routine.

ND-to-PC flags These are four flags, called ATTEN1, ATTEN2, CONF2 and ATTEN3. The ND sets the flags under the following circumstances:

ATTEN1 The ND mailbox requests attention from the PC.

ATTEN2 There is terminal-1 output to the PC (having filled £ST-MAIL).

CONF2 The terminal-1 input from the PC has been processed, and the receive buffer is empty.

ATTEN3 A TRR PANC instruction has been executed, and the £SM-PANC has been loaded. The PC must process the new calendar command.

On receipt of these flags, the PC initiates the appropriate response:

ATTEN1 The mailbox interrupt handling routine.

ATTEN2 The terminal-1 input interrupt handling routine.

CONF2 The terminal-1 output interrupt handling routine.

ATTEN3 The calendar routine, which reads £SM-PANC and processes the command, then replies by filling £SM-PANC and generating a PC to ND flag CONF3.

PROSIGINT Process signal box interrupt utility

This utility disables the causing interrupt. It then passes control to the CLRSIGFLG utility.

CLRSIGFLG Clear signal box flag utility

This utility reads the flags that can cause an interrupt. If it finds one set (non-zero), it resets it, and transfers control to the appropriate higher level interrupt handler. If not, it calls CHKSIGBOX.

SETSIGINT Set signal box interrupt utility

This utility takes as a parameter the flag that is required to be set. It checks that the flag is not already set, and if it is, loops until it is reset. Then it sets the flag, and sets the interrupt line to the other processor. The utility returns to where it was called from.

For the ND side, the parameter of the flag is 0, 1, 2 or 3, passed in the Q register. If OPCOM is set and the PC is the console, the routine does not loop on the flag (for polled access).

CHKSIGBOX

Check signal box flags utility

This utility reads the interrupt pending set of flags. If a flag is set it passes control to the CLRSIGFLG utility, for processing of the new interrupt. Otherwise, it resets and re-enables the interrupt line. It then reads the interrupt pending set of flags again, in order to catch any that were set between reading last time and resetting the interrupt line. If any are found active, the interrupt line is set back active, and control passed to the PROSIGINT utility. If no flags are active, the utility checks the CALLOP scratch register, and if this is set, it passes control to the 20ms routine at entry point MS20C, otherwise it terminates.

Signal box
watchdog timer

A "watchdog timer" is user in the signal box when the ND uses its signal box to the PC. If the PC does not respond by clearing down the active flags in the signal box within a set time (1-2 seconds) of a second request from the ND, then the ND assumes that the PC has failed. It then sets its Powerfail Internal Interrupt, which is processed accordingly.

6.3.2.3 IOX simulator switch (IOXSWITCH)

The IOX simulator switch sequence shown in fig 6.5 applies. The IOX table is searched. If the entry is not found, it is an illegal IOX(T), so an "IOX error" internal interrupt is generated. If the entry is found, the appropriate IOX simulator is called.

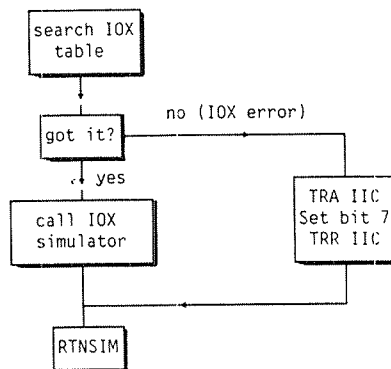


Fig 6.5: IOX simulator switch

6.3.2.4 IDENT handler (IDIOX)

The sequence shown in fig 6.6 describes the events involved.

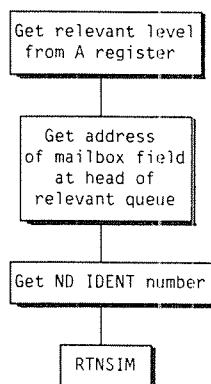


Fig 6.6: IDENT handler routine

6.3.2.5 PC interrupt (PCINT)

The PC interrupt entry point is indicated by the P Register while level 16 is inactive, so that a PC interrupt will start at the correct address. The microcode changes made for the IDENT and IOX(T) instructions then set the P Register on level 16 to the appropriate value.

The address of the mailbox which is the source of this PC interrupt, is obtained from the PCSOURCE FIFO pointer, and the level queue handler called. The operation is summarised in fig 6.7.

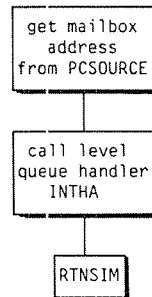


Fig 6.7: PC interrupt handler routine

6.3.2.6 Timer (TMINT)

For asynchronous mailboxes, the interrupt Timer reduces the number of interrupts to the ND. On receiving an interrupt from the PC, the IOX simulator code looks up a Timer table to determine whether this interrupt is for an async type of mailbox. If so, it reasonably expects that at least one byte of data is in that mailbox. Then, instead of immediately proceeding through the significant IOX simulator activity (PCINT, then IDIOX, then IOXSWITCH) to transfer what may be only one byte, it starts the interrupt Timer.

Async mailboxes have three locations (TTMR, TMRD, TMSUB) for the timer function, a minimum bytes value (MINBH), plus control words for handling the mailbox ring buffer - see section 6.2.4 (async mailbox format) and Chapter 3 section 3.5.3 (async mailbox data buffer).

Essentially, on being started, the Timer counts towards zero from its initial value TMSUB, during which time more bytes may be entered into this mailbox's FIFO buffer. When either the Timer times-out (maximum delay before this mailbox must be processed) or the minimum bytes in FIFO buffer value MINBH is reached, the IOX simulator is triggered to process this mailbox.

6.3.3 IOX instruction simulators

The IOX instruction simulators are called from the simulator switch IOXSWITCH. The call and return formats are:

- Call: A Register - data (if relevant)
T Register - IOX number
X Register - pointer to mailbox field
- Return: A Register - data/status (if relevant)

6.3.3.1 Terminal IOXs

See section 6.2.9 for "terminal-1" IOXs (IOX 301-307).

Input channel (interrupt level 12)

These also apply to the additional four terminals in a Teamstation; in these cases the instructions are IOX x3x0, x3x1, etc., where the leading x may be 0 or 1.

IOX 3x0	Read a byte. Take data from the data area of a mailbox field and return it in the A register
IOX 3x1	Set baud rate. This may be used to set the data rate for terminal input and output. It writes the baud rate from the A register into the DRATE word of a mailbox. The bit patterns representing baud rates that can be specified in ND-110 are listed in the ND-100 Functional Description, ND-06.026.02EN, Appendix B. Not all are supported in Butterfly-110.
IOX 3x2	Read input status register. Take the relevant data word (SINSTAT) from a mailbox field and return it in the A register.

The resulting bits in the ND-110 input channel status register are:

- 0 - ready for transfer (RFT) interrupt enabled
- 2 - device active
- 3 - device ready for transfer
- 4 - inclusive OR of errors
- 5 - framing error
- 6 - parity error
- 7 - overrun

IOX 3x3 Write to input control register. Take the A register, containing input channel control bits as listed below, and write it to control word (IOXSTAT) in mailbox.

The relevant bits in the input channel control register are:

- 0 - enable interrupt on device RFT
- 2 - activate device
- 3 - test mode
- 4 - clear device
- 8 - X-on/X-off to be used
- 9
- 10 $\left\{ \begin{array}{l} \text{character length} \\ \text{number of stop bits} \end{array} \right.$
- 11 - number of stop bits
- 12 - parity generation/checking.

Output channel (interrupt level 10)

IOX 3x4 In normal ND-110, returns 0 in the A register.

In Butterfly-110:

- on terminal log-out, IOX3x4 simulator with A = value of CESC (picked up from the Terminal datafield - see also section 6.4.3) initiates presentation of the log-in display on the logging-out terminal - see section 6.3.6. The default value of CESC is the ESC character (33), but this may be changed by the user.
- if A contains either of the bit patterns 100000 or 140000, IOX3x4 performs a byte data transfer with a terminal (async) mailbox:
 - A=100000 specifies input from mailbox. Mailbox address is obtained from the PCSOURCE queue (location 140030),
 - A=140000 specifies output to mailbox. Mailbox address is obtained from the NDSOURCE queue (location 140031).

By reference to the MAX, PUT and GET pointers in the mailbox, a byte is read or written directly with the mailbox data buffer, so bypassing the much more time-consuming IOX simulator routines.

IOX 3x5 Write data to device. Take byte from the A register and write it into the mailbox ring buffer.

IOX 3x6	<p>Read output status register. Take the relevant status bits from the status field and return these in the A register.</p> <p>The relevant bits in the output channel status register are:</p> <ul style="list-style-type: none">0 - ready for transfer (RFT) interrupt enabled2 - device active3 - device ready for transfer.
IOX 3x7	<p>Write to output control register. Take the A register and write to control area in the mailbox. Only two bits have significance:</p> <ul style="list-style-type: none">0 set - enable interrupt on device ready for transfer2 set - initiate ND-to-PC interrupt.

6.3.3.2 Hard disk IOXs

These codes apply to disk system 1. Each disk system may consist of two disk units. For disk system 2, add 10 to the specified codes. The interrupt level is 11, and the ident code is 1 (for system 1) or 5 (for system 2).

IOX 500	Read memory address register.
IOX 501	<p>Load memory address register.</p> <p>The memory address register is loaded by two successive IOX501 instructions. The first loads the eight upper bits of the 24-bit address (A Register bits 0-7 → address 16-23). The second loads the lower 16 bits.</p> <p>The memory address register is read in the opposite sequence, the first IOX reading the 16 lower bits, and the second reading the 8 upper bits.</p>
IOX 502	Not used
IOX 503	Load block (disk) address register (BLAD).
IOX 504	Read status register
IOX 505	Load control word
IOX 506	Read block (disk) address register (BLAD)
IOX 507	Load word count.

6.3.3.3 Floppy disk IOXs

IOX 1560	Read data from floppy disk buffer. It is only used during controller testing.
IOX 1562	<p>Read floppy disk status register. Take the hardware status from mailbox and return in the A register.</p> <p>The significant bits are:</p> <ul style="list-style-type: none">2 - device active3 - device ready for transfer4 - inclusive OR of errors10 - hard/DMA error
IOX 1563	Write to floppy disk control register. Take the A register and write to control area in the mailbox. Interrupt the PC if bit 8, 2 or 3 is set.
IOX 1564	Same as IOX 1562.
IOX 1565	Load pointer high. Take the A register, containing the high order memory address byte (bits 16-23), and write it into the MADHI location of mailbox. The least significant byte of the A register contains this high order byte of the command/status block address.
IOX 1567	Load pointer low. Take the A register, containing the low order memory address word (bits 0-15), and write it into the MADLO location of mailbox.

6.3.3.4 Standard disk status and control words

Status words	The "status word" area in a disk mailbox is 4 words long: this allows for future extensions. These status words hold the returned status information in a fixed form, for each currently anticipated status signal. In this way the software in the PC side may remain completely general, and need not be concerned with the format of status words expected by the ND. Fig 6.8 lists the available status information that may be used.
Control words	In a similar way to the Status Words, the Control Words are used to send control information to the PC in a fixed form. Fig 6.9 gives the current assignments.

STATUS WORD 1	
Bit 0:	RFT interrupt enabled
Bit 1:	error interrupt enabled
Bit 2:	device active
Bit 3:	device RFT
Bit 4:	OR of errors
Bit 5:	framing error
Bit 6:	parity error
Bit 7:	overflow
Bit 8:	DMA error
Bit 9:	carrier missing
Bit 10:	write protect violation
Bit 11:	timeout
Bit 12:	missing clock/disc fault/seek error
Bit 13:	address mismatch
Bit 14:	compare error
Bit 15:	motor stopped (not used)
STATUS WORD 2	
Bit 0:	transfer complete
Bit 1:	transfer on
Bit 2:	on cylinder/seek complete on unit 0
Bit 3:	high/dual density controller
Bit 4:	hardware error
Bit 5:	deleted record
Bit 6:	retry on controller
Bit 7:	CRC error
Bit 8:	
Bit 9:	
Bit 10:] floppy error code
Bit 11:	
Bit 12:	
Bit 13:	
Bit 14:	
Bit 15:	
STATUS WORD 3	
Bit 0:] bytes
Bit 1:] /sector
Bit 2:	double sided
Bit 3:	double density
Bit 4:	trying to read/write during RTZ
Bit 5:	not used
Bit 6:	not used
Bit 7:	
Bit 8:] selected
Bit 9:] unit

Fig 6.8: Standard status word settings

CONTROL WORD 1	
Bit 0:	enable RFT interrupt
Bit 1:	enable error interrupt
Bit 2:	activate device
Bit 3:	test mode
Bit 4:	device clear
Bit 5:	start timeout for breaking connection [§]
Bit 6:	marginal recovery
Bit 7:	unit selection
Bit 8:	
Bit 9:	side
Bit 10:	
Bit 11:	
Bit 12:	
Bit 13:	not used
Bit 14:	
Bit 15:	

CONTROL WORD 2	
Bit 0:	write format
Bit 1:	activate autoloader
Bit 2:	direction of seek
Bit 3:	bad track
Bit 4:	
Bit 5:	device op. code
Bit 6:	
Bit 7:	
Bit 8:	
Bit 9:	
Bit 10:	
Bit 11:	floppy disk control word bits 9-13
Bit 12:	
Bit 13:	
Bit 14:	
Bit 15:	not used

Fig 6.9: Standard control word settings

6.3.4 IOX 30x: the PC console as Terminal-1

Terminal-1 handling is contained in the Butterfly-110 microcode.

- IOX 300 If OPCOM is asserted then nothing happens. Otherwise, the data in TRMDAT is put in the A register, and input RFT (data available) and retain errors bits are reset in TRMREG. The CONF2 ND-to-PC interrupt is called by placing 2 in the Q register and calling SETSIGBOX:.
- IOX 301 Nothing happens.
- IOX 302 This takes status from TRMDAT and reformats it for transfer to the A register. The input PIN bit is transferred from TRMREG to the A register, and if not in OPCOM the input RFT is similarly transferred.
- IOX 303 The PCI transmitter and receiver are disabled, and the PCI control register is set according to the A register. Then the PIN and RFT are transferred to the input part of TRMREG. The receiver is then unconditionally re-enabled, whilst the transmitter is conditional upon TRMREG bit 15. The Tx must be re-enabled in order to get a Tx buffer empty interrupt.
- Bit 13=0 of the A register for IOX 303 should set 2 stop bits, or 1.5 stop bits if a 5-bit data word has been selected. In Butterfly-110, if bit 13 is zero it will always select 2 stop bits, regardless of the data length selected.
- IOX 304 Passes 0 to the A register.
- IOX 305 This enables the PCI transmitter and sends the data to it. It resets the Tx disabled and output RFT bits in TRMREG. It writes the data to the #DLB-TxD secret memory location, and sets bit 0 of #DLB-TxR. The ATTN2 ND-to-PC interrupt is called by placing 1 in the Q register and calling SETSIGBOX:.
- Note that the written data may overwrite the transmit register of the alternate console device, as RFT is only based on the "in-use" device.
- IOX 306 Sets the A register according to the output PIN and RFT in the TRMREG.
- IOX 307 Transfers bit 0 of the A register to the output PIN of TRMREG.

6.3.5 IOX 10-13: Real-time clock (RTC)

The real-time clock is an on-board device for the ND-110PCX.

IOX 10	The A register is set to zero.
IOX 11	Synchronises the RTC. The RESRT signal in the I/O control register is asserted and then reset.
IOX 12	Transfers the RTCREG to the A register.
IOX 13	Sets the RTCREG according to the A register.

6.3.6 Login display following logout

When a user logs-out, Butterfly-110 presents the log-in display. This is effected by two patches, one to the login routine and one to the logout routine.

The logout patch starts the simulated login if a variable RETCO = 0.

The login patch tries to log in. If it is unsuccessful it sets RETCO = 1, and logs-out. The login routine then sees RETCO = 1 and makes no further attempt to log-in.

The login mechanism is that an IOXT is issued to the "h/w-device + 4" (Chapter 7 fig 7.4 shows a list of terminal device numbers), writing into that mailbox's ring buffer. Any code will suffice to stimulate the interrupt to the ND.

The IOX simulators respond by setting the SPEC1 bit in the mailbox's SPBYTE, and since a character has been entered into the input mailbox, an interrupt is generated to the ND. The ND then proceeds to process this interrupt as if in response to a code input from the terminal. On seeing the SPEC1 bit set in SPBYTE, it sends the log-in display to that terminal.

6.3.7 Other Butterfly-110 IOXs

IOX 30 Reset PC-NDI/O after Warm or Cold Start:

This IOX initialises PC-NDI/O.

IOX 31 Set POWERFAIL flag in Secret Memory:

The IOX simulators include a powerfail flag (PFFLG) which the ND sets in response to a variety of abnormal conditions it detects in the PC (power failed, signal box or interrupt watchdog timers timed-out, etc.).

6.3.8 Effects of no cache

There are a number of internal cache registers affected by ND-110PCX not having a cache. The following TRR instructions have no effect:

10	CCLR	cache clear
11 ₈	LCILR	lower cache inhibit limit
12 ₈	UCILR	upper cache inhibit limit
13 ₈	CILP	cache inhibit/enable island (new for RASK)

The cache status register TRA,CSR (10₈) will always read 000004₈, i.e. manual disable set, cache off and not updated.

6.4 BUTTERFLY-110 ASYNC DRIVER

To avoid heavy time overheads in calling the IOX simulator routines, to pass individual bytes of data between the ND and an asynchronous mailbox device, a special SINTRAN async driver has been written which allows direct byte transfers with the ring data area of async (terminal-type) mailboxes. This facility significantly improves the speed of asynchronous data transfers between the ND and a mailbox.

6.4.1 Operation

Existing SINTRAN Async Drivers use IOXs to perform data transfers with terminal-type devices; in Butterfly-110, these then call upon the IOX simulators to perform the transfer. This process does not take advantage of the fact that because the terminal mailbox and the corresponding SINTRAN terminal datafield are in the same ND memory (fig 6.10), the same data transfer can be achieved much faster by direct memory-to-memory transfers.

Initiation Characters are entered into a terminal's mailbox. When sufficient characters have been entered, or if a timer-tick (20msec) occurs beforehand, an interrupt to the ND is given for this mailbox. The interrupt is handled by the IOX simulators, and the mailbox entered on the appropriate program level (12 for input). Then, to input from this mailbox, the special Butterfly Async Driver routine is run.

Terminal input Characters are entered into a terminal's input mailbox. Then, when the Butterfly async driver starts, it performs memory-to-memory transfers, as follows:

- characters are taken from the mailbox ring buffer and entered into the terminal input datafield for as long as there are characters in the mailbox ring buffer and the terminal input datafield is not full.

While the async driver is taking characters from the mailbox, the PC may enter further characters into the mailbox.

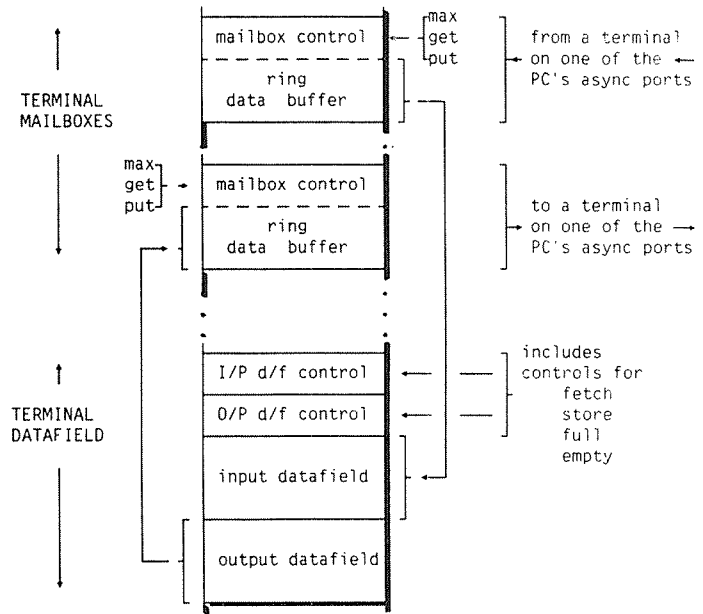


Fig 6.10: Async data transfers on ND side of mailbox

- When the mailbox becomes empty, the async driver stops and user program restarted.
- If the terminal datafield becomes full, reading from mailbox stops, interrupts associated with input from that mailbox are disabled, and user program is restarted. When space becomes available in the datafield, reading from mailbox automatically restarts, until mailbox is emptied.

Terminal output

Characters are written from the terminal output datafield into the corresponding mailbox output ring buffer. On entering a character into the output mailbox, that mailbox is sent to the PC, and an interrupt is issued to the PC.

The async driver continues to transfer characters to the output mailbox until either the datafield ring buffer becomes empty or the mailbox becomes full.

6.4.2 MON 373 - special INSTR

The INSTR (input string - MON 161) monitor call reads in a string of characters from a device.

In Butterfly-110, a special version of INSTR is used which reduces the CPU usage. This new version is assigned a new SINTRAN monitor call - MON 373. It reduces the monitor call overhead (i.e. parameter passing), which is significant because a monitor call has to be made for each line when a packet is received. COSMOS async is connected to the terminal device on the PC's COM1 port.

One of the features of the special Butterfly async driver is that it uses break mode 12 to respond to break characters in a COSMOS async string. COSMOS async forms its character strings into "frames", these being separated by a special end of frame (EOF) character. On input, whenever the terminal async driver detects this EOF character, it signals a break condition.

6.4.3 Terminal datafield extension

The SINTRAN III datafield layout for terminals is given in SINTRAN III System Documentation, Appendix A - Data Fields (ND-60.112).

To handle the additional requirements of the special SINTRAN async driver, 9 more variables/pointers are added to the terminal datafield. These are:

d/f displ	Name	Function
45	NBREAKS	number of break characters in buffer
46	BCMWFIEL	address of current MON call work field
47	UACTPRI	PCR when accessing caller's buffer
50	USADDR	address of caller's buffer
51	XBUFST	logical window address to ring buffer
52	NCHARS	number of characters in caller's buffer
53	CPITENTRY	PIT entry of terminal datafield
55	BRKCHAR	break character
56	BRKMODE	break mode number

6.5 PC-NDI/O

6.5.1 Basic structure

"PC-NDI/O" is a resident program which runs under the MS-DOS operating system to provide the input/output facilities required by the ND-110 processor. Since MS-DOS also has access to those facilities, PC-NDI/O has to synchronise access from the PC and the ND-110PCX.

PC-NDI/O has the following major components:

- ND-110 interrupt handler ("signal box")
- Terminal-1 interface & logging facilities
- Calendar
- Mailbox handling
- Peripheral sharing
- Peripheral handling:
 - fixed disk(s)
 - floppy disk(s)
 - serial ports
 - parallel port (printer)
- PC display and keyboard (interface to System Manager and Keyboard Handler).

The basic functions within PC-NDI/O are represented in fig 6.11. The following sections describe each of the functional modules shown.

Entry/Exit

Some operations performed by PC-NDI/O take a relatively long time to complete (floppy disk access for example), so it is necessary that further interrupts, especially terminal output, are serviced at the same time, to give the appearance of a continuous service to terminal screen(s). In addition a number of external asynchronous events - such as real-time or serial port interrupts - may result in calls to PC-NDI/O. For this reason, PC-NDI/O is made re-entrant, so allowing several I/O requests to be processed simultaneously.

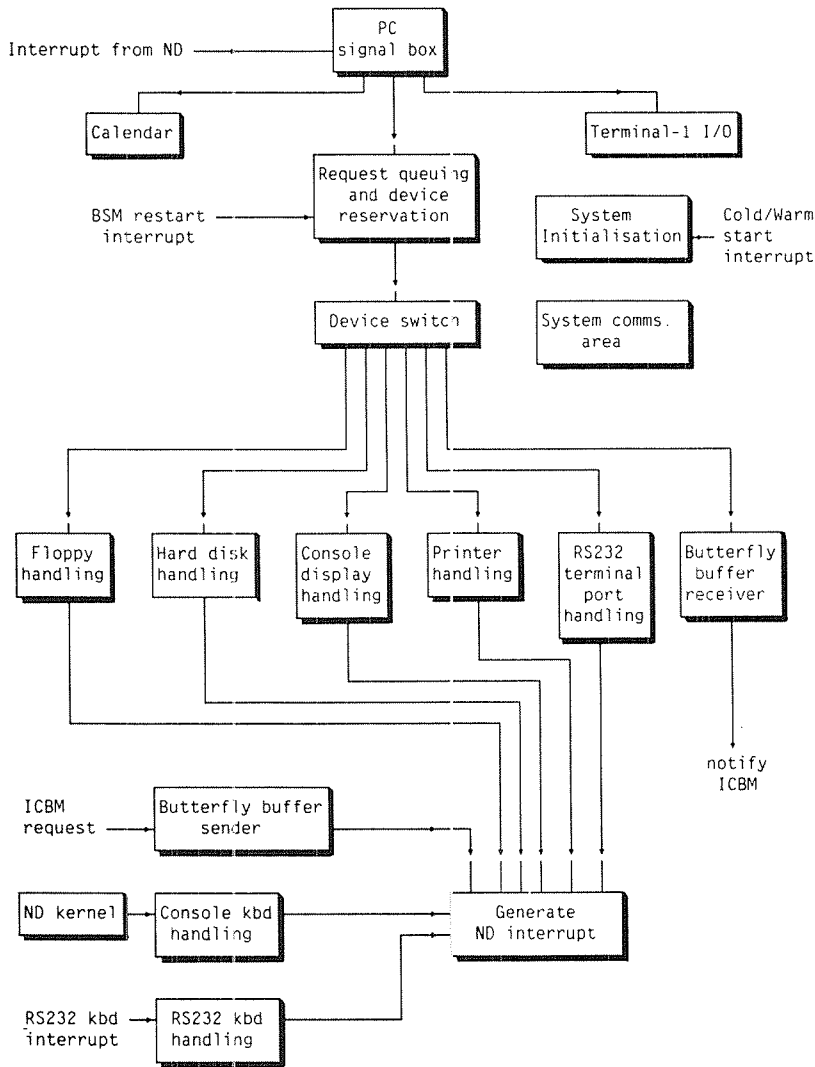


Fig 6.11: PC-NDI/O basic block structure

The points of entry are:

- signal box interrupt from the ND
- an interrupt from one of the RS232 ports
- a Butterfly buffer send request
- BIOS Sharing Module restart interrupt (see section 6.5.4.4).
- the System Manager.

Calling PC-NDI/O by any of the above methods results in a new instance of PC-NDI/O running; it then processes the request, and dies. A new instance in this context consists of allocating a data area for stack and local data (fig 6.12).

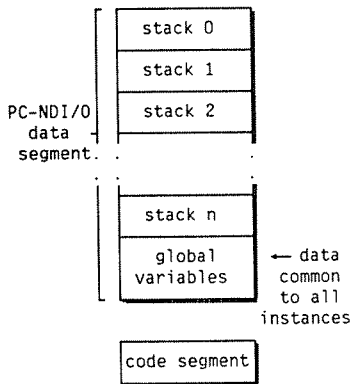


Fig 6.12: PC-NDI/O segments

There is only one segment for data, with global variables at the bottom and space for stack above; local variables are stored on the stack. PC-NDI/O splits the stack into a number of sub-stacks: each new instance of PC-NDI/O uses one of the sub-stacks. The code is either fully re-entrant (no global variables modified) or "serially reusable" (interrupts turned off during critical regions, such as modifying globals).

At each entry point into PC-NDI/O, the "save-context" routine is called, which stores the registers and offset register (port 165h) on the callers stack, and allocates one of the PC-NDI/O stacks. When processing of the task is complete, the "restore-context" routine is called; this frees the PC-NDI/O stack, and restores the processor and offset registers.

6.5.2 Interrupts - PC signal box

The PC signal box handles the signal protocol (see section 6.3.4), organises the context and stacking requirements within PC-NDI/O, and calls the appropriate handler (mailbox, terminal-1 output or input, or calendar) to service the interrupt requirement.

Signal box

As described in fig 6.3, communication between the PC and ND-110 processors is via interrupts (one in each direction) and shared memory. The lowest level of communication is the "signal box" interrupt handler. PC-NDI/O's signal box provides the PC side of the machine with the corresponding function to the signal box code in the ND-110 microcode. At this level, four interrupt sources are multiplexed onto the one physical interrupt line (fig 6.3). These sources are mailbox, terminal 1 output, terminal 1 input, calendar input/output, and mailbox input/output.

PC interrupt controller

Interrupts from the ND-110 are received on the IRQ5 line of the master 8259 programmable interrupt controller (PIC), which corresponds to interrupt 0Dh. On receiving an interrupt, further interrupts are masked out by writing to the mask register of the PIC, and the end-of-interrupt command (EOI) is sent. This prevents further interrupts from the ND while still allowing interrupts from lower priority devices, for example the floppy disk, which may be required to service the ND interrupt. Finally the interrupt line is reset by toggling bit 3 (PRNTOP) of the ND's Keys register (the PC's I/O port 166h).

Interrupt from ND

The reason for the interrupt is determined by the signal box flag settings. These are in ND "secret" memory, at address offset FF98.

Offset	Flag name	Meaning
FF98h	ndpc-atten1	Mailbox arrived
FF9Ah	ndpc-atten2	Terminal 1 output
FF9Ch	ndpc-conf2	Terminal 1 input conf.
FF9Eh	ndpc-attf3	Calendar

The flag is cleared and the appropriate routine called to service it. The multiplexing of the interrupt line means that there may be several interrupt sources for one physical interrupt, hence it is essential to service all ND interrupts before enabling them again, in order to ensure that none are lost.

Interrupt to ND To interrupt the ND-110, a flag is set to indicate the source, and bit 0 (PSPTON) of the ND's Keys register (port 166h) is toggled. The corresponding signal box flags are:

Offset	Flag name	Meaning
FF90h	pcnd-atten1	Send mailbox
FF92h	pcnd-atten2	Terminal 1 input
FF94h	pcnd-conf2	Terminal 1 output conf.
FF96h	pcnd-conf3	Calendar conf.

6.5.3 Calendar handling

The calendar module comprises two source files, namely CAL.C and CALASM.ASM. When compiled, these files are linked into PC-NDI/O.

6.5.3.1 Get and Set commands

The battery backed calendar device for the ND-110 is physically part of the PC, and is only directly addressable by the PC. The calendar module provides a transparent interface between the ND PANS and PANC secret memory registers, and the PC's calendar device.

The calendar device presents its record of time in a BCD format of Century, Year of century, Month, Day of month, Hour (24-hour format), Minute and Seconds. Unfortunately, the ND expects to get (read) or set (write) time as two 16-bit word-counts: one of seconds and another of half days since 0000:00 of 1-1-1979. Complicating matters further, the PANS and PANC registers allow 8 bits for transferring data, so that a full get/set comprises four operations.

The calendar module therefore provides two basic command types (get time, set time), each providing four functions, due to the restriction of 8-bit transfers through PANS and PANC data fields:

- get low 8 bits of seconds
- get high 8 bits of seconds
- get low 8 bits of half days
- get high 8 bits of half days
- set low 8 bits of seconds
- set high 8 bits of seconds
- set low 8 bits of half days
- set high 8 bits of half days.

6.5.3.2 Operation through signal box

A command in the secret memory PANC register originates from ND macrocode (instruction TRR 0). The microcode invoked by this command sends an interrupt to the PC, with the secret memory flag ATEN3 set and the secret memory PANC register holding the copy of the A-register.

When PC-NDI/O responds to the interrupt, the ATEN3 flag (which is then cleared) results in the calendar module being called. Assuming the command is both valid and supported, then the calendar module formulates the result and places it into the secret memory PANS register data field, copies the 8 most significant data bits of PANC to PANS, sets the secret memory flag CONF3, and interrupts the ND. An invalid or not supported PANC command is simply copied to the PANS, the CONF3 flag set, and the ND interrupted.

6.5.3.3 Get time

A get time command results in the following sequence of events:

- 1 the PC time is read,
- 2 this time is converted from its BCD format into an integer format (logical),
- 3 this logical format is then converted into an absolute format (two 16-bit unsigned integers, one for seconds in half-day and the other for half-days since 1st January 1979),
- 4 finally the PANS register is set up with the 8 data bits from one of the two 16-bit unsigned integers, depending upon the PANC command (e.g. read least significant 8 bits of days).

The procedure calls for a get time command are thus:

```
readtime();  
conv-from-bcd();  
logical-to-abs();
```

The logical-to-abs() procedure makes further calls to a procedure called leap-year(year), which provides a true or false return value for the year passed, depending upon the following rule:

a leap year occurs when the year is multiple of 400

OR

the year is a multiple of 4 but not a multiple of 100.

6.5.3.4 Set time

A get time command results in the following sequence of events:

- 1 it make the three procedure calls used by "get time" to obtain the current time.
- 2 then it sets the 8 bits of the appropriate 16-bit integer to zero before copying the 8 bits supplied by the command into that 16-bit integer.
- 3 the updated absolute time is then converted to logical time.
- 4 this logical time is copied to a second area of storage, for use when updating the DOS software clock.
- 5 the logical time is then converted to BCD,
- 6 this time is then written to the calendar device.
- 7 the DOS time then needs to be updated. This can only be done if DOS is not already entered.
 - If DOS is not entered, then DOS set time and date function calls are made.
 - If DOS is entered (i.e. busy), then the two function calls in a small procedure are linked into the DOS INT 28 job queue, where they will be executed as soon as DOS is free.

The procedure calls for a get time command are thus:

```
readtime();
conv-from-bcd();
logical-to-abs();

abs-to-logical();
preserve-for-dos();
conv-to-bcd();
writtime();
update-dos-time();
```

Multiple clock sources

It should be noted that from a system view point the setting of the PC based calendar device should be made from the SINTRAN command UPDAT if Sintran is running, as this will result in the DOS software clock being updated as well as the PC's calendar device and the SINTRAN software clock.

Updating the PC calendar device using the internal DOS commands TIME and DATE only updates the DOS software clock - if the clock.sys driver is resident then the calendar device would also be updated - so an already started SINTRAN would remain with the old time (i.e. that read using the calendar module just after SINTRAN load time).

6.5.4 Terminal-1 operation

The terminal-1 I/O module allows a PC application to interact with the ND. Characters sent to the ND in this way are received as if they originated from the ND console (i.e. terminal-1).

6.5.4.1 Output

Output from the ND's terminal-1 are first written to the message log file \BFLY-110\CONSLOG.CUR (current) and \BFLY-110\CONSLOG.PRV (previous). Following this, INT64h is executed. If an application wishes to receive output from the ND, it must attach to this interrupt. If no application has attached, INT64h points to a module which outputs the character(s) to a display window area at the top of the PC screen.

The format of an INT64h call is:

INT64h AL = character

on return AH \neq 0 signifies that the application does
 not want any further characters until
 it requests them (see section 6.4.4.2).

6.5.4.2 Input

If an application wishes to send a character to the ND, an INT63h is executed. The ND receives the character as if it has been input via the terminal-1 (ND console) port. The format of an INT63h call is:

INT63h AH = 0 AL = character

 or AH = 1 implies the application is ready
 for further characters.

6.5.5 PC peripheral device handling

6.5.5.1 Request queuing and device reservation

Sharing devices Fig 6.1 indicates that requests for BIOS service originate from two sources:

- from MS-DOS, in response to a PC user program
- from the ND, via PC-NDI/O.

The activity to share BIOS between these two sources is referred to as "multitasking".

In Butterfly-110, both DOS and PC-NDI/O have concurrent access to the disks, PC display, and parallel printer port, via the BIOS routines INT 13h, INT 10h and INT 17h respectively. A layer of software called the BIOS Sharing Module (BSM) mediates between the two sources of request, to share these resources. The BIOS sharing module (BSM) works with PC-NDI/O to take control of these three interrupts at installation time.

Control mechanism The principle of operation is that DOS functions are never called as a result of a hardware interrupt (functions which use the DOS timer interrupt, such as the print spooler, are special cases - see Print Spooler, below), so DOS can only make BIOS calls when no hardware interrupts are being serviced. The BIOS call is intercepted by the BSM, which sets a reservation flag to "reserved by DOS" status, and passes the call to BIOS. Device reservation flags in the System Communications Area (see section 6.4.9) are set by PC-NDI/O for ND requests, and by the BSM for PC requests. These flags ensure that if a device is busy, it is allowed to complete its current task before it can be given another task.

Before processing a mailbox, PC-NDI/O checks the reservation flag for that device; if it is set to "reserved by DOS" status, it sets a "PC-NDI/O waiting" flag and puts the mailbox on a pending queue, using the pcqlink field of the mailbox to chain them together. After a time, the BIOS call returns to BSM, which then clears the reservation flag, and because the "PC-NDI/O waiting" flag is set, it calls PC-NDI/O. This results in the mailbox being taken off the pending queue and processed. Finally, control returns to BSM, which in turn returns to DOS.

The device reservation flags are also used to serialise multiple access to the same peripheral device from SINTRAN, which can occur for instance because both floppy and hard disks are handled by a single controller and BIOS call. After checking that a device is not reserved, PC-NDI/O sets the flag to "reserved by PC-NDI/O" and processes the mailbox. Any subsequent mailboxes arriving for that device are put on the pending queue, and are processed when the previous mailbox has been processed.

Print spooler The print spooler uses the DOS timer interrupt. It only does so if there are no other interrupts being serviced by the PC, i.e. PC-NDI/O is not active.

6.5.5.2 Device switch

The device switch module uses the PC identifier in the current mailbox, to switch execution to the appropriate code to handle the device involved.

6.5.5.3 Floppy disk handling

The floppy disk handling module performs the request encoded in the mailbox by the ND, and returns any resulting status/error code. This module allows ND-format floppy disks to be handled by the PC.

Standard Commands The following commands are accepted and executed by PC-NDI/O:

- read-data
- write-data
- find-eof
- write-eof
- format-floppy
- read-format
- read-deleted-record
- write-deleted-record
- format-track
- check-floppy

Deleted records The BIOS does not provide the necessary status bits to handle deleted records. The functions read-data, find-eof, write-eof, read-deleted-record and write-deleted-record use code in PC-NDI/O to access the floppy controller directly, bypassing the BIOS.

Byte ordering Because the ND-110 and Intel machines order their bytes within words differently, the byte-ordering on the floppies is also different. Because of this, PC-NDI/O does not DMA directly from floppy to ND memory. All transfers go via a one-sector buffer in the PC, where the byte ordering is reversed before transferring to the destination address in the ND.

BIOS floppy parameter block The PC BIOS contains a parameter block, pointed at by Int1Eh, which describes the type of floppy inserted. The types of floppy disk that can be handled are the standard PC format floppy and ND floppies type 0 and 17.

When a request from the ND is received, the BSM points INT1Eh at the parameters for an ND type 17 floppy, and restores the PC parameters when the transfer has completed. ND type 0 is only used for checking the format of an unreadable disk.

The parameter block values are:

Byte				
Offset	PC	ND-0	ND-17	Description
0	223	223	223	SRT=0, Hd unload=0: 1st specify byte
1	2	2	2	Hd load=1, mode=DMA:2nd specify byte
2	37	37	37	wait after operation until motor off
3	2	2	3	bytes/sector (2=512, 3=1024)
4	15	8	8	last sector on track
5	1BH	1BH	35H	gap length
6	FFH	FFH	FFH	DTLdt1
7	54H	54H	74H	gap length for format
8	F6H	40H	40H	fill byte for format
9	15	15	15	head settle time (milliseconds)
10	8	8	12	motor start time (1/8th seconds)

Activate autoload This function is used solely for booting from floppy. The first such request causes the first sector of the disk to be read and the first byte of this sector returned. Subsequent calls read the following bytes, until the whole sector has been read. Another call causes the next sector to be read from disk and the process repeats.

In practice only part of the first sector is ever read this way.

Test mode

The Z80-based controller on standard ND machines contains a host of test features, all of which are irrelevant for the PC floppy disk drive. However, some of the older bootstraps use one of the tests (read Z80 buffer) as a lazy way of getting in the bulk of the bootstrap.

For this reason, test 14 is handled by PC-NDI/O. The mailbox format for this is:

Word	Offset	Name	Use
	18	nd-addr	address to transfer to
	19	z80-addr	address to transfer from
	20	byte-count	no. of bytes to transfer

Z80-addr: 2200h gives the offset into a hypothetical buffer from which the transfer is to start. The buffer "starts" at the beginning of the sector that was last read in for activate-autoload. Thus any remaining bytes left in the buffer from previous activate-autoloads may be transferred to the ND, followed by subsequent sectors as necessary to satisfy "byte-count".

Error codes

Errors returned by the BIOS are translated into the equivalent ND error code before returning to the ND. The error mapping is as follows:

PC error code/description	ND error code/description
2 address mark not found	11 disk defect
3 write protect violation	16 write protected disk
4 sector not found	6 sector not found
8 DMA error	33 internal DMA errors
9 DMA boundary error	33 internal DMA errors
16 CRC error	5 CRC error
32 FDC error	75 floppy controller error
64 seek error	7 track not found
128 timeout	26 timeout

The floppy disk command block contains information showing how far a transfer had got before it failed. PC-NDI/O always sets these locations as if the transfer failed immediately - i.e. to the destination address and start word count.

6.5.5.4 Hard disk handling

The disk is partitioned into two areas for Butterfly-110:

- approximately 5Mbytes are allocated to the MS-DOS partition,
- the remainder is allocated to the SINTRAN partition.

The disk handling code in PC-NDI/O receives requests from the ND via mailbox, and performs the required functions on the SINTRAN partition.

Standard commands The following commands are accepted and executed by PC-NDI/O:

read-disk	
write-disk	
rpar-disk	read but do not transfer
comp-disk	
seek-disk	hd-word-cnt = no. of tracks to seek
fdisk-track	format single track
rzero-disk	reset disk and seek to sin-start cylinder

Block address calculation

The disk geometry assumed by SINTRAN is not the same as the disk geometry available on the PC. The disk geometry expected by SINTRAN is set up in the mailbox by the IOX-simulators. It is read the first time the disk is accessed. The variable `sin-cyl-size` is then calculated, giving the number of sectors per cylinder assumed by SINTRAN.

Also at start up, the BIOS is called to provide details of the actual disk geometry. These details are held in variables `pc-sides`, `pc-sectors-track` and `pc-cyl-size`.

When a disk access is requested, the (SINTRAN) absolute sector address is calculated by:

$$\text{abs-sector} = (\text{track-num} * \text{sectors-per-cyl}) + \text{sector-num} + (\text{side-num} * \text{sin-sectors-track})$$

This value is then multiplied by:

$$2^{**}(\text{sin-words-sector} / \text{pc-words-sector}) - 1$$

This changes the (SINTRAN) absolute sector number to a (PC) absolute sector number. Note that this calculation assumes that the SINTRAN sector size is either equal to or greater than the PC sector size by an integral power of 2.

This value is then converted to a real PC disk address by:

$$\begin{aligned} \text{track} &= (\text{abs-sector} / \text{pc-cyl-size}) + \text{sin-start} \\ \text{side} &= (\text{abs-sector} \% \text{pc-cyl-size}) / \text{pc-sector-track} \\ \text{sector} &= ((\text{abs-sector} \% \text{pc-cyl-size}) \% \text{pc-sector-track}) + 1 \end{aligned}$$

where `sin-start` is the cylinder offset to the start of the SINTRAN partition.

Disk types

Currently, two sources of hard disk are encoded so that PC-NDI/O can handle their disk geometry:

- type E7 (CDC):
 - 512 bytes/sector (4 sectors per SINTRAN page)
 - 17 sectors/track, numbered 1-17
 - 5 heads/cylinder, numbered 0-4
 - 989 cylinders, numbered 0-988
- type E9 (NEC):
 - 512 bytes/sector (4 sectors per SINTRAN page)
 - 17 sectors/track, numbered 1-17
 - 8 heads/cylinder, numbered 0-7
 - 615 cylinders, numbered 0-614.

For accesses to the SINTRAN partition, an appropriate "track offset" is added to the cylinder number: the track offset is the number of the cylinder which SINTRAN assumes is cylinder 0. With an MS-DOS partition of approximately 5Mbytes, then for CDC disks this value is 120; for NEC disks it is 80.

Bad track handling

A track is condemned as "bad" if one or more bits on that track are detected as bad. The bad disk tracks on a hard disk are detected when the disk is set up by the Diskprep program, which is run during installation - for further information, refer to the relevant Program Description sheets for HARDINI, ND-230011. Bad track details are held in the MS-DOS file C:\BFLY-110\BFLY\BADTRACK. This is a text file, with the first line giving the SINTRAN start cylinder, then one line per bad track entry, in the format:

```
<bad cyl><bad head><relocated cyl><relocated head>
```

They are read into PC-NDI/O memory at startup, and held in the badtrack table.

Before a disk transfer is started, PC-NDI/O checks to see if the transfer will involve more than one track. If so, the transfer is split so that a DMA operation never crosses track boundaries. In this way, if one track is bad the transfer can still be done.

When an error occurs, the bad track table is checked, and if the failed transfer was from a known bad track, the transfer is restarted using the remapped track.

Error mapping

Errors returned by the BIOS are coded into STWD1, and translated by the IOX-simulators into an ND error. The error mapping is as follows:

PC error code/description		ND description
1	invalid command	ignored
2	address mark not found	address mismatch
4	sector not found	address mismatch
5	reset failed	disk error
7	drive para. activity failed	disk error
9	DMA boundary	DMA error
10	bad sector	address mismatch
17	ECC corrected data	ignored
32	controller failure	disk error
64	seek failure	disk error
128	timeout	timeout
170	drive not ready	disk error
204	write fault	disk error
224	status error	disk error
240	compare error	compare error

6.5.5.5 PC console display handling

This module takes one character at a time from the display mailbox, and passes it to the DSP module (see section 6.6.2) for output.

6.5.5.6 PC console keyboard handling

Input from the PC keyboard is only of interest to PC-NDI/O when the Butterfly-110 is in ND mode. When this happens, the System Manager (see section 6.6) calls the ND-kernel via the System Communications Area (see section 6.4.9). This polls the keyboard. When input is received, it is placed in the appropriate mailbox and the ND then interrupted in order to take the character.

6.5.5.7 Serial port handling

Hardware configuration PC-NDI/O contains drivers for the serial ports on the PC side of Butterfly-110. These comprise:

- the standard Ericsson RS232 port, when this is used for a COSMOS connection
- the Serial Communications Controller (QuadPort with extension board), which provides an extra 5xRS232 ports and one parallel port.

Both use the INS 8250 serial communications chip.

The following table summarises the port configurations.

Port Name	Address	Interrupt	DOS	ND Terminal
Std. serial	3F8h	IRQ4	COM1 ¹	5
QuadPort F	288h	IRQ3	COM2 ²	-
QuadPort G	298h	"	-	9
QuadPort H	2A8h	"	-	10
QuadPort I	2B8h	"	-	11
QuadPort K	2C8h	"	-	12

Note that:

- 1 The COM1 port cannot be accessed from DOS while the Butterfly-110 PC system software is installed, since it is taken for use by ND's COSMOS asynchronous communications facility.
- 2 The COM2 port is not hardware compatible with IBM-PC, since a different address is used, but it can be accessed by BIOS calls and from MS-DOS. This is done by patching the address of the port (288h) into the BIOS communication area at 0040:0002h when PC-NDI/O is installed.

The Interrupt select port address is 2DBh, and is initialised with 01H to select IRQ3.

It is possible to add a further QuadPort board, to provide 5 more serial ports. In this case they should be configured to use interrupt IRQ4, and the PC's standard serial and parallel ports reallocated. However, this would require significant changes to the way ND-COSMOS (on the standard serial port) and a local printer (on the parallel port) are currently handled.

The physical identity of these ports is shown in fig 6.13.

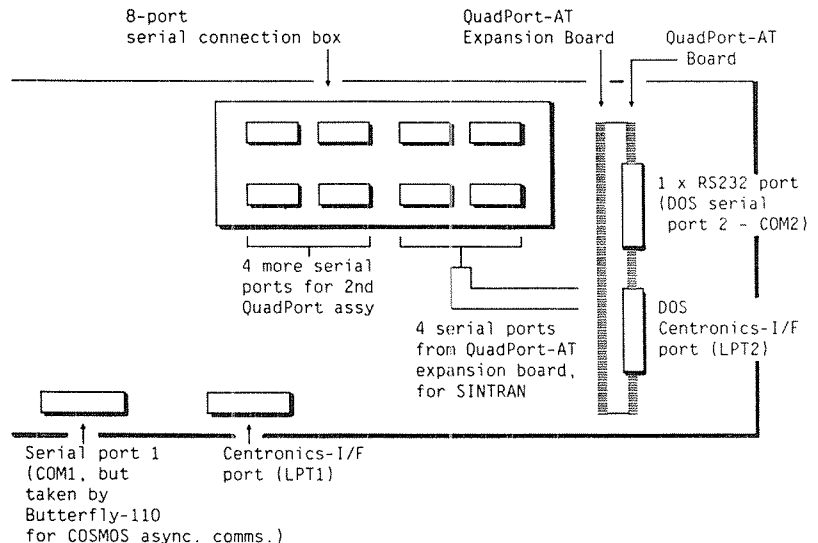


Fig 6.13: Butterfly-110 serial ports

**Interrupt
handlers**

There are two interrupt handling routines: one for IRQ3 (QuadPort) and one for IRQ4 (PC standard port). In both cases it is essential to service all sources of interrupt before the End of Interrupt (EOI) command is sent to the interrupt controller, because the controller is operated in edge triggered mode. If this were not done, the interrupt line would not return non-active, so the next pending interrupt would not be seen.

In the operating mode used, each 8250 Interrupt Controller chip has two sources of interrupt: "character received" and "transmit register empty". Both must be checked.

When an IRQ3 is serviced, the routine loops, serving ports until no further bits are set in the QuadPort interrupt request port.

**X-on/X-off
handling**

Receiving an X-off character (13H) from a device stops transmission from the associated port until an X-on character (11H) is received. X-on and X-off characters are not passed on to the ND.

Likewise, for input to Butterfly-110, an X-off character is sent when the mailbox ring buffer has space for less than 6 more characters. An X-on character is then sent when the buffer is empty.

Mailbox handling

As described in Chapter 3, there are two mailboxes per async port, one for input and one for output. Chapter 3 also includes a description of the data transfer and interrupt control mechanism for this type of mailbox.

6.5.5.8 Parallel port handling

This is similar to display handling (section 6.4.5.5), but instead of passing characters to the display handler, they are output to the parallel port.

Butterfly-110 fitted with the QuadPort card assembly provides two parallel ports, as follows:

Port	PC Interrupt number	MS-DOS name
std //	IRQ7	LPT1
QuadPort //	IRQ5	LPT2

SINTRAN accesses LPT1 as terminal 37.

6.5.6 System initialisation

This module is entered via INT65h from the supervisor package, when Butterfly-110 is warm-started or cold-started. It resets all important variables and mailboxes, and re-initialises the BSM. Following this, PC-NDI/O and the BSM are in the same state as after system start-up.

6.5.7 ND-kernel

This module "takes over" the system when the user selects ND mode. It is called by the System Manager, via an entry in the System Communications Area. It takes the form of an infinite loop which captures the PC, stopping any MS-DOS activity.

The loop performs two tests. Firstly, it checks for any existing input from the PC keyboard. If any is present, it calls the keyboard handling module and the input character(s) are sent to the ND. Secondly, it checks two flags in the System Communications Area, which are set if the SYS key has been pressed. If they are set, ND-kernel returns to the System manager. While this loop is executing, interrupts are ON, so requests (originating from the ND or the PC) can interrupt the loop and be processed.

6.5.8 System communications area

The System Communications Area exists within the PC-NDI/O program. It is a common data area which is used to pass information between modules in the PC side of the system. It is pointed to by INT62h, and each module is allocated a fixed offset from this location to its area of interest. These areas are shown in fig 6.14.

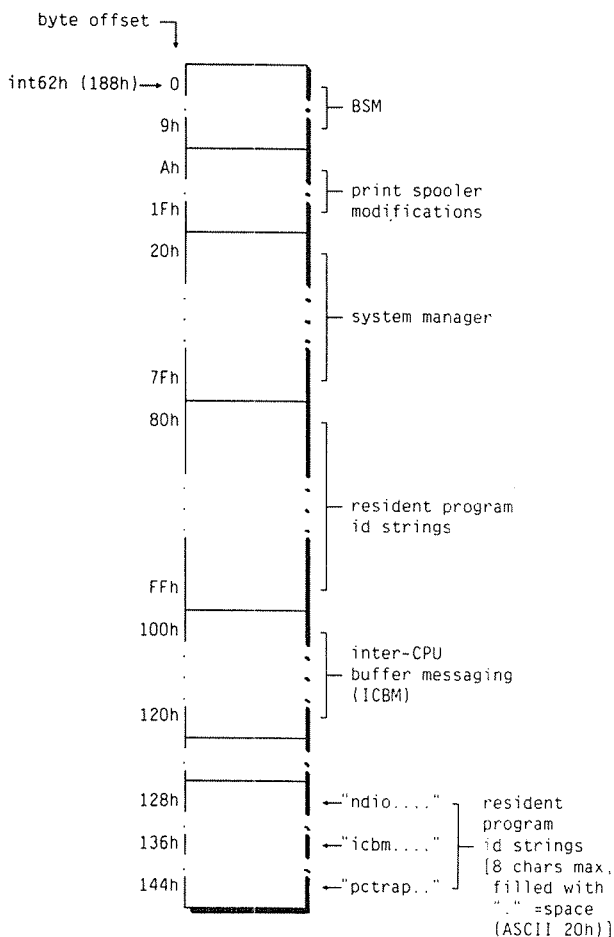


Fig 6.14: System communications area

The start of this system communications area contains device reservation flags, as shown in fig 6.15:

disk flag	scrn flag
ptr flag	in BIOS
NDIO wait	SYS

Disk flag = 0 if disk not reserved
 1 if disk reserved by BSM
 2 if disk reserved by PC-NDI/O

scrn flag = 0 if display not reserved
 1 if display reserved
 2 if display reserved by PC-NDI/O

ptr flag = 0 if printer not reserved
 1 if printer reserved
 2 if printer reserved by PC-NDI/O

in BIOS = 0 usually
 1 if BSM has called a BIOS routine

NDIO wait = 0 usually
 1 if PC-NDI/O is waiting for a device

Fig 6.15: System area - device reservation flags

Each module that can call the BSM ensures that it does not request an I/O transfer on a device that is already in use for another module.

The BIOS treats the hard disk and floppy disk as a single device i.e. it uses the same device driver. Thus a hard disk transfer cannot be allowed to interrupt a floppy disk transfer. The ND however treats them as separate devices. Code in PC-NDI/O therefore sets the same single flag for hard disk and floppy disk requests.

6.5.9 PC interrupts used

INT08h	(QuadPort) 4 x serial I/O
INT0Ch	COM1
INT0Dh	interrupt from ND
INT1Eh	floppy disk parameter block
INT60h	System Manager internal
INT62h	pointer to System Communications Area
INT63h	terminal 1 input
INT64h	terminal 1 output
INT65h	system initialisation
INT6Fh	file transfer internal

6.6 MULTITASKING PC SOFTWARE

The system running in the PC must appear from a PC-user viewpoint to be identical to that of a standard PC, so that existing PC application programs can run without modification. It must also function as an I/O processor for the ND, which may expect several I/O requests to be executed simultaneously. The PC must therefore function as a "normal" PC, but also handle several concurrent I/O requests.

6.6.1 MS-DOS and BIOS

The standard PC system software runs:

- the MS-DOS operating system
- the basic input/output system - BIOS.

MS-DOS provides the user interface and user-level facilities. This is loaded from the disk into PC memory, at start-up time.

The BIOS provides the device drivers which communicate directly with the PC's hardware devices. Higher level device drivers call the BIOS through interrupts in the range 10h-17h (see section 6.5.3.2). The BIOS also contains various start-up and test routines. It resides in PC ROM.

Access to the BIOS and DOS routines is made via software interrupts. These pick up the address of the appropriate interrupt server from interrupt vectors (see section 6.5.3) stored in the bottom 1k of PC-RAM.

The BIOS provides drivers for keyboard, display, hard disk and floppy disk, printer, and RS232 interfaces. Most of these drivers loop on the relevant hardware status register, waiting for the device to indicate that it is no longer busy.

The device interrupts are directed to an area in BIOS which accepts them but does not use them, except in the case of interrupts from the keyboard and disk drivers.

- The keyboard interrupt is directed to an interrupt handler which decodes the character and places it in a "look-ahead" buffer.

-
- The floppy disk interrupt is used to set a software status flag, because the hardware does not provide a suitable "busy" status bit. See section 6.4.9.

6.6.2 Additional requirements for Butterfly-110

Device requests to BIOS (see fig 6.1) may originate from:

- mailbox
- a PC application, or DOS.

A means of accepting these requests, queuing them, and assigning them appropriate priorities, is therefore necessary. This is described further in section 6.5.4.

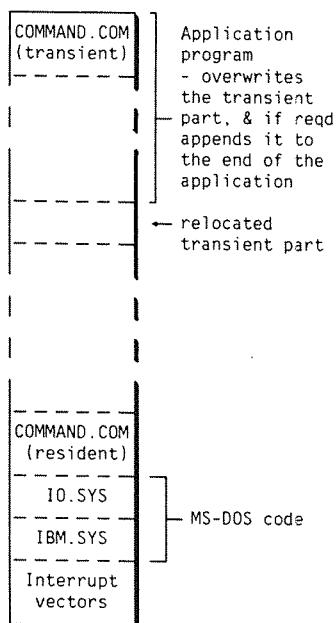
6.6.3 Prerequisite MS-DOS information

The additions to software on the PC side of Butterfly-110 are constrained by the structure and interrupt system of MS-DOS, and its interface with the BIOS. These constraints are outlined here.

6.6.3.1 MS-DOS structure

A typical MS-DOS memory map is shown in fig 6.16. The elements shown are the:

- interrupt vector table, which points to the start of the interrupt handler routine for respective interrupts.
- IO.SYS, which provides the MS-DOS device drivers to control the hardware, via the BIOS. In some cases, these device drivers access the hardware directly.
- IBM.SYS, containing interrupt handlers and service routines. These are independent of the hardware. Communication with device drivers is by CALL instructions. Access to DOS is through interrupts 20H and above.
- Resident and transient parts of COMMAND.COM. These conduct the communication between the user and the DOS operating system.



The resident part includes a number of commands which optimise performance, as well as interrupt handlers for interrupts 22H-24H.

The transient part is replaced (or if the application demands - relocated) when an application is loaded.

Upon exiting from a PC application, code in the resident part reloads the transient part.

Fig 6.16: Typical MS-DOS memory map

6.6.3.2 PC interrupt system

In the PC, hardware interrupts (fig 6.17) are input to the PC's type 8259 Programmable Interrupt Controllers (PIC). There are two PICs. On receiving an interrupt, they generate an interrupt to the 80286 microprocessor, and also output a PIC level. When the interrupt is serviced, the PIC level points to a location in the interrupt vector table (bottom 1k of RAM). This contains the address vector, which directs program to the start of the interrupt handler code appropriate to that interrupt. The PICs are attached to interrupts 8h-Fh and 70h-77h.

Refer to the System BIOS and MS-DOS technical manuals for the specification of:

- the fixed int/vector/BIOS entries relevant to the PICs
- the implementation-dependent int/vector/function relationships for interrupts 20H and above.

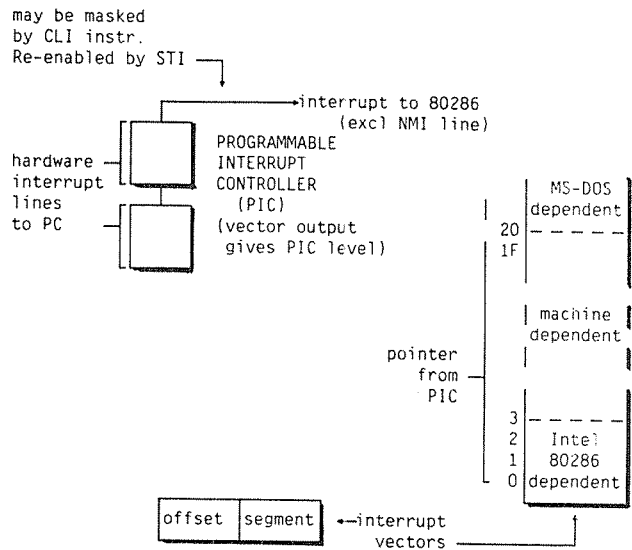


Fig 6.17: PC interrupt system

The highest level hardware interrupt is the "timer-tick", which occurs every $\frac{1}{8}$ sec. When interrupts are enabled, this initiates an "MS-DOS free - timer" sequence (fig 6.18), to poll various statuses and to service background tasks (e.g. BIOS, housekeeping, etc.).

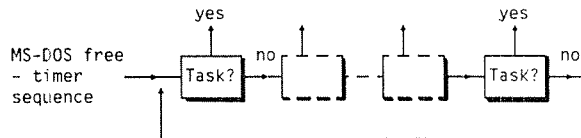


Fig 6.18: MS-DOS free - timer sequence

6.6.4 Multitasking

6.6.4.1 The sharing requirement

The term "multitasking" is not strictly correct when describing the sharing of facilities in the PC side of Butterfly-110. The system does not attempt to run two or more programs simultaneously; instead it runs a single program that may be interrupted in order to give more immediate service to a higher priority task (e.g. I/O operation). In this way it optimises performance.

The sharing requirement is described in sections 6.5.1 and 6.5.5.1.

6.6.4.2 PC-NDI/O multitasking control

Entry/exit for PC-NDI/O is described in section 6.5.1.

When PC-NDI/O is started, on interrupt from the ND, there is a start-up phase during which interrupts are turned off. During this phase, the relevant reservation flag(s) are checked. If they are free, PC-NDI/O sets them, sets

interrupts on, and processes the interrupt request. If the flag is already reserved, PC-NDI/O sets another location in the system communications area, indicating that it is waiting for a device. PC-NDI/O then "IRETs" to the previous task.

The process is summarised in fig 6.19.

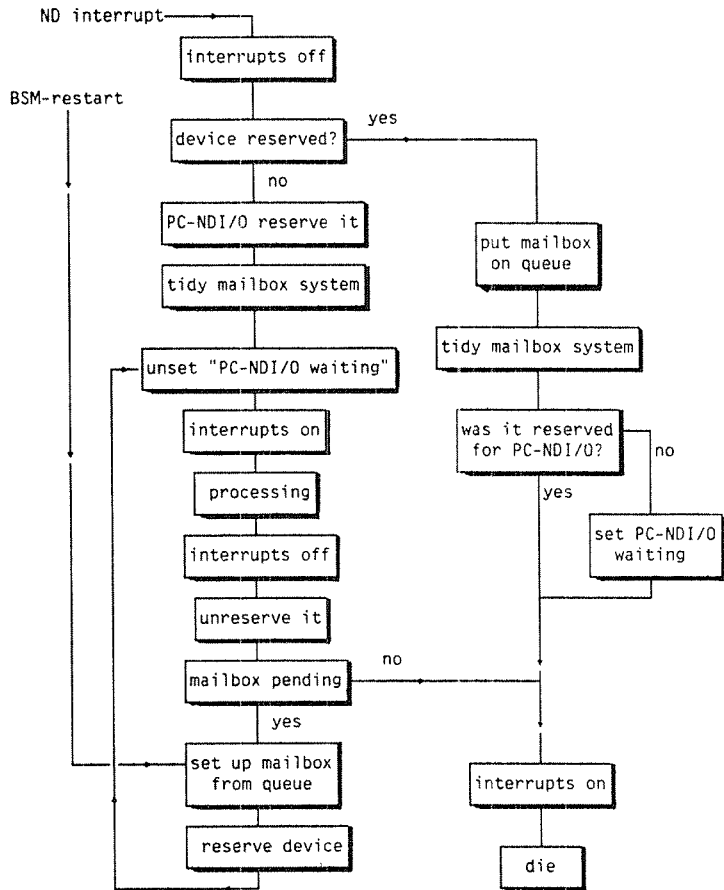


Fig 6.19: PC-NDI/O multitasking control

6.6.4.3 BSM to PC-NDI/O multitasking control

This control mechanism is outlined in section 6.5.5.1. It is represented in fig 6.20.

Whenever the BIOS sharing module is started, it checks the relevant reservation flag. If set to "reserved by PC-NDI/O", it continues. If not, it sets it to "reserved", then continues.

When the BSM finishes processing a request (i.e. when it is returned to from the BIOS), it again checks the relevant reservation flag. If set to "reserved by PC-NDI/O", it returns. Otherwise, it checks the "PC-NDI/O waiting" flag. If this is set, it generates an interrupt to start PC-NDI/O running (i.e. it generates a dummy PC-NDI/O request). PC-NDI/O will not have interrupted the PC since it last set the "PC-NDI/O waiting" flag, because the acknowledge for that request has not been transmitted.

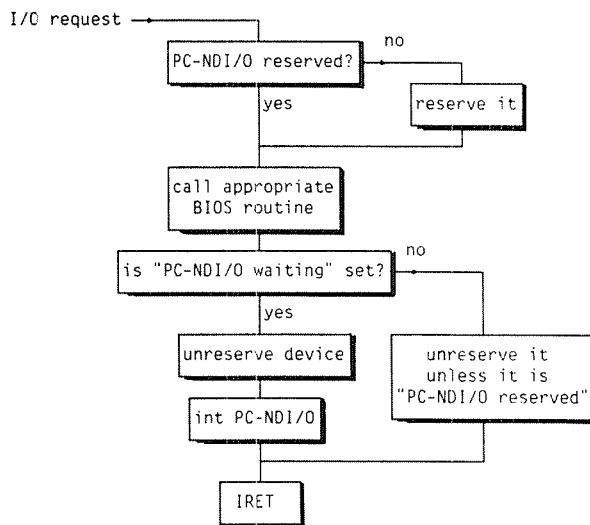


Fig 6.20: BSM multitasking control

The BIOS sharing module (BSM) allows sharing of BIOS, between PC-NDI/O and MS-DOS. It comprises two main parts:

- BSMINI
- BSM.

These are shown in fig 6.21.

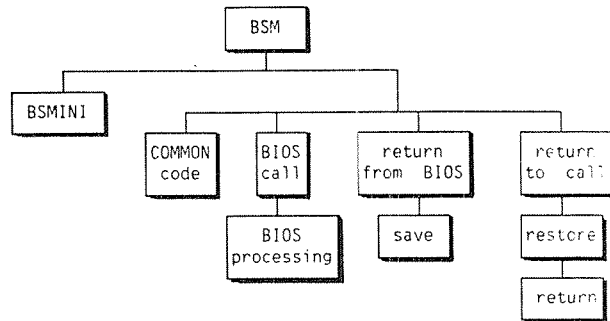


Fig 6.21: BSM modules

BSMINI saves the existing software interrupt vectors (for disk, display, printer), initialises BSM/PC-NDI/O communication flags, and sets the software interrupt vectors for disk, display and printer (to point to appropriate BSM module entry points).

Following issue of the BSM call to BIOS, interrupts are disabled.

On return from BIOS, the results of the call are saved (register values) if it was for a MS-DOS request. PC-NDI/O saves these for ND requests. Then the device flag is reset to zero, and an INT61h call made in order to service any pending ND requests.

6.6.4.4 Terminal manager multitasking

An "in BIOS" flag in the system communications area is set by BSM every time it calls a BIOS routine, and is reset when BIOS exits.

6.7 SYSTEM MANAGER

6.7.1 Overview

The system manager provides several broad functions. These include:

- the software interface to the keyboard, to convert incoming key code data into codes appropriate to the machine in use. This is performed by the keyboard driver.
- the mechanism for choosing which machine to use next. This is handled by the system manager.
- handling the output of the ND for screen manipulation. This is done by the display manager.

6.7.2 System modules

The system manager deals with the keyboard input, mouse input, display output, option and push-key setting, and switching between MS-DOS (PC) and SINTRAN (ND) environments.

These modules (fig 6.22) are all resident programs, loaded at start-up time. The keyboard and mouse input modules are used by all applications asking for input. The other modules are only used by the ND for output, or by supervisory programs.

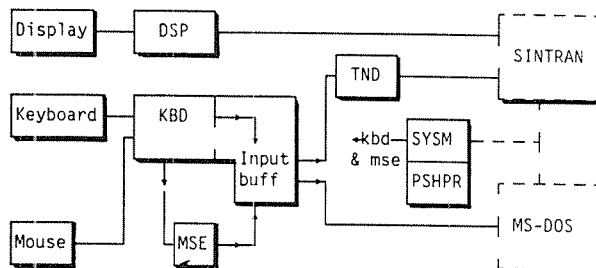


Fig 6.22: System manager structure

Display Manager (DSP)	DSP provides for ND output to the display. All ND-Tandberg display and keyboard lamp control sequences are accepted, suitable simulation being performed where a capability does not exist. ND terminal types 53 and 83 are supported. In a future release, the control sequences include graphics for NOTIS-BG and NOTIS-DRAW. Aside from accepting characters for display, it also inputs display attributes (colour, normal/inverse, underline, double-width) and lines per frame. It also traps ND output of push-key strings for programming push keys, passing these to the keyboard driver.
Keyboard Driver (KBD)	<p>The KBD module extends the functionality of the standard IBM-PC/AT-compatible keyboard handler. These extensions cope with the extra scan codes that are generated, and also with the mouse input. The extra scan codes are converted into a variation of the IBM Extended ASCII, so that all PC programs may read these keys if they wish. They are not converted into Tandberg sequences by this module - that function is done by TND.</p> <p>The keyboard driver also expands push-keys, so that they are transparent to most applications. It provides an increased size of input buffer over that of standard IBM-PC. As well as key codes, the KBD module outputs the SYS flag. It inputs push-key strings for programming of the push keys.</p>
Mouse Driver (MSE)	MSE takes input from the keyboard module, and interprets it as either cursor key-strokes or as movement of a free pointer. The particular mode is dependent on which application is asking for input. If it is an ND application or a non-graphic PC application, the mouse will normally be in "cursor" mode. If it is a graphic PC application, it will be in "free-pointer" mode. Inputs to the MSE module include free-pointer/cursor mode, and mouse button strings.
Tandberg-ND (TND)	<p>The TND key emulator is part of the Display Manager module. It interfaces between an application wanting Tandberg-like key-strokes, and the keyboard driver. It has two modes of operation:</p> <ul style="list-style-type: none">• numeric pad• extended control mode. <p>These may be toggled, either by an application sending the appropriate code to the display module, or by the User, via the parameter setting option of the System Supervisor utility.</p>
System Manager (SYSM)	When the SYS key is pressed, SYSM displays a menu, which provides the option to switch between MS-DOS and SINTRAN environments, and to program the push-keys or set up some standard switches. It indicates if a SYS is outstanding. Outputs include push-key strings, mouse button strings, colour attributes, extended control mode, numeric keypad mode, and mouse mode.

6.7.3 Keyboard compatibility

The keyboard driver has to meet a number of compatibility criteria, so that as many IBM-PC programs and ND programs as possible will run unaffected by the presence of the Butterfly-110 keyboard.

6.7.3.1 PC

The program-to-keyboard interface remains unchanged in the new Butterfly-110 keyboard driver, and the addition of extra features has no impact on this interface.

This interface incorporates the following:

INT9 This is the keyboard input interrupt, for incoming scan codes. The keyboard driver decodes each scan code into a 16-bit extended ASCII format, puts the result into the keyboard buffer if necessary, and sets the state of the "shift", "control", "alt" and "lock" keys in the variable "kb-flag".

INT16 This is the program interrupt for checking and reading characters from the keyboard buffer.

Some programs look directly at the keyboard buffer and kb-flags, i.e. not using INT16. They do this either by polling, or by intercepting INT9 and looking at them after the keyboard driver has done its job (e.g. Sidekick). Other programs modify these directly (e.g. IBM Professional Editor). The keyboard driver handles both these situations.

The special PC key-sequences Ctrl+Alt+Del (for PC warm-start) and Ctrl+Alt+Exit (for PC cold-start) retain their meanings in the PC. They are similarly available to the ND environment.

The IBM keyboard has a feature whereby "alt" + a keypad key produces an extended ASCII character in the keyboard buffer. This is a means of entering characters for which there is no key, such as the IBM graphic characters. It is also available in ND mode.

6.7.3.2 ND

The keyboard interface to the ND is in the form of 8-bit characters. For the standard characters, the normal ASCII codes are used. For the extended characters, i.e. HELP, EXIT, MARK, etc, the transmitted codes are dependent on the source device and are translated into NOTIS keys by the VTM tables. Consequently there are no fixed rules for the format. For compatibility reasons the form used by ND-Link is adopted.

The Tandberg keyboard has two soft switch settings that are of particular relevance to the compatibility question:

- "Extended Control Mode"
- "Numeric Keypad".

These switches respectively enable/disable Control Sequence Introducers (CSI) for the NOTIS function keys and the numeric keypad. They are set by the NOTIS applications and PED.

- The Extended Control Mode redefines the "special" NOTIS keys, to send escape sequences instead of control characters. Provision is included in the keyboard driver, to ensure that applications that do not use VTM and have not disabled "escape" are properly handled.
- The Numeric Pad Mode changes the meaning of the keys on the numeric pad, from numbers to escape sequences. They can then be identified independantly from the ASCII numbers. This feature is used by NOTIS-WP-M for its User Definable Keys (UDKs).

6.7.4 New keyboard features

Serial-I/O This feature is transparent outside the keyboard. Any input to it emerges as a stream of scan codes which are identical to normal keyboard input.

Mouse The mouse performs in different ways, according to the (ND or PC) environment.

PC The mouse interface is Microsoft format, i.e. it is a "free pointer" which moves a cursor, graphic or text, around the screen, and sends information when mouse keys are pressed and released. The interface mechanism from an application is through software interrupt INT33, with a parameter which specifies the function to be performed. There are around 16 functions available in the Microsoft interface. These include:

- initialising the mouse,
- receiving reports on its position and the status of the buttons, i.e. whether they are up or down, and the position at which they were last pressed,
- setting the form of the graphic and text cursors,
- specifying a handler for a number of events, such as button presses and releases.

ND The mouse emulates striking of "direction" keys, so that it may be used with existing applications. The term "direction" keys refers to movements left, right, up, down; it does not include diagonal movements:

i.e.



Total movement on the display is dependent on the speed of movement of the mouse.

The mouse has three buttons - SW1, SW2, SW3. These can be made use of in the ND environment by assigning keystrokes to them in the same way as push-keys. The strings associated with them will normally be of limited length, such as "carriage return", "home" and "mark".

Push-keys

There is one set of user-programmable push-keys. These are on the "unshift", "shift" and "control" states of keys P1-P8. This single set of 24 Push-Keys is available for use in both environments. A set of push-keys is also available for programming from an application, so that the user-defined keys are not lost. One or other set of push-keys is selected using the "local" mode from the Terminal Manager menu.

- The Push-Keys are programmed by putting the terminal into a local state, entering the keystrokes for the desired push-keys, then returning to the on-line state. This local state is accessible from the System Manager menu.
- If the user strikes a key while a push-key is being expanded, then that key stroke is stored until the expansion is complete.
- The user-defined push-key expansion strings are held in RAM area of the keyboard driver, so that they are not lost following power-off. They are also stored on hard disk (in \BFLY\AUTO.PSH), so that those programmed for the PC are not dependent on the availability of the ND.

The application-defined push-key expansion strings are stored in RAM in the keyboard driver only.

- Push-key expansion takes place before the keyboard input buffer, so that it is totally transparent to the PC and ND.

Extended keyboard buffer

The standard IBM-PC keyboard buffer only accommodates around 15 keystrokes. This is considered to be too small for acceptable margins in Butterfly-110, and so is extended by 48 for Butterfly-110. This extension is in addition to the standard buffer, because there are a number of programs that bypass the BIOS and DOS keyboard input routines, and directly read the driver's buffer.

6.7.5 System manager

System Manager					
1 ND User Environment					
2 ND SINTRAN					
3 PC MS-DOS					
4 Push-key programming					
5 Mouse programming					
6 Parameter setting					
7 Kill PC application					
... ..					
15	12	9	6	3	0

Fig 6.23: System Manager display window

The system manager is activated by the SYS key. On being called, it displays a menu (fig 6.23), and interacts with the user via the mouse and the keyboard. To allow use of a free pointer mouse at all times (regardless of the current mode), the keyboard and mouse modes are switchable independently.

The system manager controls the user dialogue when choosing the next environment to run. The choices are:

- to use the ND under USER ENVIRONMENT
- to use the ND under SINTRAN
- to use the PC under MS-DOS
- to return to where it came from
- to go into "local" mode, for push-key and mouse button programming, and parameter setting (if appropriate i.e. using the colour adaptor).

Also, it displays an indicator line showing the ND's current active program level (0-16)

Program level This displays a "1" for an active level, and blank for non-active level. When the ND is running, the display contains a characteristic pattern of "1s", in which either or both of levels 0 and 1 show a "1". If at any time a "1" is not displayed in levels 0 or 1, the ND is probably in a "hung" condition, in which case it requires a restart (Warm or Cold start) - see Chapter 7.

Local functions The following functions need to be performed in the "local" mode, accessible from the Terminal Manager:

- Artificial Colours. This is not relevant to a Butterfly-110 with a monochrome display. For colour display, the setting of the display mode and the colours is included in "local" mode.
- Tandberg/ND-Link style soft-switch parameter settings. Most of the features provided by these soft switches are concerned with line transmission characteristics. The soft switches provided include switching between monochrome and colour screen modes, key click, margin bell, and cursor type.

- Push-Key Programming. The push-key programming environment displays all the current programmed keys, and allows full editing of them. There are 24 programmable keys (8 push-keys in unshift, shift and control states) plus 6 programmable mouse buttons (3 sequences when pressed, and a further 3 when released). Non-printing characters are indicated by a special character.

Context variables In switching between the two environments, there are a number of context variables that need to be saved, or a state defined for them. The following is a list of the types of variables involved.

- The mouse state - its position, the cursor shape, and whether or not a cursor is visible.
- The screen state - which mode it is in, what colours and attributes prevail, what is the cursor position, and which page is on display.
- State of lamps and shift/lock keys. These are not changed on switching environments, unless special conditions are introduced.
- Type-ahead. The "SYS" request is serviced immediately. As a result, there may be characters left in the buffer. If so, they are interpreted in the new environment.

6.7.6 Keyboard Driver

The keyboard driver is structured so as to be largely table-driven, and common between the two environments. The table (fig 6.24) contains four entries for each incoming scan code. These are selected according to the state of the shift, control and alt keys, in reverse order of priority. The entries in the table consist of two bytes. The low byte is either the normal IBM ASCII conversion of the key, or a special code to identify one of a number of special cases. The high byte is used only by the special cases, and is dependent on the particular situation. In general, it is an index into another table, e.g. as for the ND-specific keys.

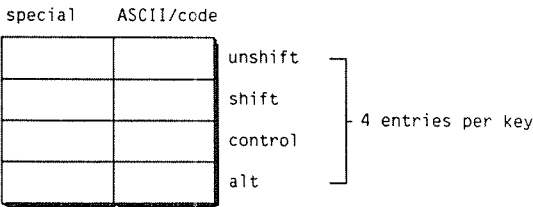


Fig 6.24: Keyboard driver table

The special cases are summarised below.

Normal IBM shift Right-shift, left-shift, control and alt perform a set/reset on bits in kb-flag, when they make/break.

Caps, ins, num-lock and scroll-lock, toggle bits in kb-flag and kb-flag-1, on make only.

Reset, break and print-screen, have specialised functions to be performed.

Alt-0 to alt-9, produce a decimal value in the range 0-255. This is inserted in the ASCII byte of the IBM extended ASCII characters, or as ND ASCII.

IBM specific keys These include most of the alt-shifted keys.

The high byte and a new low byte of 0 are appended to the keyboard buffer, to give the appropriate extended-ASCII code in the PC mode only.

ND specific keys These include mark, field, para, sent, word, copy, move, command, deftab, defbord, underline, justl/r, justa/r, justs/c, exp/app, linefeed, cancel, hyphen, prev, next.

A table lookup is performed, using the high byte to produce a two-byte code for the NOTIS key. This is then appended to the keyboard buffer, along with the necessary pre-amble and post-amble in the ND mode only.

Mixed IBM/ND These include backspace, F1-F10, ↓ ↑ ← →

A table lookup is performed, using the high byte (offset by 0/2, depending on ND or IBM) to produce the appropriate machine mapping of these keys.

	The function keys and the direction keys could insert the same characters into the buffer, with the VTM tables in the ND doing the conversion.
Push-keys P1-P8	The appropriate expansion string, if any, is appended to the keyboard buffer.
Mouse	<p>Key operations are alt-index (to toggle between right and left-handed use), index, middle and ring.</p> <p>Mouse movement and button make/breaks arrive from the keyboard as scan codes. These are decoded into two parts: one is a special code, the other a byte to tell the form of the data, movement or button press/release.</p> <p>Movement data is in the form:</p> <div style="text-align: center;"> <p>ALT-Index char char char</p> <pre> graph LR A[ALT-Index char char char] --> B[lead in] A --> C[3 digit range 0-7, to give 9-bit number] C --> D[9-bit number] D --> E[dir] D --> F[magnitude] </pre> </div> <p>Because of this format, once the lead-in character is seen, the next three characters <u>must</u> be directed to the mouse-driver.</p> <p>The mouse driver behaves in two ways:</p> <ul style="list-style-type: none"> • free pointer • cursor. <p>The former is for PC use, emulating the Microsoft Mouse interface. The latter is for ND use, emulating the striking of direction keys. This mode may be switched independently of the keyboard mode.</p> <p>When the mouse driver is in "free pointer" mode, movement of the mouse may be tied to a graphic or text cursor on the display, depending on the current mode of the display.</p> <p>When the mouse driver is in "cursor" mode, movement is converted into cursor keystrokes, with no direct tie to a screen cursor. In addition, button-presses may be converted into user defined keystrokes.</p>
Terminal-manager	SYS-REQ calls a routine which handles the interaction with a menu, and implements the selection.

Ext'd kbd buffer All PC program fetches of characters, must be from the standard 16 character buffer. Extension of this is provided using a local circular buffer, and filling the "real" buffer on one of the following events:

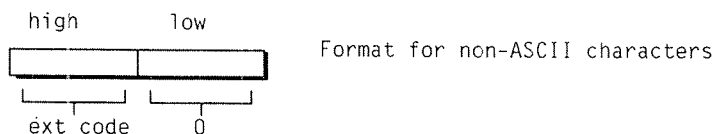
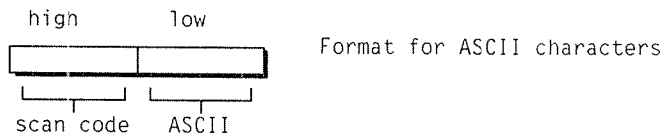
- new input (int9),
- character being read (int16),
- timer.

Language conversion is supported by the keyboard driver. On the PC, this is normally done by running a program such as "keybuk". This Butterfly-110 keyboard driver is a complete replacement for the standard PC keyboard driver. Therefore, to prevent the NOTIS facilities being overridden, ND language versions are provided in a similar fashion.

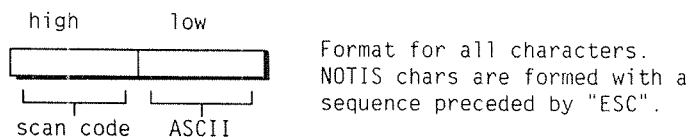
6.7.7 Character format in input buffer

All input, independent of mode, is entered into the same keyboard input buffer. The two modes have a slightly different format for the characters inserted in the buffer.

The PC format is:



The ND format is:



In normal use, only the low byte is used. The high byte is provided for use by the push-key programming module, in order that it can correctly identify NOTIS key sequences.

6.8 ND SERVERS

6.8.1 ND file services available to the PC

Servers have been added to Butterfly-110's SINTRAN (fig 6.25), to accept and execute a selection of MS-DOS system calls, in response to requests from programs running under MS-DOS.

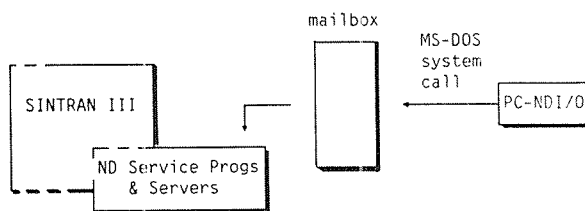


Fig 6.25: ND servers for the PC

The subset of MS-DOS system calls are:

- 1 3CH - Create Handle
- 2 3DH - Open Handle
- 3 3EH - Close Handle
- 4 3FH - Read Handle
- 5 40H - Write Handle
- 6 41H - Delete Directory Entry
- 7 42H - Move File Pointer
- 8 57H - Get/Set Date/Time
- 9 5BH - Create New File

These are described further in section 6.7.4

6.8.2 Communication between ND and PC

Software design in Butterfly-110 is such that the ND-110PCX accesses the PC only by means of IOX/IOXT and IDENT instructions, via the mailboxes. Code in "secret memory" simulates the behaviour of real devices. In this way the system is completely transparent to software running on ND-110PCX.

The design of the ND fileserver software which serves system function calls from DOS is similarly transparent.

6.8.2.1 Extensions to MON 144 (MAGTP)

Data to be passed between the servers on the two processors (ND-110PCX and 80286) is usually in blocks of between 10 and 1000 words (all words are 16-bit), and the normal SINTRAN III monitor call to access block devices is MON 144 (MAGTP). The call MAGTP has therefore been extended to access the PC.

These extensions are in the form of changes to the files SIN1-GEN, SIN2-GEN, SIN3-GEN, SIN4-GEN and MRES-SIN2-GEN, and patches to SINTRAN in the WORK-MODE file. Thus, the patches will be loaded from the SINTRAN disk, not run afterwards.

A summary of these changes is given below. The resulting allocation of ND memory space is shown in fig 6.26.

Butterfly DMA devices

Five "Butterfly DMA devices" have been created. These have datafields with Logical Device Numbers (LDNs) 1764, 1772, 1773, 2170, 2171, for devices 1, 2, 3, 4, 5 respectively.

The devices have device numbers 34-37, 40-43, 44-47, 50-53, 54-57, and ident codes 30, 31, 32, 33, 34, for the devices 1, 2, 3, 4, 5 respectively.

Unit datafields

Each DMA datafield has an input and output datafield for Unit 0. These have Logical Device Numbers (LDNs) 1743, 1744, 1745, 1746 and 1747, respectively.

RT programs

To make the devices available from foreground programs, five RWRT programs RWRT50, RWRT51,, RWRT54 have been created, together with their respective DF datafields with LDNs 1765, 1766, 1767, 1770, 1771.

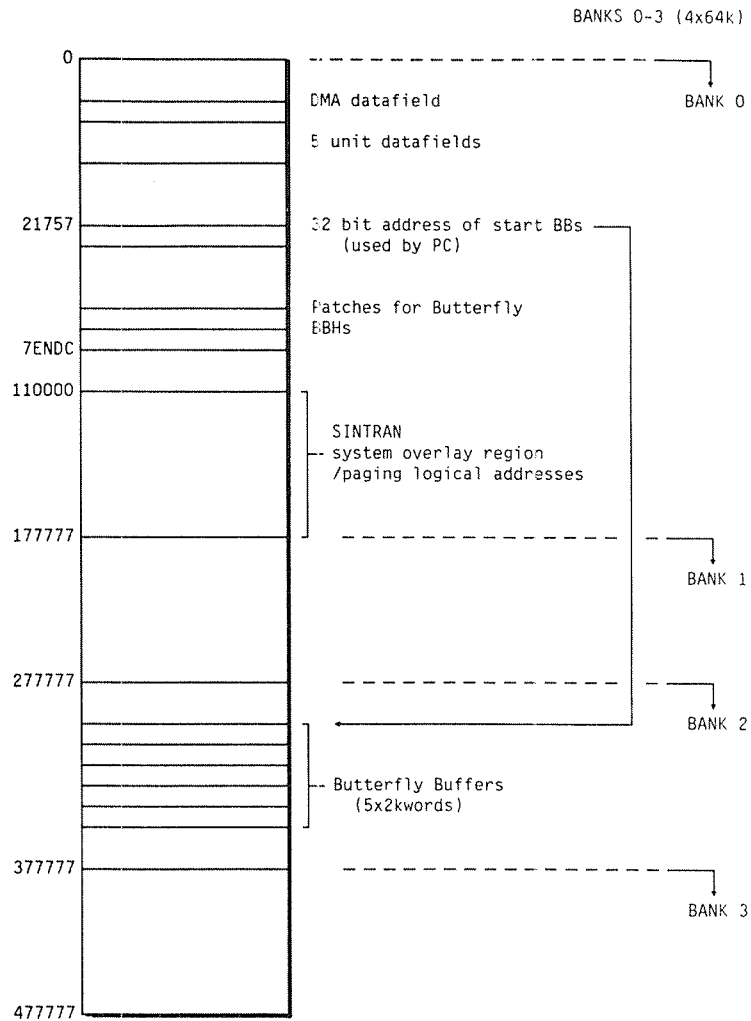


Fig 6.26: ND servers - allocation of ND memory space

There is a one-to-one correspondence between a server on the ND-110PCX, a Butterfly Device unit number, the buffer in ND-100 memory, and a server channel (in the ICBM, see Chapter 3 section 3.9.2) on the PC. Initially, two server pairs (one on the ND-110PCX and one on the PC) are created, as follows:

- Extension of the remote file access on ND-110PCX (PCFSERV)
- PC file access server (ICBM).

The numbers refer to the Butterfly unit used by the pair, and the RT-program names refer to the ND-110PCX.

A future release of Butterfly-110 software may include new server-pairs, for such purposes as a Monitor Call server, and a System Call server.

All file access on the ND-110PCX goes to a user specified in the file name, or if absent, to a default user which is RT-USER.

All these RT-programs share common code on the same segment, and local data organised around a base field.

Butterfly buffers Each unit has an associated Butterfly Buffer (BB) and Butterfly Buffer Header (BBH). These are analogous to device buffers and device buffer headers, except that each buffer is 2kwords long, and is statically associated with the corresponding Butterfly unit. The maximum data block that may be read/written is 2kwords.

Functions The following functions are available on a Butterfly unit:

- 0 - read block
- 1 - write block
- 21 - clear device
- 26 - read block (length in bytes)
- 27 - write block (length in bytes)
- 73 - an atomic write followed by a read.

Function 73 is converted by MON 144 to a write followed by a read, but appears as a single function to the user software.

Data path All data is passed through the Butterfly Buffers (BBs). Only "device registers" are passed through the mailbox system.

Buffer headers Within each buffer, data transferred always starts with a header which has the following layout.

<u>Word</u>	<u>Mnemonic</u>	<u>Contents</u>
0	NDPRO	ND-110PCX process identifier (rt-decription)
1	TRAID	transaction identifier
2	PCPRO	PC indentifier
3	SFUNC	function to be performed
4	ERRND	ND file system error code
5	ERRPC	PC error code
6	NUMBY	Number of bytes in buffer (in case of EOF or filename)
7	NBLKN	} unspecified
10	BLKLE	
11	NFILN	
12	BPTR	Pointer to data buffer (Relative to start)
.		
.		
.		
n	BUFR	data buffer of up to 4kbytes

Data from the accessed file is copied into the relevant Butterfly Buffer (BB), and then on to the destination program.

There is a one-to-one relationship between a BB and a Butterfly device unit. This is essential because the PC and ND-110PCX processors operate asynchronously on these buffers.

6.8.2.2 Status registers

The interaction between the ND-110PCX and the PC is effected by means of SINTRAN's standard structure of MTRANS, CTR... and DRIVER routines. This is modelled very closely on the disk and mag-tape drivers. In a similar fashion to the disk sub-system, the Butterfly is given several "registers". These are used for signalling and returning status between the different device units (and their associated servers) on the PC and ND-110PCX. They are part of the mailbox structure - see section 6.2.5.

6.8.3 File access requests from PC to ND-servers

Standard PC programs - which have no special knowledge of the Butterfly-110 and its ND-110PCX processor (with its own file system) - may make system calls to MS-DOS in the normal way. Only when the characters "ND:" precede the filename will a server be used to obtain access via the ND filesystem. The file access requests which the ND must handle for such programs are outlined in this section.

Every such request from the PC will be transferred via a Butterfly Buffer (BB). For each request, a brief functional description, and summary of the header values for FUNC and ERR codes, is given below.

NOTE

All values in the following headers are in octal.

Create handle
3Ch

The ND creates the file as an indexed sequential file, and opens it for random read or random read/write, according to the access parameter. Also, the file is flagged to be closed by MON BCLOS if the archive bit is set. If the file already exists, it is deleted and re-created.

Buffer SFUNC on request from PC:

SFUNC 0 for create file.

Buffer SFUNC, ERRND and ERRPC on return from ND:

SFUNC 101 - STATUS returned from ND.
ERRND SINTRAN III error number if error, else 0.
ERRPC 0 - OK.
3 - invalid file name.
4 - too many open files.
5 - attributes specified were unobtainable.

Create new file (5Bh)	<p>This is the same as create file, except that if the file already exists, an error is returned.</p> <p>Buffer SFUNC on request from PC:</p> <p>SFUNC 7 for create new file.</p> <p>Buffer SFUNC, ERRND and ERRPC on return from ND:</p> <p>SFUNC 101 - STATUS returned from ND. ERRND SINTRAN III error number if error, else 0. ERRPC 0 - OK. 3 - invalid file name. 4 - too many open files. 5 - attributes specified were unobtainable.</p>
Open handle (3Dh)	<p>The ND opens the file for random read or random read/write operations, according to the access parameter.</p> <p>Buffer SFUNC on request from PC:</p> <p>SFUNC 1 for open file.</p> <p>Buffer SFUNC, ERRND and ERRPC on return from ND:</p> <p>SFUNC 101 - STATUS returned from ND. ERRND SINTRAN III error number if error, else 0. ERRPC 0 - OK. 2 - file not found. 4 - too many open files. 5 - access denied.</p>
Read handle (3Fh)	<p>The server on the ND attempts to read a block of data specified by NUMBY (maximum number of bytes = 2048) from the specified open file number, starting at the (internally held) current byte pointer. The current byte pointer is updated by this operation. On reaching the end of file, both ERRND and ERRPC are set to 3, and the actual number of bytes read is put in NUMBY.</p> <p>Buffer SFUNC on request from PC:</p> <p>SFUNC 11 for read file.</p> <p>Buffer SFUNC, ERRND and ERRPC on return from ND:</p> <p>SFUNC 101 - STATUS returned from ND. ERRND SINTRAN III error number if error, else 0. ERRPC 0 - OK. 3 - end of file. 5 - access denied. 6 - invalid file number.</p>

Write handle (40h)	<p>The server on the ND attempts to write a block of data specified by NUMBY (maximum number of bytes = 2048) from the specified open file number, starting at the (internally held) current byte pointer. The current byte pointer is updated by this operation.</p> <p>Buffer SFUNC on request from PC:</p> <p>SFUNC 12 for write file.</p> <p>Buffer SFUNC, ERRND and ERRPC on return from ND:</p> <p>SFUNC 101 - STATUS returned from ND. ERRND SINTRAN III error number if error else 0. ERRPC 0 - OK. 5 - access denied - file not opened for write. 6 - invalid file number.</p>
Close handle (3Eh)	<p>The server on the ND attempts to close the file with the specified open file number. If the archive bit is set, MON BCLOS is used.</p> <p>Buffer SFUNC on request from PC:</p> <p>SFUNC 13 for close file.</p> <p>Buffer SFUNC, ERRND and ERRPC on return from ND:</p> <p>SFUNC 101 - STATUS returned from ND. ERRND SINTRAN III error number if error else 0. ERRPC 0 - OK. 6 - invalid file number.</p>
Delete file (41h)	<p>The server on the ND attempts to delete the file specified by the file name string in the data buffer.</p> <p>Buffer SFUNC on request from PC:</p> <p>SFUNC 2 for delete file.</p> <p>Buffer SFUNC, ERRND and ERRPC on return from ND:</p> <p>SFUNC 101 - STATUS returned from ND. ERRND SINTRAN III error number if error else 0. ERRPC 0 - OK. 2 - file not found. 5 - file was read only.</p>

Move file pointer (42h) The server on the ND changes the current byte pointer associated with the specified open file, by the relative amount specified in the parameter list. The change is relative to the beginning or end of file, or relative to the current byte pointer.

Buffer SFUNC on request from PC:

SFUNC 14 for move file pointer.

Buffer SFUNC, ERRND and ERRPC on return from ND:

SFUNC 101 - STATUS returned from ND.

ERRND SINTRAN III error number if error else 0.

ERRPC 0 - OK.

6 - invalid file number.

Get Date and Time of Last Write to file (57h) The server on the ND returns the date and time the file was last written.

Buffer SFUNC on request from PC:

SFUNC 17 for get date and time of last write.

Buffer SFUNC, EFRND and ERRPC on return from ND:

SFUNC 101 - STATUS returned from ND.

ERRND SINTRAN III error number if error else 0.

ERRPC 0 - OK.

6 - file not open

Set Date and Time of Last Write to File (57h) The server on the ND sets the date and time the file was last written.

Buffer SFUNC on request from PC:

SFUNC 20 for set date and time of last write.

Buffer SFUNC, ERRND and ERRPC on return from ND:

SFUNC 101 - STATUS returned from ND.

ERRND SINTRAN III error number if error else 0.

ERRPC 0 - OK.

6 - file not open.

6.9 PC SERVICE PROGRAMS

6.9.1 File access

This Butterfly-110 file access software allows access to SINTRAN files from DOS. The functions provided include open, close, read, write and create, and thus provide the basis of a utility for copying files between DOS and SINTRAN, in both directions.

6.9.1.1 Facilities

The file access software consists of two main elements (fig 6.1):

- a resident program in the PC. This comprises the PC trap handler, and ICBM.
- a server program in the ND-110 which access the SINTRAN file system on behalf of the PC user. See section 6.7.

Together, they extend the range of the DOS filing system, to include SINTRAN files.

The PC trap handler filters calls to DOS made via INT 21H, to divert filenames which begin with "ND:", or file handles which have been returned from the ND-110. Thus the PC application need not be aware of which file system is being used.

File Names

The PC program simply passes the string following the "ND:" to the server program running under SINTRAN. The filename syntax accepted is:

- 1 normal Sintran file syntax:

[<directory>:<user>]<filename>;<type>;<version>

The default user is RT. Files accessed in this way must allow RT the appropriate access rights.

2 COSMOS remote file access syntax:

```
<machine>{<user>{<password>}}.(directory:user)<filename>:<type>;
<version>
```

If the machine name is omitted the local Butterfly-110 is assumed. In this way local files may be accessed without making them available to user RT.

Note that when creating files it is not necessary to put double quotation marks round the file name, and that the default "type" of a file so created will be :DATA.

File data formats DOS and SINTRAN store their file data in different ways. In particular, least significant and most significant bytes within a word are ordered differently, and 7-bit SINTRAN files have even parity.

Butterfly-110's file access facility transfers byte data so that it is ordered correctly, but other types (word, double word, or combinations) are not catered for.

To overcome the SINTRAN 7-bit files parity problem, the parity bit has to be stripped from a 7-bit SINTRAN file when passing it from SINTRAN to DOS, and must be regenerated when passing the file from DOS back to SINTRAN. To specify this type of transfer, an extension to the file type has been added, in which the 4-character type may now be followed by a minus (-) and a number. Currently the valid numbers are

0 or no number	: 8-bit transfer
7	: clear the parity bit
8	: generate even parity
9	: generate odd parity.

Thus to copy a 7-bit text file from SINTRAN to DOS, the SINTRAN file must be opened with the file name in the form of the following example:

```
nd:(system(system)).fred:text-7
```

To copy a text file from DOS to SINTRAN, the SINTRAN file must be opened or created with the filename in the form:

```
nd:(system(system)).new-fred:text-8
```

If the file type is specified as "-0", or the "-N" part is omitted altogether, the data is transferred without change.

Note that it is still possible to access SINTRAN files with the valid file type "-7", for example, by specifying "-7-0" if no data transformation is required, or "-7-7", "-7-8", "-7-9" for one of the above transformations.

Error handling An error resulting from a call to file access may be from one of three sources: DOS, SINTRAN or the file access programs. It may be differentiated as follows:

0 to 88	- DOS
200 to 1023	- file access
1024 upwards	- SINTRAN.

An extra function is provided which returns a pointer to a string describing the error.

Registers when called:

AH	= OFFH (function code)
BX	Error number

Registers on return:

DS:DX	Pointer to an ASCII string.
-------	-----------------------------

For DOS errors, the string is that given in the MS-DOS Reference Manual (ND-60.271). For file access error messages, see section 6.8.2.3. SINTRAN error messages are given in the SINTRAN Reference Manual (ND-60.128).

For SINTRAN errors, the text of the error message is passed from the ND server when the error occurs, so that it is only possible to return an error string for the last reported SINTRAN error.

6.9.1.2 PC trap handler

The PC trap handler is a resident program running in the PC. It is installed on INT21h - the DOS function request interrupt.

The task of the PC trap handler is to filter the file requests presented from a PC-user program, such that all requests relating to ND files are diverted to the Inter-CPU Buffer Messaging (ICBM) module, rather than being passed to the normal MS-DOS INT21h routine. All non-ND file requests, and all non-filing requests, pass through the filter to the normal INT21h routine.

Trapped functions A subset of the DOS INT 21 functions are filtered for SINTRAN references. These, and their deviations from standard DOS usage, are listed here. For a full definition of the function and the parameters used, refer to the MS-DOS Reference Manual (ND-60.271).

In principle, any DOS application which uses the subset of functions implemented should be able to access SINTRAN files. If an application uses functions not yet implemented (for example changing the current directory, or using File Control Blocks - FCBs), then file access cannot process it. Problems may also arise if assumptions are made about the length or structure of file names, or where the translation of the function into SINTRAN is not exact. In practice, it has been found that few standard DOS utilities can access SINTRAN files ("type" is an exception). In most cases it will be necessary to write a new utility which uses only the functions listed here.

DOS call that may access SINTRAN	Notes
3CH Create Handle	Of the DOS attributes, only the following are passed to SINTRAN 0 (normal) 1 (read only)
3DH Open Handle	The access code (lower 4 bits of CX register) is translated: 0 (read only) → SINTRAN code 3 non-zero → SINTRAN code 2
3EH Close Handle	
3FH Read Handle	
40H Write Handle	
41H Delete Directory Entry	
42H Move File Pointer	
57H Get/Set Date/Time	Get or set the time last written of a SINTRAN file
5BH Create New File	

The calling routine interprets the return registers, such that the INT21h returns as would be expected from a normal (non-ND filing) function request. It then performs an IRET to the calling program.

6.9.1.3 Inter-CPU buffer messaging

ICBM (inter-CPU buffer messaging) enables blocks of information to be passed between the ND and PC processors in Butterfly-110. The ICBM is the PC half of the package, the other being the "Butterfly device drivers" (see section 6.7, but note that the buffer headers described are necessary for file transfer only). Together, they allow application programs running in the PC to communicate with application programs running in the ND.

Operation

The ICBM is called by one of the user-defined interrupt vectors, INT 6FH, passing the parameters in registers as defined in ICBM Interface, below. Information is passed in "Butterfly buffers", which are un-structured data areas 4096 bytes long. There are 5 such buffers, which are differentiated on the PC side by specifying a channel number from 0 to 4, and on the ND side by device number. Channels 0 and 1 are reserved for the Butterfly file access programs.

The basic functions are write data to a channel and read data from a channel. Typically, the two application programs will be synchronised, each alternately writing and reading from the channel.

Because DOS and SINTRAN use different ordering of bytes in a word, both read and write operations take an extra parameter which specifies whether byte or word data is to be transferred. This saves the applications from having to do their own byte swapping. If the data is neither byte nor word, or is a mixture of both, word data should be specified (as this is more efficient), and the applications must then perform the necessary manipulations.

ICBM is a resident DOS program which is normally loaded when Butterfly-110 is booted. A program can check that it is resident by looking for its "ident string", which is at the location pointed to by interrupt vector 62H (the PC-NDI/O communications area) plus an offset of 136 decimal. The 8-character string "icbm..." is written at this location at load time.

ICBM interface**Write data**

Call: AH = 0 (function code for write data)
DS:DI Address of buffer containing data.
CX Number of bytes in the buffer (<= 4096)
AL Channel number (0 to 4)
BL 0 for byte data, non-zero for word data

Return: AX Error code (see below)

Comments: Data is copied from the callers buffer into a Butterfly buffer, and a mailbox is sent to the ND to notify the event. The calling program resumes as soon as the message has been sent, which may be before it has been received.

Read data

Call: AH = 1 (function code for read data)
DS:DI Address of buffer into which data is to be read
CX Number of bytes to be read (<= 4096)
AL Channel number (0 to 4)
BL 0 for byte data, non-zero for word data
DX Timeout (in seconds)

Return: AX Error code (see below)
CX The number of bytes read

Comments: If an un-read buffer is waiting on the specified channel it is copied into the callers buffer and the calling program resumes immediately. If not, the calling program is blocked until either a buffer is received or the timeout expires. A timeout of zero will ensure that the call returns immediately.

The number of bytes transferred is the lesser of the number of bytes written and the number of bytes requested.

Clear channel

Call: AH = 3 (function code for clear channel)
AL Channel number (0 to 4)

Comments: Any partially complete transactions on this channel are aborted. This allows recovery from situations such as a program being aborted and restarted, or the ND being reset from the supervisor menu.

Error codes If an error is detected, an error code is returned in AX and the carry flag is set, otherwise AX contains zero and the carry flag is clear.

Error code (decimal)	Description
201	Invalid function. The function code in AH was not 0, 1 or 3.
202	Invalid channel. The channel number was not in the range 0 to 4.
203	Channel busy. Already dealing with a write from the PC.
204	ND server not started. No ND program has declared an interest in this channel.
205	Invalid transfer length. The number of bytes specified for a read or a write was greater than 4096.
206	Timed-out. No data was received in the specified timeout period.

6.9.2 File transfer program (DUPLI)

The DUPLIcate command provides a basic file copying facility to the user under MS-DOS. No SINTRAN version of this command currently exists.

A functional description of DUPLI is given in Chapter 3 section 3.3.10.

Operation In MS-DOS terms, the DUPLIcate program is an External command.

If a filename is missing from the command line the user is prompted for it.

If the destination file already exists it is over-written, but if it does not exist it is created. The well-known SINTRAN syntax for creating and using a filename as a destination filename, by placing double quotation marks around that filename, does not apply to DUPLI and will result in an error.

The "last write date and time" of the destination file is set to the last write date and time of the source file. In the case of SINTRAN, the destination file has the creation date and time and the last read date and time set to the date and time of DUPLICating to that destination file.

Where the destination file is a SINTRAN filename with the file type unspecified, the default type "DATA" is attributed.

When a SINTRAN filename is specified without machine, pack and user information, then the filename applies to the default user "RT-USER".

SINTRAN filenames can be abbreviated (if they already exist) in the same way as SINTRAN allows.

The message "@ bytes copied" is displayed after successful DUPLICation of a file.

Environment

The MS-DOS/SINTRAN file server programs (namely ICBM.EXE and PCTRAP.EXE) must be resident. They are loaded during the Butterfly-110 startup phase (controlled by the ND-110.BAT file).

The SINTRAN/MS-DOS RT program "PCFSERV" must be active; it is installed as an RT process during SINTRAN startup (controlled by the LOAD-MODE[:MODE] file).

Limitations

Wildcard characters are not permitted in filenames.

Due to restrictions inherent in the MS-DOS command line handler, the maximum number of characters available on the command line is 128. This restriction renders "DUPLI filename to/from filename" formats unusable when the filenames are so long that the limit is exceeded. (This can occur because SINTRAN filenames can involve up to 256 characters.) In these cases, the command format where only "DUPLI" is entered, and the program then prompts the user for source and destination filenames, provides for the entry of long filenames.

Error messages Error messages are copied to the MS-DOS standard output. The messages may originate from four sources:

- DUPLI program
- MS-DOS
- MS-DOS/SINTRAN File Server program ICBM
- SINTRAN/MS-DOS File Server program PCFSERV

DUPLI program error messages;

These messages are associated with errors made in entering the format of the DUPLI command, plus one message that indicates the MS-DOS/SINTRAN file server is not resident, and another message indicating that the destination device is full.

- Usage: DUPLI source TO destination
or DUPLI destination FROM source
- Too many parameters
- Source and destination not specified
- Source not specified
- Destination not specified
- TO/FROM directive not specified
- Invalid directive specified
- Cannot copy file to itself
- File Server not resident
- Destination device full, not enough space to complete

MS-DOS error messages

For the complete list of MS-DOS device error messages, see the MS-DOS Reference Manual (ND-60.271).

ICBM error messages

See section 6.8.2.3.

PCFSERV error messages

The error messages from the PCFSERV RT program are in upper case, and in this way are distinguishable from the other messages. SINTRAN error messages are given in the SINTRAN Reference Manual (ND-60.128). Note only the "explanatory text" part of the error messages is copied to the MS-DOS standard output.

6.9.3 Print spooler service program (PSSP)

The Print utility in MS-DOS V3.10 is modified for Butterfly-110. These modifications enable the print spooler to accommodate spooling of ND files alongside the standard facility for spooling PC files.

The changes in functionality are as follows:

- at interrupt level, it forces a new file selection after a ND-user has finished spooling.
- at interrupt level, it detects if PC-NDI/O is waiting to spool. If so, it forces the spooler to allow it to spool after printing the current PC file.
- at interrupt level, it prevents a MS-DOS timer interrupt from interrupting a PC-NDI/O request currently being processed for the same device.
- at non-interrupt level, it allows cancellation and termination of files in the print queue and currently being printed, while also maintaining the correct interfacing with PC-NDI/O spooling.
- at non-interrupt level, it stops messages relating to cancelled and terminated files appearing on the printer when PC-NDI/O is spooling.
- It stops print screen calls (INT5h) operating when PC-NDI/O spooling is taking place.

See also section 6.4.5.1 ("request queuing").

6.9.4 PC servers

The software structure block diagram in fig 6.1 indicates that a future release of Butterfly-110 software could include PC servers. Current design requirements suggest that this is unlikely to be implemented.

The original purpose of PC servers was to enable ND programs to make file access requests to the PC filesystem. These requests would originate from a modified version of ND's COSMOS remote file access code. The facility would also allow writing of ND applications which make use of a corresponding PC server, to exploit features found in MS-DOS but not in SINTRAN III.

6.10 SECRET MEMORY LOCATIONS

The addresses reserved for various parameters, programs and flags stored in the secret memory, are listed below.

- ND addresses are in octal. These do not include the secret memory bit (40000000₈).
- PC addresses are in hexadecimal, and are for the LOWER byte (i.e. bits 0-7) of the word.

6.10.1 PROM addresses

The PROM address range is 32kwords. Only 8k is currently fitted. A vector area from 0 to 17 points to 8 programs. The currently allocated addresses are shown in fig 6.27.

On initialisation, the area specified by the first two pointers is downloaded to DRAM, starting at address 140000. The microcode vectors IOX, IDENT and PC interrupt routines to the vector area 140000 to 140017. PROM is accessed from the PC by setting the General Offset Register (PC I/O port 165h) to 80h.

6.10.2 NV-RAM addresses

The address range for NV-RAM is 8kbytes. Currently, only 2kbytes are fitted (addresses 100000 to 103777). The currently allocated addresses are shown in fig 6.28.

NV-RAM is accessed from the PC by setting the General Offset Register to 81h. Note that bytes in NV-RAM are in the least significant half (i.e. bits 0-7) of a word.

6.10.3 DRAM addresses

The address range for DRAM is 48-64k (400000 to 177777). The currently allocated addresses are shown in fig 6.29.

Secret memory DRAM accommodates approximately 1k of mailbox, 2k of IOX simulation code, 1k of hard disk DMA area, 4k of mailbox buffer, and a 1k HDLC buffer area. It is accessed from the PC by setting the General Offset Register to 81h.

Addr	Parameter
00	start address of PROM to DRAM initialisation download (set to 00000)
01	end address +1 of PROM to DRAM initialisation download
02	1st address of IOX code (IOXSWITCH)
03	2nd address of IOX code
04	1st address of IDENT code (IDIOX)
05	2nd address of IDENT code
06	1st address of PC mailbox interrupt code (PCINT)
07	2nd address of PC mailbox interrupt code
10	1st address of spare 1
11	2nd address of spare 1
12	1st address of spare 2
13	2nd address of spare 2
14	1st address of spare 3
15	2nd address of spare 3
16	location of LEV16TIM when copied into DRAM
17	location of PF-FLAG (PFFLG) when copied into DRAM
27	IOX simulator PROM version number

Fig 6.27: Secret memory: PROM address allocations

PC ₁₆	ND ₈	Parameter
0000	100000	ERRLOG - location of error report
0002	100001	ALD - automatic load descriptor
0004	100002	AUTO-FLAG - ND may auto boot
0006	100003	HDLC-CNFG1 - HDLC configuration word 1
0008	100004	HDLC-CNFG2 - HDLC configuration word 2
000A	100005	HDLC-CNFG3 - HDLC configuration word 3
0400	101000	SIN-START - SINTRAN start cylinder
0402	101001] Cylinder / Head information
↓	↓	
0406	101003	SCRATCH - used by ACT0-WP to check if Butterfly-110 is present
0408	101004	

Fig 6.28: Secret memory: NV-RAM address allocations

PC ₁₆	ND ₈	Parameters
8000 ↓ 801E	140000 ↓ 140017	} vector area as laid out under PROM
8026 8028 802A 802C 802E	140023 140024 140025 140026 140027	ID-Q10 - IDENT queue pointer for level 10 ID-Q11 - IDENT queue pointer for level 11 ID-Q12 - IDENT queue pointer for level 12 ID-Q13 - IDENT queue pointer for level 13 - version number of installed IOX simulator
FC00 ↓ FF7E	177000 ↓ 177677	} HDLC chain load area
FF80	177700	HDLC-CNFG4- change of HDLC parameters in NV-RAM
FF90 FF92 FF94 FF96	177710 177711 177712 177713	ATTEN1 - PC to ND signal box flag (mailbox) ATTEN2 - PC to ND signal box flag (terminal 1 I/P) CONF2 - PC to ND signal box flag (terminal 1 O/P) CONF3 - PC to ND signal box flag (calendar)
FF98 FF9A FF9C FF9E	177714 177715 177716 177717	ATTEN1 - ND to PC signal box flag (mailbox) ATTEN2 - ND to PC signal box flag (terminal 1 I/P) CONF2 - ND to PC signal box flag (terminal 1 O/P) ATTEN3 - ND to PC signal box flag (calendar)
FFB0 FFB2 FFB4 FFB6	177730 177731 177732 177733	DLB-RXD - PC to ND: terminal 1 receiver buffer (st-mail) DLB-RXR - PC to ND: terminal 1 receiver ready (st-mail) DLB-TXD - ND to PC: terminal 1 transmit buffer (st-mail) DLB-TXR - ND to PC: terminal 1 transmit ready (st-mail)
FFC6 FFC8 FFCA FFCC FFCE	177743 177744 177745 177746 177747	DISP-RDY - display ready from PC (ND to update) ACTLV - active level information to the PC INDIC - PON, POF, ION, IOF information to the PC SM-PANC - panel control register for calendar SM-PANS - panel status register for calendar
FFD2 FFD4	177751 177752	STS-FLAG - self test status from ND BOOT-FLAG - ND boot sequence
FFEC FFEE	177766 177767	OPCM-FLAG - ND in OPCOM CONS-FLAG - PC is the OPCOM output
FFFC FFFE	177776 177777	SET-CONS - PC command to change OPCOM output SET-OPCM - PC command to enter/leave OPCOM

Fig 6.29: Secret memory: DRAM address allocations

CHAPTER 7 MAINTENANCE

TABLE OF CONTENTS

Section	Page
7.1	SYSTEM UNIT ASSEMBLY 7-6
7.2	PREVENTIVE MAINTENANCE 7-9
7.2.1	Fan 7-9
7.2.2	Calendar battery 7-9
7.2.3	Floppy disk drive 7-9
7.2.4	Tape streamer 7-10
7.2.5	Printer 7-11
7.2.6	Care of magnetic media 7-12
7.2.7	Routine cable inspection 7-12
7.3	SYSTEM COMPONENTS 7-13
7.3.1	Butterfly terminals 7-13
7.3.2	IBM-compatible PC-AT System Unit 7-13
7.3.3	EGA Board 7-13
7.3.4	Display Unit 7-13
7.3.5	Winchester Disk Drive 7-15
7.3.6	Floppy Disk Drive 7-15
7.3.7	Disk Controller Board 7-16
7.3.8	Keyboard 7-16
7.3.9	Mouse 7-16
7.3.10	ND-110PCX 7-16
7.3.10.1	Variants 7-16
7.3.10.2	Installation PROM 7-17
7.3.10.3	Memory upgrade 7-19
7.3.10.4	HDLC upgrade 7-19
7.3.11	Installing software on Winchester Disk 7-20
7.3.12	Extra RAM for MS-DOS 7-22
7.4	INSTALLATION TOOLS 7-23
7.4.1	PC Setup 7-23
7.4.2	Butterfly Supervisor: System Parameter Definition 7-31
7.4.3	Butterfly Supervisor: Serial Port 1 & Modem Setup 7-35
7.5	MOVING THE EQUIPMENT 7-39
7.6	CORRECTIVE MAINTENANCE 7-41
7.6.1	Basic system checks 7-41
7.6.2	Power-on self-test 7-44
7.6.2.1	Start-up sequence 7-44
7.6.2.2	Self-test operations 7-44
7.6.2.3	Self-test status flag (STS-FLAG) 7-48
7.6.2.4	Self-test error log (ERRLOG) 7-49
7.6.2.5	Self-test mailbox (st-mail) 7-50

7.6.3	Diagnostic programs	7-51
7.6.3.1	Butterfly diagnostic suite	7-51
7.6.3.2	Which diagnostic?	7-53
7.6.3.3	PC diagnostics	7-55
7.6.3.4	PC-ND interface diagnostics	7-62
7.6.3.5	ND diagnostics	7-68
7.6.4	Butterfly Supervisor maintenance facilities	7-69
7.6.4.1	Console facility	7-71
7.6.4.2	Warm Start and Cold Start	7-76
7.6.4.3	Telefix	7-80

LIST OF ILLUSTRATIONS

Figure		Page
7.1	Removing the System Unit cover	7-6
7.2	System Unit main equipment modules	7-7
7.3	Removing ND-110PCX from expansion chassis	7-8
7.4	Butterfly-110 terminals	7-14
7.5	Butterfly Diagnostic program - PC Setup Menu	7-24
7.6	BIOS Resident Setup display	7-28
7.7	System parameter definition - ALD option	7-32
7.8	System parameter definition - HDLC option	7-33
7.9	HDLC baud rate - LOW and HIGH option windows	7-34
7.10	Serial Port 1 & Modem Setup - main menu	7-35
7.11	System Manager - ND "active levels" display	7-43
7.12	Butterfly start-up sequence	7-45
7.13	ND self-test - ERRLOG codes	7-49
7.14	Butterfly Diagnostics - main menu	7-51
7.15	Diagnostics menu structure	7-52
7.16	The PC block	7-53
7.17	PC block and ND block	7-53
7.18	PC-ND interface	7-53
7.19	Butterfly-110 structure	7-54
7.20	PC diagnostics - Help window	7-55
7.21	PC diagnostics - main menu	7-55
7.22	PC diagnostic - printout: from printer test	7-57
7.23	PC diagnostic - colour 80x25 display test	7-58
7.24	PC diagnostic - 40x25 display test	7-58
7.25	PC diagnostic - graphic 320x200 display test	7-59
7.26	PC-ND interface diagnostics - Help window	7-62
7.27	PC-ND interface diagnostics - options menu	7-63
7.28	Butterfly Supervisor - main menu	7-70
7.29	Butterfly Supervisor - Console option: Help window	7-71
7.30	Butterfly Supervisor - Console function select	7-72
7.31	Butterfly Supervisor - Console Help option	7-73
7.32	SINTRAN Warm Start	7-76
7.33	SINTRAN Cold Start	7-76
7.34	Telefix system	7-80
7.35	Butterfly Supervisor - Telefix connection menu	7-80
7.36	Telefix - originate connection	7-81
7.37	Telefix - Auto-answer	7-83
7.38	Telefix - LISTEN On/Off	7-84
7.39	Telefix - Local and Remote	7-84

This Chapter presents information concerned with

- access to hardware in the System Unit
- configuring of main sub-assemblies
- installation of Butterfly-110 software
- system set-up parameters
- preventive maintenance
- corrective maintenance
- diagnostic test programs.

It does not attempt to prescribe fault-finding and repair procedures. However, it does aim to provide sufficient information to enable support personnel to understand and use the support environment that is provided for Butterfly-110.

7.1 SYSTEM UNIT ASSEMBLY

The following diagrams show how to gain access to the inside of the System Unit, and the arrangement of main equipment modules in the Unit.

Remove cover

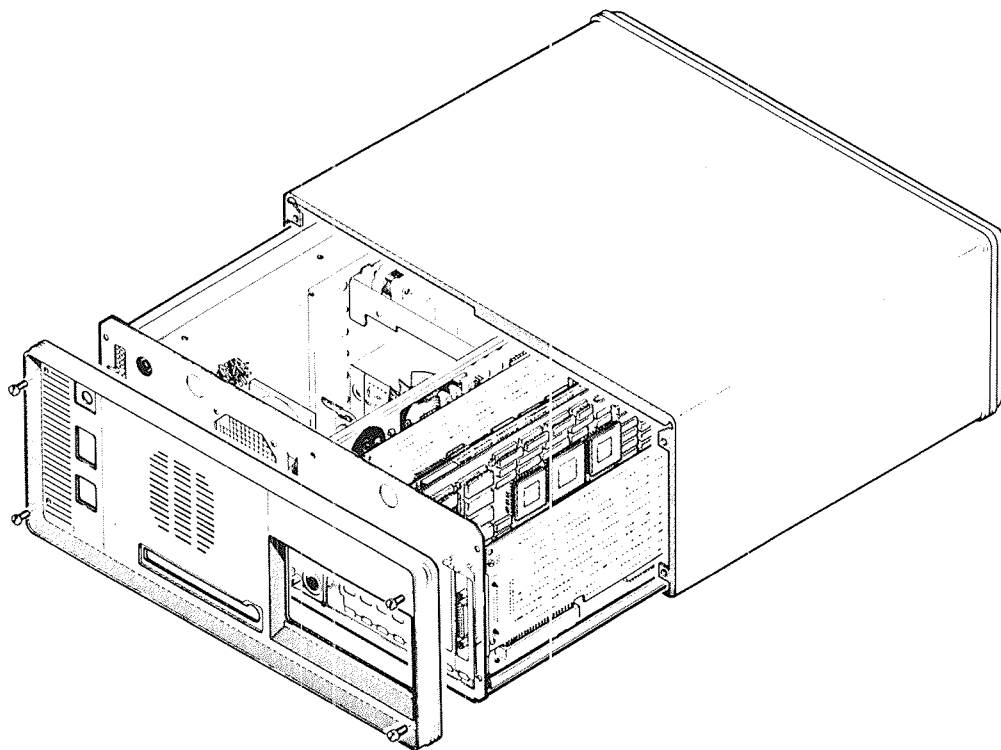


Fig 7.1: Removing the System Unit cover

Main modules

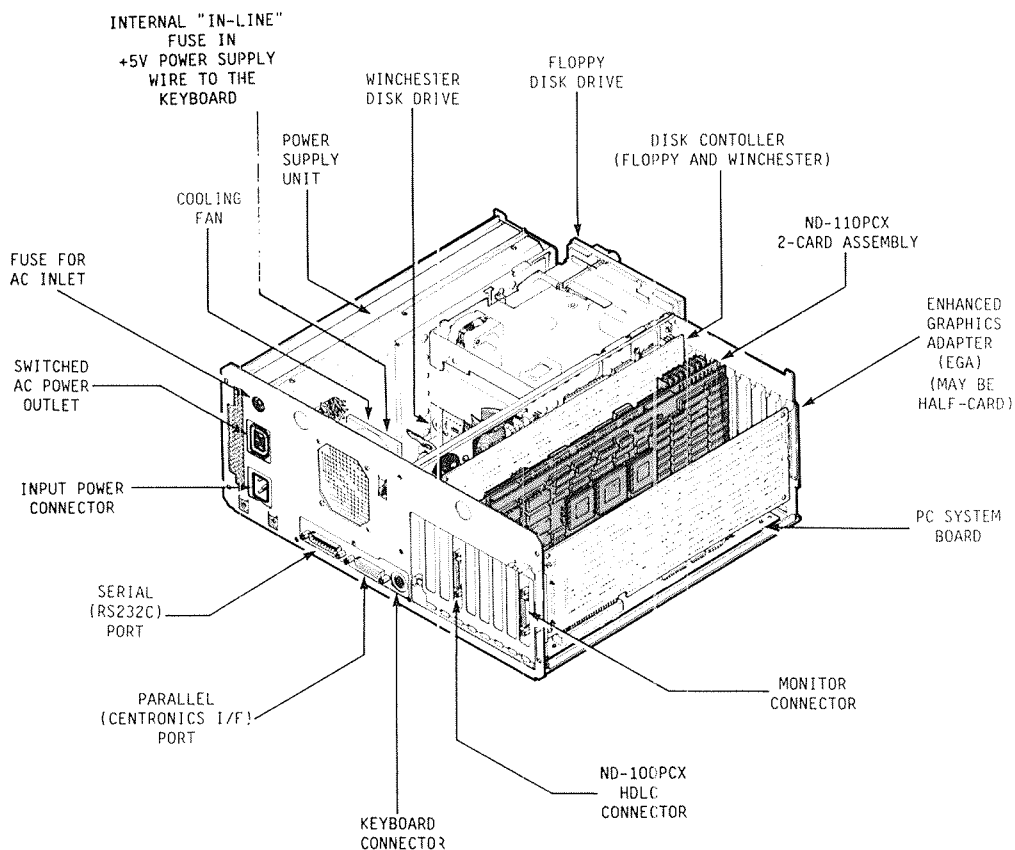


Fig 7.2: System Unit main equipment modules

ND-110PCX module

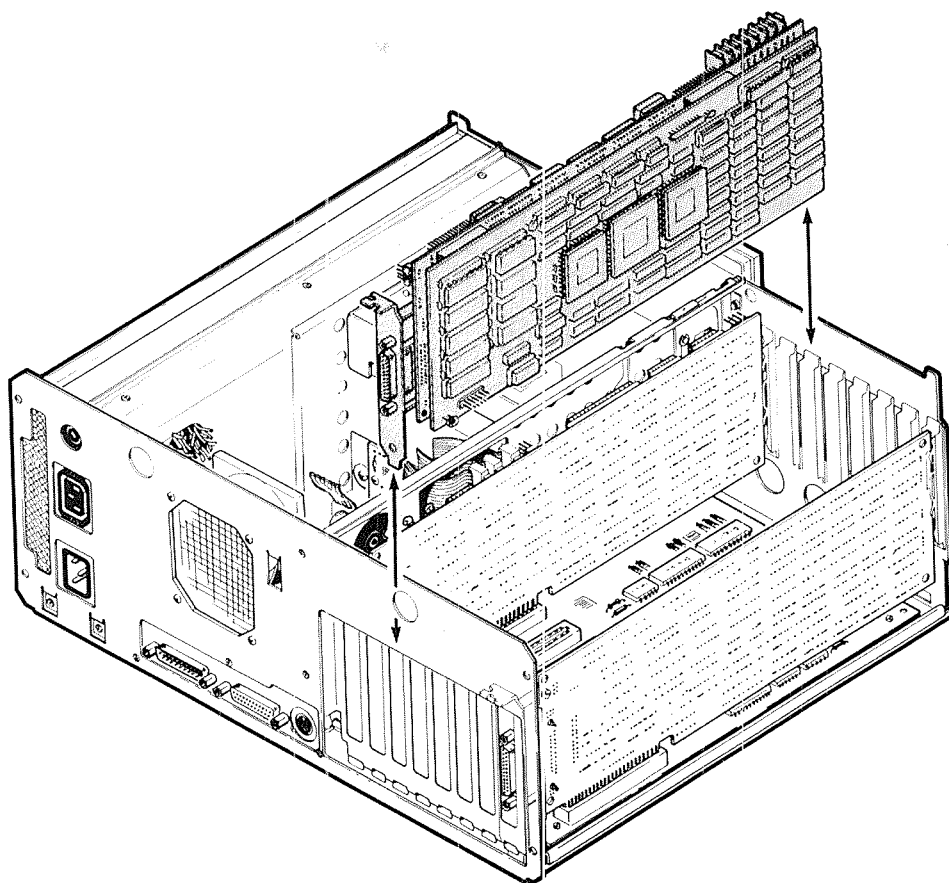


Fig 7.3: Removing ND-110PCX module from expansion chassis

7.2 PREVENTIVE MAINTENANCE

7.2.1 Fan

The System Unit incorporates a cooling fan, fitted to the rear metalwork. It expels air from the inside of the Unit, out through slots in the rear moulding.

CAUTION

Ensure that there are no obstructions to the free flow of air from the fan vent.

7.2.2 Calendar battery

The real time clock and calendar in the PC uses an MC146818 chip, which includes 50 bytes of available C-MOS RAM. The PC uses this available RAM to store its set-up parameters - see section 7.4.1.

The power to this chip is backed up by a lithium battery, so the PC calendar and setup parameters are retained for as long as the battery remains serviceable. The lifetime of this battery is typically three years.

The battery is located on the central metal support strut inside the PC enclosure, beneath the speaker. When the battery is replaced, it is necessary to run the Setup program, to restore the calendar (time and date) and other setup parameters to their required values. The procedure to do this is described in section 7.4.1.

7.2.3 Floppy disk drive

The magnetic heads in the floppy disk drive gradually accumulate a coating of dirt as they read and write to floppy disks. This coating can build up to a point where it impedes secure reading and writing, and so needs to be removed at periodic intervals. Clearly, the interval between cleaning depends on how often the floppy drive is used.

There are two methods for cleaning the heads:

- 1 Run a proprietary Head Cleaning Disk in the drive
- 2 Use a proprietary cleaning kit to wipe the read/write heads clean.

Proprietary
cleaning disk

Use a proprietary Head Cleaning Disk.

NOTE

Note that the Butterfly floppy drive is a 5.25 inch double-sided double-density unit. Check that the proprietary head cleaning disk is suitable for this unit.

To run the cleaning disk, bring up the SINTRAN prompt "@". Then proceed as follows:

```
@CREATE DIR ↵
  DIRECTORY NAME: ↵
  DEVICE NAME: FLOPPY-DRIVE ↵
```

An error message may appear on the screen, but this is of no importance in this situation. The cleaning is completed when the SINTRAN "@" prompt reappears.

Cleaning kit

Use a proprietary cleaning kit comprising isopropyl alcohol and lint-free cloth. Recognised standard cleaning kits are 3M's CK-90, and Tandberg's 961536. Wrap the lint-free cloth round a wooden spatula, soak it with drops of the isopropyl alcohol, open the disk drive door, and gently stroke this cleaning agent between the two disk drive heads.

7.2.4 Tape streamer

As in the case of the floppy disk drive, the magnetic heads in the tape streamer drive gradually accumulate a coating of dirt as they read and write to streamer tape cartridges. This coating can build up to a point where it impedes secure reading and writing, and so needs to be removed at periodic intervals. Clearly, the interval between cleaning depends on how often the tape streamer is used. In normal circumstances the heads should be cleaned after every 8 hours of use, but in adverse conditions or where new tape cartridges are frequently used, the heads should be cleaned more often.

There are two methods of cleaning the heads:

- 1 run a proprietary Head Cleaning Cartridge in the drive
- 2 use a proprietary cleaning kit to wipe the read/write heads clean.

Clean cartridge Use a proprietary Heads Cleaning Cartridge.

Cleaning kit Use a proprietary cleaning kit, comprising isopropyl alcohol and lint-free cloth. Recognised standard cleaning kits are 3M's CK-90, and Tandberg's 961536. Wrap the lint-free cloth round a wooden spatula, soak it with drops of the isopropyl alcohol, open the tape streamer drive door, and gently stroke this cleaning agent between the two streamer drive heads.

7.2.5 Printer

The manual supplied with the printer explains how its various controls (e.g. print pressure, paper guides) should be set, and how to load paper, change print ribbons, and clean the platen and print head.

It may also recommend periodic removal of dust and paper debris from inside the printer, using a small vacuum cleaner.

WARNING

In all such activities, ensure that you DISCONNECT the printer from its mains supply BEFORE going inside.

Refer to the supplier's manual for guidance on opening covers to gain better access to the inside of the printer. The supplier's manual will also list any other routine maintenance required.

7.2.6 Care of magnetic media

Ensure that users keep their floppy disks and streamer tape cartridges in their protective envelopes when not in use, and store them under safe conditions. The Butterfly Supervisor Guide Chapter 5 details specific points that should be observed.

Floppy disks and tape cartridges are durable items and can withstand surprising amounts of misuse. However, if read or write problems are encountered, first check the media visually for any signs of damage, and then check that the heads are clean.

7.2.7 Routine cable inspection

Faults with cable connections are a common cause of problems. Therefore, whenever the opportunity arises, check that cables are not mechanically stressed, are securely connected, and have sustained no visible damage.

Any signs of damage (cuts or abrasions in insulation, damaged strain-relief fittings at connectors, etc.) should be rectified without delay.

7.3 SYSTEM COMPONENTS

This section is concerned with identifying the system components in Butterfly-110, pointing out those features that require consideration when installing or replacing a component.

7.3.1 Butterfly terminals

Fig 7.4 identifies terminal device components in Butterfly-110, within the context of its SINTRAN environment. It is presented a self-sufficient reference information for those who require it. See also Chapter 6, fig 6.2.

7.3.2 IBM-compatible PC-AT System Unit

Butterfly uses the Ericsson WS-286 (IBM-compatible PC-AT from Ericsson Information Systems - EIS) as the basic PC part of its System Unit. The PC Technical Reference Manual (ND-06.028) provides information on this unit, including the main board.

7.3.3 EGA Board

ND part no 530075 The Paradise Autoswitch EGA-1A card from Paradise Systems is the preferred source. Alternative sources are the Sigma Designs EGA card and the Taxan Paradise EGA card. The relevant supplier's manual gives instructions on the installation and set-up of this sub-assembly.

7.3.4 Display Unit

This may be a monochrome or colour monitor.

ND part no 527069 (colour) The current colour monitor is either the Hitachi CM1255SE unit (which has a 12-inch screen), or the Wyse WY-640 (which has a 14-inch screen). The relevant supplier's manual gives instructions on the installation and set-up of the monitor.

At the time of writing, a monochrome monitor has not been selected.

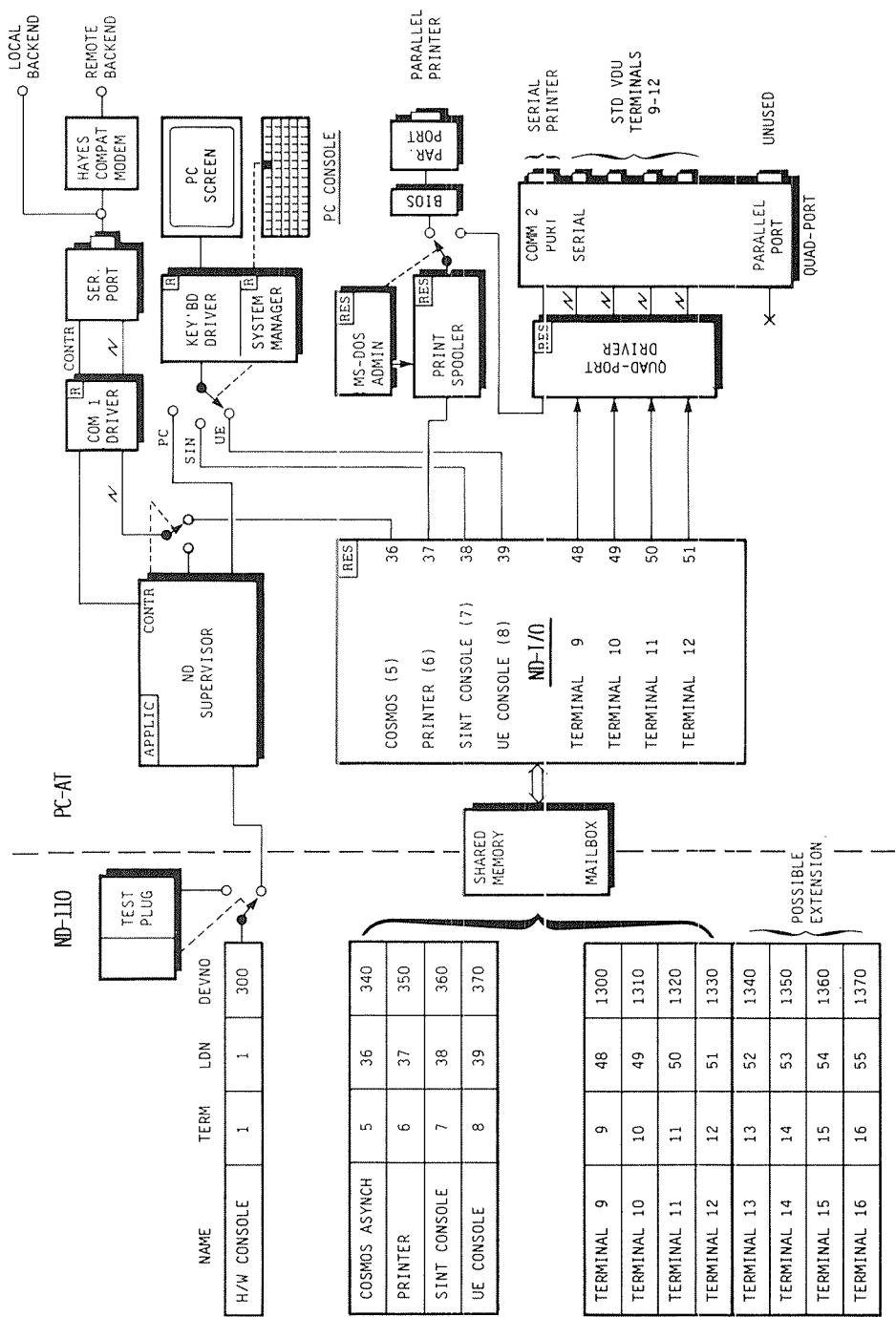


Fig 7.4: Butterfly Terminals

7.3.5 Winchester Disk Drive

ND part no
525055

The nominal storage capacity is 51Mbytes unformatted, giving approximately 41Mbytes formatted. Partitioning allocates a nominal 5Mbytes to MS-DOS, leaving 36Mbytes to SINTRAN. This disk is specified in the system as MS-DOS disk C (i.e. >C).

The currently approved source for this 5.25-inch half-height unit is NEC (type D5146H-9), with CDC (Wren II 94205-51) as an alternative source. See the relevant supplier's manual for installation and set-up information.

The Setup program requires the disk type to be identified, in the form of an "E" number:

e.g. E9 for NEC; E7 for CDC.

This is described further in section 7.4.1 (Setup parameters - hard disk C and D). If other disk types are approved, they will require their own disk-type (E) identifier.

See section 7.3.11 for a summary description of the procedure to prepare and install software on the Winchester disk.

7.3.6 Floppy Disk Drive

ND part no
529023

Approved sources are the TEAC FD55 and NEC FD1155C. See the relevant supplier's manual for installation and set-up information on this sub-assembly. The floppy disk drive is a 5.25-inch half-height double-sided double-density PC-compatible unit, known as a 1.2Mbyte/360kbyte drive. This disk is specified in the system as MS-DOS disk A (i.e. >A). Two floppy disk types are used on Butterfly:

- high-density (HD), 96 tracks-per-inch, 1.67 Mbytes unformatted, MFM recording mode
- double-density (DD), 48 tracks-per-inch, 500 kbytes unformatted, MFM recording mode.

The Butterfly floppy disk will read and write correctly to HD disks recorded with 1.2AT and 1.2ND formats. It will also read and write to DD disks recorded with 360PC format. When writing in 360PC format, the recorded track width is narrower than on normal 360PC devices. This may cause marginality problems when reading on a different drive, and should therefore be avoided.

7.3.7 Disk Controller Board

ND part no The Western Digital WD1002-WA2 Disk Controller board provides control for both the Winchester disk and the floppy disk. Refer to the supplier's manual for information on this board.

525056

7.3.8 Keyboard

ND part no Butterfly's PC-NOTIS keyboard is type CEC128-ND-1000 supplied by Pendar/Clare Electronique, France, to ND's specification. This specification is given in Appendix F of this manual.

530079

7.3.9 Mouse

ND part no The mouse is an industry-standard unit. It is currently the type M3/100 mouse, from Penny & Giles Potentiometers (England).

530080

7.3.10 ND-110PCX

7.3.10.1 Variants

CPU board 3401

Part No.	Schematic	Description
324051	3401	ND-110PCX CPU & Memory Management

Local Bus board 3402

Part No.	Schematic	Description
324052	3402	ND-110PCX Tmstn Local Bus: 2MB & HDLC
324143	3402	ND-110PCX Tmstn Local Bus: 2MB
324144	3402	ND-110PCX Tmstn Local Bus: 1MB & HDLC
324145	3402	ND-110PCX Tmstn Local Bus: 1MB

7.3.10.2 Installation PROM

Descriptor An "installation descriptor", comprising a piggy-back Installation PROM board (3404) which plugs into connector J2 on the 3401 CPU board, defines the CPU number of the machine, its authorised software configuration, and its authorised hardware configuration of hard disk(s), HDLC and any extra terminals (on a QuadPort Communications Controller expansion card). This scheme is outlined in chapter 4 section 4.3.4.11. For convenience, the six PROMs on the Installation PROM board define the configuration of the following features:

- PROM A: CPU number
- PROM B: software configuration
- PROM C: hard disk C configuration
- PROM D: hard disk D configuration
- PROM E: HDLC configuration
- PROM F: terminal configuration

Each of the PROMs on this board contains a Copyright statement, rendering it illegal for any unauthorised copying of them.

Upgrading If an upgrade to Butterfly-110 affects any aspect of the configuration information held in Installation PROM, then it will include the requirement to update this Installation PROM, either by replacing the Installation PROM board, or by inserting/replacing one or more plug-in PROM chips in the header sockets on the board.

The CPU number Only the CPU number PROM is necessary for the minimum configuration of Butterfly-110. The other PROMs are added when upgrades are installed.

Self-test checks	During the start-up self-test sequence, the microcode checks if the PROMs contain the Copyright statement. If not, it issues an error code to the PC startup program, and the microcode enters a loop preventing any further progress in starting-up the ND. Note that at this stage, the CPU number is not checked.
SINTRAN start-up checks	<p>The CPU number is checked during the SINTRAN start-up phase. The start-up routine compares the CPU number in PROM with that which is configured in the software. It displays the hardware installation number that it finds, and if it finds a difference, it resets the System/CPU number to 0 (which means "undefined"), and displays an error message to this effect. Note that at this stage, SINTRAN only resets the CPU number on segments, not on files, so SINTRAN symbol tables are not therefore altered.</p> <p>New systems have the SINTRAN CPU number = 0, so if an Installation PROM board gives (correctly) a different CPU number from SINTRAN CPU number 0, the software outputs an error message advising of this situation.</p>
DO Supervise-Sys	To update the SINTRAN CPU number to become the same as the Installation PROM CPU number, the DO-script "DO Supervise-System, Update CPU number" is used. This copies the Installation PROM CPU number to the necessary (6) places in the software.
DO System-Info	<p>This DO-script displays the CPU numbers that are in the software and the Installation PROM, and so provides a convenient facility to check what these values are set to.</p> <p>It also displays the values of all the other configuration parameters.</p>
HDLC checks	During the start-up sequence, assuming that self-test finds the Copyright statement is present and correct, it checks if HDLC is configured as present (coded in Installation PROM E), and if so, it sets an "HDLC present" flag that is checked before HDLC IOX instructions can be executed.
IOX simulators check	The IOX-simulators-check reads the Installation PROM to determine what terminals configuration (PROM F), and hard disk configurations (PROM C for disk C; PROM D for additional disk D), are authorised for this machine. The corresponding IOX instructions are then enabled accordingly.

7.3.10.3 Memory upgrade

Currently, the 1Mbyte RAM on the ND-110PCX comprises four 30-pin device assemblies, which are soldered into the Local Bus board. The additional 1Mbytes of RAM (a further four 30-pin RAM device assemblies) fit into adjacent sockets on the Local Bus board.

The memory upgrade kit (ND-110117) comprises four 256kx9 SIP memory devices (ND-516034) and one PAL (ND-324051-41800). On the Local Bus board (schematic 3402), the SIP memory devices plug in to sockets, and the PAL replaces that in U186 (issue C board). Note that to ensure proper heat dissipation the SIP components must be mounted vertically $\pm 10\%$.

This upgrade converts a 1MB Local Bus board (Teamstation or OWS-110) into its 2MB counterpart, i.e.

Part No 324145 into Part No 324143
.. .. 324144 324052.

7.3.10.4 HDLC upgrade

This upgrade involves fitting two IC chips into the Local Bus board:

- synchronous communication controller (Z08530-08 PSC0652)
- DMA controller (AM9516-6).

It also involves upgrading the Installation PROM (see section 7.3.10.2).

This upgrade converts

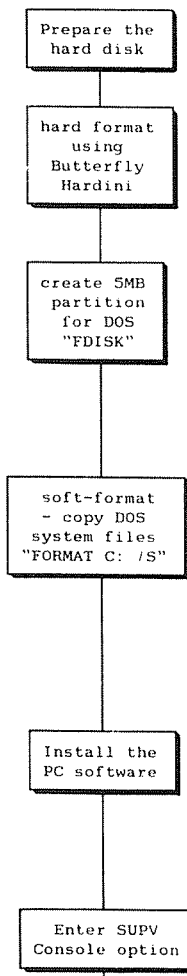
Part No 324143 into Part No 324052
.. .. 324145 324144

7.3.11 Installing software on Winchester Disk

The following flowchart identifies the sequence of main operations involved in installing Butterfly-110 software on a new or corrupted Winchester disk. Note that this refers to the procedure at the time of writing: this procedure is given in the PD (program description) sheet Reg No 211092A, "Butterfly-110 ND System Software".

Format Hard Disk:

PD sheet Reg No 230011A "Butterfly Hard Disk Formatter - Hardini" details the procedure to format the hard disk.

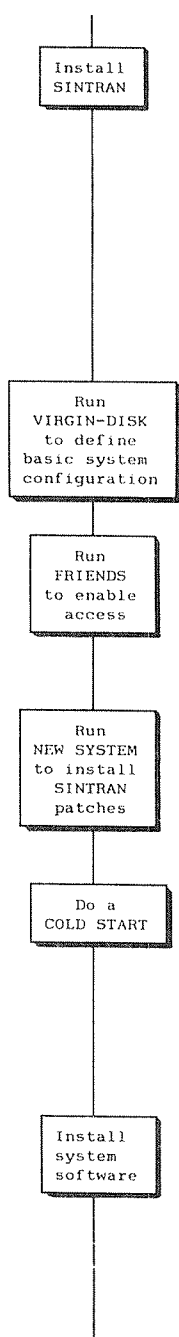


- Preparing the disk involves specifying the type of hard disk that is fitted: E7 for CDC 94205-51; E9 for NEC D5146H. This is set using the Setup Utility.
- Hard Formatting uses the Ericsson "Hardini-1" program. Firstly, it "hard" formats the hard disk. Then it requests "bad-track" cylinder/head addresses (those with known media defects - these are listed on the top of the hard disk unit): type them in one-by-one. It then creates a 33Mbyte DOS partition, and finally reboots from the Hardini floppy disk to enable the "Hardini" newly formatted DOS partition to be seen by DOS). program
- Partitioning the hard disk for Butterfly uses the FDISK program on the Hardini floppy disk. At the DOS prompt, type "FDISK*". Then select option 3 to delete the 33Mbyte partition created during hard disk formatting, and continue by pressing ESC then selecting option 1 to create a new DOS partition. Give the partition size as 120 for a CDC disk (E7) or 80 for a NEC disk (E9). This partitions the disk with 5Mbytes for DOS. Set the start cylinder number to 0. Then make the newly-created DOS partition active by selecting the "change active partition" (option 2). Finally enter the partition number as 1, and exit by pressing ESC twice. Then reset the machine (by pressing CTRL+ALT+DEL): this causes it to reboot.
- Soft Formatting involves copying the DOS system files (with the Butterfly Hardini disk in floppy drive A, then from the DOS >A prompt, type "FORMAT C: /S*", followed by "Y" at the prompt to continue. This "soft-formats" the DOS 5Mbyte partition, with the DOS System files in place.

Install PC System Software

PD sheet Reg No 230001A "Butterfly-110 PC System Software" details the procedure to format the hard disk.

- Insert floppy disk 230001A, reset the machine (by pressing CTRL+ALT+DEL), then follow the prompt instructions to install files from the two DOS 3.10 floppy disks (ND-230004) onto the DOS partition of the hard disk. This process creates all the necessary DOS subdirectories, copies all the necessary DOS programs, creates a Butterfly directory, and then runs the DISKPREP maintenance program, which prepares a file containing the list of any re-allocated tracks that have been marked on the hard disk.
- Press CTRL+ALT+DEL. This resets the machine and reboots from drive C.
- Select MS-DOS from the System Manager, and at the DOS prompt (>C\>) enter the System Supervisor program by typing "SUPV*" followed by "*" for No Password, then select the Console option.



Load SINTRAN from floppy disk

- Insert floppy disk 211087J "SINTRAN III for Butterfly-110", and type 1560&←. This invokes MACM, which has been configured for the type of hard disk in Butterfly-110. When the message "type any MACM command" appears, type 10,C\$←, which is the MACM command to copy SINTRAN from floppy disk to the correct Save areas of hard disk. It also generates the Image copy on hard disk, and loads SINTRAN into ND-memory.
- When the message "**** 000000 DIAGNOSTICS ****" appears, type "221←". This starts the program running from memory location 22.
- When the message "PAGES FOR SWAPPING (OCT): ..." appears, press ESC, and at the Log-in display, type "←←" (log in as No User and No Password). When the message "NO MAIN DIRECTORY" appears, remove the floppy disk and proceed to define the Basic SINTRAN System.

Define Basic SINTRAN System

- Insert floppy disk "Butterfly-110 ND System Software", and at the SINTRAN prompt (@) type "ENTER-DIRECTORY,,FLOPPY-DISC-1←". Then log out, then back in again as user "SYSTEM" with no password. Then at the @ prompt, type "MODE VIRGIN-DISC:MODE,,←". This creates and enters a directory on the hard disk, creates users and files, gives user space, sets the hard disk as the main directory and finally logs itself out.
- Press ESC and log back in as "SYSTEM" with no password. Then, at the @ prompt, type "MODE (211092A:SYSTEM)FRIENDS:MODE,,←". This creates friends, access rights and passwords (System password is SYSTEM by default), and creates dummy files for use by Load-mode and Hent-mode. When this is completed, log out, then back in as user "SYSTEM" with no password. Then, at the @ prompt, type "RELEASE-DIRECTORY,2←", and remove the floppy disk.

Install SINTRAN patches

- Insert floppy disk "SINTRAN III for Butterfly-110", and at the SINTRAN prompt(@) type "ENTER-DIRECTORY,,FLOPPY-DISC-1←". Then, to start the patching process, at the @ prompt type "(ND:SYSTEM)NEW-SYSTEM←". After an initialisation phase, this asks for the floppy disk 211088J "SINTRAN III Patch File for Butterfly-110": insert this floppy disk, and type "Y" for patching to proceed (this takes 30-60 minutes to complete).
- When patching is completed, at the @ prompt type "COLD-START←" to execute a cold start (cold start is described in section 7.6.4.2). When the message "PAGES FOR SWAPPING (OCT): ..." appears, press ESC, then log back in as No User with No Password (by typing "←←"). Then, when the message "NO MAIN DIRECTORY" appears, at the @ prompt type "ENTER-DIRECTORY,,D-36MB-C←", then log out and remove the floppy disk.

At this point, the empty hard disk has been brought up to the level where the Standard Installation Procedure starts.

Install ND System Software

- Press ESC, and log in as user "SYSTEM" with password "SYSTEM". Insert floppy disk 211092A "Butterfly-110 ND System Software", and at the @ prompt type ENTER-DIRECTORY,,FLOPPY-DISC-1.0←. Then at the next @ prompt, type "MODE (211092A:FLOPPY-USER)BFLY-INSTALL:MODE,,←" to execute the Butterfly-110 installation. When this completes, type "RELEASE-DIRECTORY,2←" and remove the floppy disk, then type "MODE HENT-MODE:MODE,,←" to execute the Hent modefile.

When this completes, Butterfly-110 is fully operational and ready for installation of subsystems.

Install
subsystems

Install Subsystems

A natural order of loading subsystems is

210337H Backup System
210373J X-Message (Inter System)
210374C COSMOS Basic Module
211129A Async COSMOS
211090C User Environment for Butterfly-110
211091A DO System for Butterfly-110
210079M NOTIS-WP for ND-100
211089B File-Manager B-version
211093A NOTIS-DS/ID for Butterfly-110
210375C Telefix Files for User Sites

Further information on the SINTRAN installation procedures is given in the SINTRAN III System Supervisor Guide (ND-30.003).

7.3.12 Extra RAM for MS-DOS

The facility to donate 128kbytes of the ND's RAM as extra memory to the PC (extending it from 512k to 640k) is described in Chapter 2 section 2.2.8, and illustrated in Chapter 4 fig 4.18.

This facility is selected via a "MS-DOS memory" link on the Local Bus board. It is shown on schematic 3402, sheet 1, drawing area C5. This link is pins 1 and 2 of header J107.

- link fitted : disables the MS-DOS extra RAM
- PC memory is 512kbytes.
- link omitted : enables the MS-DOS extra RAM
- PC memory is 640kbytes.

7.4 INSTALLATION TOOLS

7.4.1 PC Setup

- Requirements** Various operating parameters must be set up correctly in the PC. These are stored in battery-backed memory (C-MOS RAM), in the System Unit.
- After a system repair or modification, it may be that some of these parameters will need to be re-entered or changed. The Setup program provides for this.
- The Setup program** Ericsson Information Systems provide a Setup program which is a part of their "maintenance program". It provides facility to enter configuration parameters into the PC.
- Loading program** In Butterfly-11C, the Setup program is available on the hard disk, and may be called from the MS-DOS prompt, by typing SETUP.
- It is also available on the Butterfly Diagnostic Program floppy disk, under the "PC Diagnostics" option. This ensures that the PC can be configured by the Setup program loaded from the floppy disk, even if the hard disk is unavailable. For information on loading this program from floppy disk, see section 7.6.3.1.
- Further, the PC's BIOS has a built-in Resident Setup utility which emulates the functions of the Setup program. If it proves impossible to load and run Setup from hard disk or from floppy disk, this Resident Setup facility provides a third resort to achieve configuring of the PC.

-
- Diskette drive B 360kB, 1.2Mb, No drive

Drive B is not fitted in Butterfly-110.

- Hard Disk C and D Specify type
E7 for CDC or E9 for NEC

The "E" number correlates software harddisk addresses with physical (cylinder/head) addresses. Two tables exist, one for EIS-types and the other for IBM-types. Butterfly-110 is supplied with type E7 (CDC) or type E9 (NEC) fixed disk drives. If and when other hard disk types are approved they will be assigned an appropriate E-type identifier.

- Base Memory Size 512kb, 640kb

The standard PC memory is 512kb. Part of the ND-110PCX memory is taken to expand this to 640kbyte.

- Exp Memory Size Specify size (0kb)

There are no memory expansion boards fitted. If fitted, the start and stop address values of that memory have to be set up on the expansion memory.

- Keyboard type Standard IBM-PC No Keyboard

Butterfly-110 uses a version of the standard NOTIS keyboard. For PC operations, it emulates the standard EIS keyboard. Other settings are for IBM-PC keyboard (invalid for Butterfly-110) and for No Keyboard (selected if the PC is used as a network-server)

- Start-up Num Lock On, Off

If this is On, the indicator lamp and the function of the Num lock key is turned on after power-up.

- Video Mode Color 40 column
 Color 80 column
 Monochrome
 No video

Color 80 column is the standard setting for both colour and monochrome monitors. Color 40 column or Monochrome should only be set for applications software that uses the mode. No video applies if no monitor is used.

- Math Coprocessor Available, Not available

If you wish to install a math coprocessor on the system, set this parameter to "available".

-
- Serial Port address 03f8h, 02f8h, Disabled

If split speed is to be used with the serial port on the system board, this address parameter must be the same as the address of the port, 03f8h or 02f8h. The address is 03F8H, unless it has been changed by moving a strap on the system board. When split speed is not used, this parameter is ignored.

- Enhancement Switch On, Off

The enhancement switch should normally be set to Off.

If set to On:

- in PC self-test, the program automatically sets the available memory size to the correct value. It is therefore possible that the PC may handle certain self-test errors more easily if the enhancement switch is set to On.
- in normal operation, the PC system performance is increased at the expense of minor reductions in compatibility with the IBM-PC. Therefore, if the enhancement switch has been set to On and problems then arise with running a particular PC software application, setting it to Off should remove the problems.

- Boot Select Standard
Drive B enabled
Diskette enabled

Standard: If a floppy disk is inserted in drive A, then after the power-on test has been executed, or the system has been reset, the operating system is booted from the floppy disk, provided that this is a bootable disk. Otherwise booting occurs from the hard disk C.

Drive B enabled: If you have a second floppy drive, the disk in this drive can be used to boot the system, provided that no boot disk is inserted in drive A. Standard Butterfly-110 does not have Drive B fitted.

Diskette disabled: This means that the operating system cannot be booted from floppy disk, only from hard disk. This allows a non-system disk to be permanently inserted in drive A.

- Sound Level Off, Low, Normal, High

This option allows you to adjust the sound level of the system units loudspeaker.

- Password

This allows you to specify a PC password or change an existing PC password. It is totally different to the ND user-SYSTEM password. A PC password would be used where you wish to restrict total machine access to only authorised people. It can be any string of up to 10 characters.

If a password is set up, it must be entered correctly by the user each time the system is powered On, otherwise the system cannot be used.

NOTE

If a password is set up, write it down and keep it in a confidential place.

If a password has been set up but forgotten, it can be cleared by removing power (0/+5V) from the PC's battery-backed C-MOS RAM. To do this, remove the cover of the System Unit, and disconnect the plug to J6 on the PC's System Board. Wait a few seconds before reconnecting the plug to J6, to ensure that the power supply to the C-MOS memory falls sufficiently for all the stored data to be lost. Then, enter the PC Setup program (see start of this section) and re-enter the Setup parameters.

- Store and Exit

When the Setup parameters have been specified, to enter them into the PC, select the "Store and exit" option and press ↵. After storing the parameters, this then exits from the program.

Alternatively, to quit the Setup program without storing, press the ESC key.

Resident Setup This is the BIOS Setup utility which emulates the functions in the Setup program. It is available for use if difficulties arise in loading the Setup program both from hard disk and from floppy disk.

To enter the BIOS-Resident-Setup, press the key combination

CTRL + ALT + Numeric keypad 0

This brings up a display showing the contents of the PC's CMOS memory for the Real Time Clock, with values in hexadecimal format (fig 7.6). Those bytes whose value may be changed carry an identifying label in this display.

Resident setup utility															
Setup CMOS RAM															
sec	min	hour	WD	Day	Mon	Year								Error	
xx 00	xx 00	xx 00	xx	xx	xx	xx	26	02	50	80	00	00			
FD	HD	Equ	Bmemory	Xmemory											
20 00	70 00	21	80 02	00 00	00	00	00	00	00	00	00	00	00	00	00
F1	KBD	HD2	Res												
00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	01	13	
Xmemory Cen Information															
00 00	19 80	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00	00 00
Press Esc when ready															

xx = current value

Fig 7.6: BIOS Resident Setup display

Values that may be changed are as follows:

- sec seconds part of time in BCD format
- min minutes part of time in BCD format
- hour hour part of time in BCD format
- WD current weekday. This is not used by DOS, or BIOS, or SINTRAN.
- Day current day in BCD format.
- Mon current month in BCD format.

-
- Year current year in BCD format.
 - Error error code from power-on test. The two error code bytes are described below. In some situations, they must both be set to 00 in order to enable the PC to boot.
 - FD floppy disk type.
Most significant 4 bits are Drive A:
 type 1 = 360 kbyte
 type 2 = 1.2 Mbyte.
 - HD hard disk type.
Most significant 4 bits are usually Drive C:
For the Ericsson table, existing values are
 type 7 = CDC
 type 9 = NEC.
If the IBM table is used, then HD2 (see below) is set to the same value.
 - Equ defines the basic equipment in the system for power-on diagnostics:
 21 = 1 floppy drive
 EGA display driver in 80x25 mode
 no math coprocessor.
 - Bmemory word, least significant byte first, giving the number of kbytes of base (i.e. below 640k) memory
 00 02 = 200H, =512 decimal
 80 02 = 280H, =640 decimal.
 - Xmemory word, least significant byte first, giving the number of kbytes of extended (i.e. above 1M) memory.
 - F1 flag, contains the following bits:
 - bit 6,7 serial port on system board (for split speed):
 00 for disabled
 01 for I/O address 03F8
 10 for I/O address 02F8.
 - bit 5 enhancement switch. If set, this might give better performance, but might also lessen IBM-compatibility.
 - bit 4,3 extended boot.
 00 for standard boot
 01 for Drive B boot enable
 10 for disk boot disable.

-
- bit 2 startup with Keyboard "Num Lock" light on.
 - bit 1,0 startup audible beep sound level:
 - 00 for normal
 - 01 for high
 - 10 for off
 - 11 for low.
 - KBD keyboard type:
 - 00 for standard (PC-AT equivalent)
 - 10 for IBM PC
 - FF for none.
 - HD2 hard disk table 2. 00 if the Ericsson table is used.
 - Xmemory memory above 1Mbyte found by power-on test.
 - Cen century part of the year date.
 - Information 80 if 64Kbyte memory is available to DOS. This represents the extra 128kbytes (see Bmemory above). Bit 6 may be used by the Setup utility to output a first user message after initial setup.

Hard RESET

Hard reset of the PC is effected by pressing ALT+CTRL+xx, where xx is a free code chosen by the user. If the PC is started up with standard ND software, the keyboard driver will choose that code to be 6Ah, which corresponds to the EXIT key. Thus, if ND's keyboard driver is active, the hard reset is obtained by pressing ALT+CTRL+EXIT.

If another keyboard driver is used (e.g. EIS') the code must be set in the PC's CMOS-RAM location 23h (labelled RES in the CMOS-RAM Setup utility - see fig 7.6). So, to ensure the hard reset is the same in the system, enter Resident Setup (ALT+CTRL+numeric 0) and set the code RES to the value 6Ah.

A future issue of the Setup program will include facility to set this code.

Error codes

The two error code bytes give status information on possible error conditions:

Diagnostic status byte (0Eh):

- bit 7 1 = RTC chip has lost power.
- bit 6 1 = configuration record checksum is bad.
- bit 5 1 = equipment byte configuration is invalid.
- bit 4 1 = memory size mismatch.
- bit 3 1 = hard disk failed initialisation (& cannot boot)
- bit 2 1 = time setting is invalid.
- bit 1,0 reserved.

Shutdown Status Byte (0Fh):

When a "shutdown" command is encountered, the Shutdown Status byte is used to determine where control must be returned to. The following values apply:

- | | |
|---|---------------------------------------|
| 0 | soft reset or unexpected shutdown |
| 1 | shutdown after memory size |
| 2 | shutdown after memory test |
| 3 | shutdown with memory error |
| 4 | shutdown with boot loader request |
| 5 | JMP DWORD request (with int init) |
| 6 | protected mode test 7 passed |
| 7 | protected mode test 7 failed |
| 8 | protected mode test 1 failed |
| 9 | block move shutdown request |
| A | JMP DWORD request (without int init). |

7.4.2 Butterfly Supervisor: System Parameter Definition

This is one of the options in the Butterfly Supervisor program. It displays the current System Parameter settings, and allows you to change them. These System Parameters are associated with loading of SINTRAN and with the ND's HDLC communications facilities.

For each parameter, a help window is available. On selecting a parameter, an options window displays the available settings for that parameter, and highlights the current selection. To change the setting, move to the required option (as described in the bottom area of the display) and press RETURN. Alternatively, to leave the current setting unaltered, press EXIT.

Program load This is described in section 7.6.4.

Automatic Load The ALD option specifies what device ND programs will be
Descriptor (ALD) loaded from, and under what conditions they are loaded.

- The available devices are floppy disk, hard disk, or nothing (i.e. enter Stop Mode).
- The condition available with each device is restart program if it is already resident in ND Memory, otherwise load it (i.e. boot) from that device. This covers the case where, if your system has battery-backed ND Memory, on power-fail the contents of memory are retained. Then when power is restored, microprogram checks that standby power was sufficient, and if so restarts the program from memory address 20. If standby power was not sufficient,

the ALD flag then directs reloading of program from the specified device.

These choices are presented in an options window on the display (fig 7.7).

Butterfly Supervisor Program		Version
Main Menu -> System Parameter Menu		
<div>1 Automatic Load Descriptor:Disk 2 Automatic Boot :Enabled 3 HDLC Connection Standard :V24 4 HDLC baud rate (low) :4800 5 HDLC baud rate (high) 6 Exit</div>		<div>Options 1 Floppy Disk 2 Hard Disk 3 Stop 4 SINTRAN, else Floppy Disk 5 SINTRAN, else Hard Disk 6 SINTRAN, else Stop</div>
Select option using ↑, ↓, and RETURN or by number		EXIT to terminate program HELP for help

Fig 7.7: System parameter definition - ALD option

ALD To change the ALD setting, move to the required option in the options window, then press ↵.

Automatic Boot This "flag" specifies whether the system is to proceed automatically to load ND program, after a power-up, ND master clear, or Warm or Cold Start. The alternative to loading SINTRAN is that the ND goes into Stop Mode, with its MOPC program running (see the Console's STOP option, section 7.6.4.1). In Stop Mode, the Console's LOAD function can be used to load a program under control of the MOPC. The options for this parameter are therefore whether or not to enable Auto-boot,

- i.e. enabled
- disabled.

To change the current setting, move to the required option in the options window, then press RETURN.

HDLC Conn Std

This parameter specifies the physical connection standard required for HDLC communications. HDLC is not available in the first release of Butterfly-110.

The options window currently offers three choices (fig 7.8).

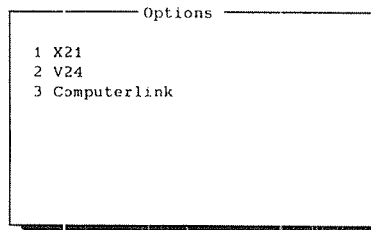


Fig 7.8: System parameter definition - HDLC option

To change the current setting, move to the required option in the options window, then press RETURN. Alternatively, press EXIT to leave the current setting unaltered and return to the System Parameter Menu.

HDLC baud rate (low & high)

This parameter specifies the HDLC line transmission speed when the Computerlink connection is selected as the HDLC communication standard.

The baud rate may be set to any one of thirteen values. To make it easy to see these values clearly in the options box on the display, they are divided into two options displays - low and high - in the System Parameter Definition menu.

- In the LOW range, six speeds from 2400 to 16000 baud are offered.
- In the HIGH range, seven speeds from 19200 to 153600 are offered.

The available speeds are presented in respective low and high option windows. Of course, only one HDLC baud rate may be set at any one time - the current setting is displayed next to its corresponding low or high option.

Fig 7.9 shows the options windows for the LOW and the HIGH options.

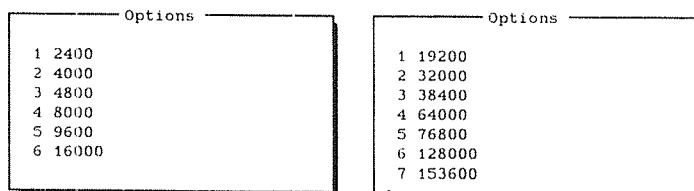


Fig 7.9: HDLC baud rate - LOW and HIGH option windows

To change the current setting, move to the required option in the options window, then press RETURN.

For Computerlink, the 76800 setting is normally used.

PIOC baud rates

You may note that the "industry-standard" baud rates:

2400, 9600, 19200, 38400, 76800, 153600

are interspersed with Norsk Data's PIOC baud rates:

4000, 8000, 16000, 32000, 64000, 128000.

7.4.3 Butterfly Supervisor: Serial Port 1 & Modem Setup

If the system is fitted with a modem, this facility enables you to set it up for communication with another ND computer, using ND's asynchronous communications facility.

Selecting this option

On selecting the Serial Port 1 & Modem Setup option, a further menu offers the following choices (fig 7.10):

```
Main Menu -> Serial Port 1 & Modem Setup
1 Originate Modem Connection
2 Auto-answer Modem Connection
3 Terminate Modem Connection
4 Serial Port 1 baud rate :
5 Exit
```

Fig 7.10: Serial Port 1 & Modem Setup - main menu

As usual, you may move to any of these options and press HELP to bring up its associated Help window.

- Originate : Supervisor does the dialling to establish a telephone connection to a remote computer.
- Auto-answer : Dialling comes from a remote computer. The modem connected to Butterfly-110 is ready to respond to any computer which dials its number.
- Terminate : Resets the modem. In doing so, any existing modem connection is terminated, and the modem is taken out of any previously set Auto-answer mode.
- Ser Bd Rate : Sets up the baud rate for serial port 1.
- Exit : Returns to the Main Menu.

**Originate Modem
Connection**

Selecting this option allows the local Butterfly to originate a modem connection to a remote computer. The display asks for the telephone number of the remote site to be entered.

The Help window for this menu gives prompts on how the telephone number should be keyed into this display. The only acceptable characters are:

- T where Butterfly-110 is connected to a telephone exchange which recognises the number dialled by the pitch of the tone it generates
- P where this telephone exchange recognises the number dialled by the number of pulses (clicks) it generates
- numbers 0 to 9 # and *
- , which introduces a 1-second delay between the preceding number and the next. This caters for the requirement in some telephone exchanges where one has to wait for the next tone before continuing to dial. If necessary, several commas can be inserted together to introduce an equivalent number of seconds delay.
- separators () - and SPACE, to punctuate the number and so make it easier to read on the display.

In the following example, the number is made up of an outside-line code to an exchange which recognises tone dialling (T9), then the STD code (in brackets to make it easier to read) and number of the Remote Telefix modem, whose exchange recognises pulse dialling. A 2-second delay is introduced between dialling for the outside line and starting to dial the STD code. The space between the STD code and the final number is included to make it easier to read:

T9,, P(0234) 56789

Press ENTER to start dialling using the number entered. If it contains any characters that Butterfly-110 does not recognise, the system gives a beep on its built-in loudspeaker. If this occurs, check for an unacceptable character in the number, and correct it as necessary.

The system then tries to establish the telephone line connection. It displays progress messages to indicate how it is progressing:

Resetting Modem ...

Programming Modem ...

Dialling ... (<Esc> to terminate dial-out)

and finally

Connection Established

An error message appears if the system is unable to establish the telephone connection. Any such message is displayed beneath the box area on the screen.

If an error message is reported, this attempt to establish the connection is terminated. You then have to decide whether to try again, or to talk to the remote computer site and perhaps ask them to attempt to establish the connection (with you in Auto-answer mode - see below).

Once the connection is established, the program returns to the Serial Port 1 & Modem Setup menu, and the connection is available to Butterfly-110's COSMOS ASYNC. facility.

Auto-answer

Selecting this option allows the Butterfly-110 modem to be set up so that it will automatically answer incoming telephone calls, and establish a modem connection with the calling site.

On entering this option, you see on the display the progress messages for resetting the modem, then programming it to auto-answer incoming calls. When the modem has been programmed, the display returns to the General Modem Setup menu. If for any reason the programming process fails, an appropriate error message is displayed on the bottom line of the screen.

After Auto-answer mode has been set up, whenever a call arrives, the modem attempts to establish the connection with the calling site. When this has been achieved, the connection is ready for use. If the attempt to establish the connection fails, the modem returns to its quiescent Auto-answer mode and awaits the next incoming call.

Terminate	<p>Selecting this option terminates any existing modem connection, and takes the modem out of Auto-answer mode if this was set.</p> <p>A progress message</p> <p>Resetting Modem ...</p> <p>is displayed while the reset operation is in progress. When it is completed, the display returns to the Serial Port 1 & Modem Setup menu.</p> <p>If for any reason the reset operation is unsuccessful, an error message is displayed. In this case, try to reset the modem manually, either using a reset switch (if fitted) on the modem itself, or by switching its power off then on again so that it re-initialises to its start-up condition.</p>
Serial baud rate	<p>This enables setting up the baud rate for the asynchronous communications port on the PC (serial port 1). The system displays the current setting, and the Help window indicates the range of valid option settings. If no setting currently exists, it is displayed as "Undefined".</p>

7.5 MOVING THE EQUIPMENT

CAUTION

It is particularly important that BEFORE YOU MOVE THE SYSTEM UNIT, YOU:

- ensure the hard disk heads are "parked"
- take appropriate action to avoid bumps, jolts, or any other mechanical shock which could cause the delicate recording heads on the hard disk, floppy disk and tape streamer (if fitted) to receive a damaging collision.

Parking the heads

Run the "park-heads" utility, to ensure that the hard disk heads are in a safe position. To run this utility, go into MS-DOS and select it:

```
>cparkdisk ↵
```

The system then either indicates that parking is not necessary:

Parking not necessary for this disk type

or proceeds to move the heads to the parked position and then beeps until you switch power off.

Switch off

Do not move the equipment, even for the shortest distance (e.g. from one desk to an adjacent desk) without first switching off the mains power to all units.

Remove the on/off key, and tape it to the outside of the System Unit, for safe-keeping.

Use the floppy transit disk	The floppy disk has a cardboard "transport" disk which, when inserted in the floppy drive, ensures that the floppy drive heads are held apart (i.e. they cannot crash together if the unit does receive a jolt). Insert this "transit-disk" into the drive before moving the System Unit.
Disconnect	Before moving the equipment, even for a short distance, disconnect the mains supply cables, then disconnect one end of each cable connecting the Butterfly units together. Coil each cable into a loom and tape it to the equipment.
Moving	<p>When you have:</p> <ul style="list-style-type: none">• parked the hard disk's heads• switched off power and removed the on/off key• inserted the floppy transit disk into the floppy drive• disconnected the mains cables, and one end of each cable connecting the Butterfly units together, then coiled and restrained them, <p>... <u>THEN</u> the equipment can be safely moved.</p>
Long distance?	<p>If the equipment is being moved further than hand-carrying distance, pack it into its original boxes, complete with its original packing material.</p> <div><p>WARNING</p><p>Do not ship any unit of Butterfly equipment for commercial transportation unless it is properly packed in its original material.</p></div>
Re-installing	When the equipment has been moved to the new site, follow the recommendations in the Teamstation Installation Guide, regarding unpacking, locating, re-connecting, and setting up the software.

7.6 CORRECTIVE MAINTENANCE

7.6.1 Basic system checks

Power ON	<p>The indicator inside the ON/OFF keyswitch is powered from the +5V rail, and thus when this is illuminated it confirms that the +5V supply rail is present.</p> <p>The display is powered from the switched mains outlet of the System Unit. To check that the display EHT is present, increase the brightness control until the picture area on the screen is illuminated by the background intensity.</p> <p>The keyboard is powered from the System Unit. The power is +5V dc. A simple way to check that keyboard +5V is present is to press the CAPS key and verify that the internal LED inside this keycap illuminates.</p>
Added features	<p>Butterfly-110 in its simplest form is an Ericsson Ws-286 with a ND-110PCX computer inside it. The PC runs MS-DOS, while the ND runs SINTRAN, and the two sides of the machine (PC and ND) cooperate with each other to share the terminal devices (keyboard/mouse and display) and the disk storage, plus printer and communications facilities.</p> <p>Many "added-value" PC expansion cards, available from diverse sources, may be plugged in to Butterfly's standard PC-AT expansion card. Likewise, a multitude of PC applications may be run on it.</p> <p>Therefore check that any problem on Butterfly is not caused by such "added-value" items.</p>
Error messages	<p>In general, if a problem arises, an error message appears on the display, indicating the cause of the problem. Error messages may originate from the PC or the ND, and may be generated by the respective operating system (MS-DOS in the PC, SINTRAN in the ND) or from an application or utility program. A good understanding is therefore necessary to enable correct interpretation of error messages.</p>

Message log file The message log file receives all messages output to the ND Console (Terminal 1). If the System Supervisor program is running the Console option, these messages are output on the Console's "terminal 1" (i.e. the PC display). Otherwise, messages are output to a "message window" comprising the top three lines of the display, and the system beeps. Messages displayed in this window temporarily overwrite what was previously displayed. Then, pressing any key causes the message to disappear, and the previous display is restored.

Messages may appear at any time. If the display is in 40-column state and a message line is longer than this, the message is truncated. If the message is longer than three lines, it is scrolled so that all of it can be read.

The system uses two files for storing Console Messages:

- the current file: CONSLOG.CUR
- the previous file: CONSLOG.PRV.

It writes messages into the "current" file, until that file becomes full. Then it switches to writing to the other file, which becomes the new "current" file (i.e. CONSLOG.CUR), while the full file is renamed "previous" (i.e. CONSLOG.PRV). Then, when the second file also becomes full, it clears out all the messages from the "previous" file, swaps the filenames, and proceeds to use the cleared file as its new "current" file. Each file can store an average of 200 messages. To see the most recent messages, you therefore read the CONSLOG.CUR file.

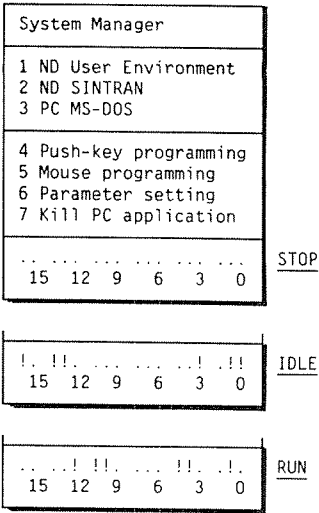
The system enters the system date into the Message file each time Butterfly-110 is started-up. Thereafter, only the time of each message is entered, unless the system date changes during a session, in which case the new date is entered.

To read the message file, enter MS-DOS, and use the "type" command:

```
>C TYPE \BFLY-110\CONSLOG.CUR ↵
```

While the file is open in this way, it is unavailable for the system to write new messages into it.

Active level display The active program level of the ND-110PCX is indicated in the bottom area of the ND's System Manager window (fig 7.11).



The characteristic patterns of this display give an immediate indication of the status (stop, idle, run with users active) of the ND side of the machine.

- If the ND is in Stop mode, or has not yet started after a power-up sequence, the display will be blank, i.e. no "1s" present (fig 7.11 - STOP).
- If the ND is running, but is in an IDLE condition, the active levels display shows a characteristic pattern of "1s" (fig 7.11 - IDLE).
- If the ND is running with users active, the active levels display shows a characteristic RUN pattern (fig 7.11 - RUN). Other "1s" may flicker, indicating program level changes in response to user activity.

Fig 7.11: System Manager - ND "active level" display Note that the "active level" display is not updated when in OPCOM.

- Checking cables and connections Check that the cables connecting the equipment together:
- are not subject to any mechanical stress, or twisted or tangled around other cables or furniture. Any excess cable length should be gathered in a tidy loom and held together by an appropriate cable-tie.
 - are securely connected to the equipment.
 - are undamaged. Any signs of fraying, cuts in the outer insulation, or damaged strain-relief fittings at the connectors themselves, should be rectified without delay.

Location of equipment The Teamstation Installation Guide (ND-30.056) lists the factors that should be taken into account when deciding where to place the equipment.

In particular, ensure it is not placed in direct sunlight, which could lead to overheating.

7.6.2 Power-on self-test

7.6.2.1 Start-up sequence

The Ericsson PC has built into it a set of self-test program routines which run automatically each time the machine is powered up. Similarly, the ND-110PCX has its own set of self-test program routines.

These self-test programs check that each side (PC and ND) of the machine is functioning correctly, and that the two sides of the machine are able to communicate with each other.

The sequence in fig 7.12 describes what happens during a Butterfly-110 start-up sequence. If the sequence stops before completion, the screen display indicates whether the problem lies within the PC or the ND side of the machine.

7.6.2.2 Self-test operations

PC-ND tests

Once MS-DOS is loaded, it invokes the AUTOEXEC.BAT file which in turn starts a Butterfly-110 "normal start-up" program. If this program completes without error, then the system initialisation programs that follow in the AUTOEXEC.BAT file are invoked.

The "normal start-up" program tests secret memory as seen by the PC, and then monitors the progress of the ND self-test program. Summarising its activities, it:

- takes a copy of the ND "secret memory", then performs a destructive test on that area. On successful completion of this test, the copy is restored into secret memory.
- clears the error logging location ERRLOG in NV-RAM.
- writes a pattern of four words into four consecutive locations in the secret memory. There are two patterns, one indicating that "normal" operations are required, the other indicating that "diagnostic" operations are to follow. Note that "diagnostics startup" is embedded in the Diagnostics program.
- sets ND self-test status flag (STS-FLAG) to "not used".

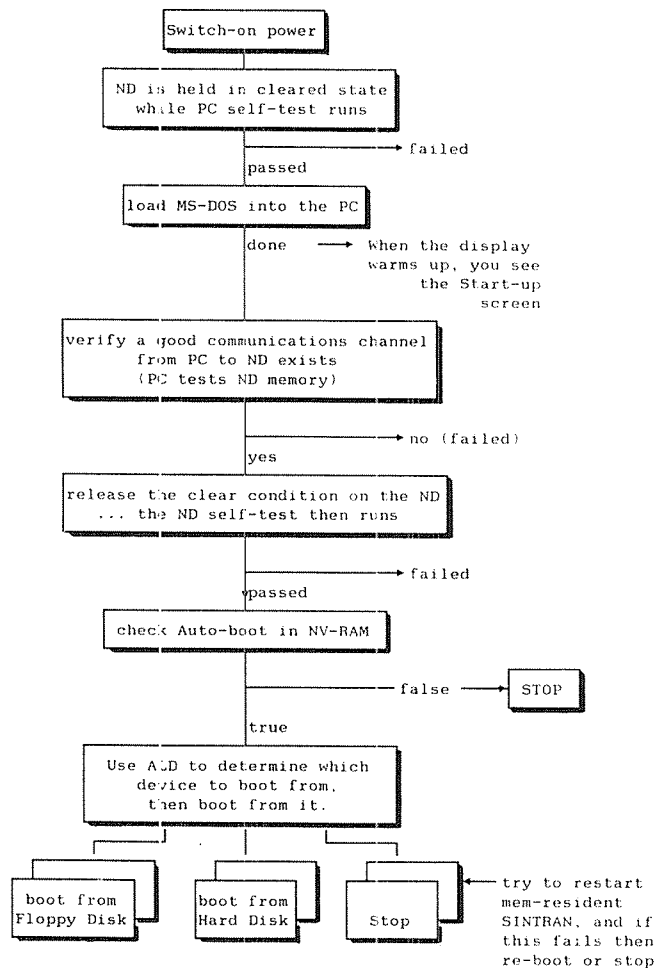


Fig 7.12: Butterfly start-up sequence

- issues the PCLETGO signal, causing the ND to leave master clear and begin its self-tests. See ND self-tests, below.
- monitors the progress of the ND self-testing over a time-out period, by repeatedly testing the STS-FLAG and ERRLOG. Each time an error is entered into ERRLOG it results in an error report and a different path through the AUTOEXEC.BAT file which on completion results in the System Manager not offering SINTRAN as an option to proceed.

A "passed" flag in STS-FLAG results in the normal sequence through the AUTOEXEC.BAT file, ultimately enabling SINTRAN as a user option.

ND self-tests

These start to run as soon as the PC issues the PCLETGO signal. They comprise the standard ND-100 self-tests plus three simple non-destructive memory tests - one for secret memory DRAM, one for all of the ND's DRAM, and one which tests the secret memory dual-port mapping between the ND and the PC. The tests are executed in that order, and are "stop on error" types of tests.

ND-100 standard self-tests:

The only modification to the standard self tests required is that to the method of error reporting. Error codes are recorded in NVRAM at the ERRLOG location, the STS-FLAG is set to ERROR and the BOOT Flag set to 1.

Secret Memory Test

This test non-destructively tests the "secret memory" DRAM, using an address-in-address, plus compliment of address-in-address, test. The test operates on a four-word test area at a time. For each test area the data is preserved by copying it into four words of the Register File. Then the test area is destructively tested. On completion of testing that area, the data preserved in the Register File is restored to that tested area. This procedure is repeated for each consecutive four-word area in "secret memory" DRAM.

This test procedure only detects any stuck-at fault in the lower three bits of the memory address register or in the decoder (due to the reduced range of the test). However, on the assumption that those two functional areas are fault free, the test can detect any stuck-at fault in the memory array, in the memory data register, or in the read/write logic. Due to the reduced range of this test, it can only guarantee that four words in the memory array are functionally correct (a fault could have occurred in the decoder such that the same four words are always referenced).

Since the PC-OPCOM channel, described later in this section) only requires four functionally correct words of memory for PC-OPCOM communication, then the reduced range is acceptable for this self-test. A comprehensive PC-ND interface diagnostic package uses the PC-OPCOM channel to load program into the ND and fully test secret memory.

Main Memory Test

This checks all main memory DRAM non-destructively. The memory is checked to determine whether it is 0.5 or 1.0 Mwords (1 or 2 Mbytes) long. Each is tested by checking that each bit (including the two parity bits) can be inverted. The paging system has been inactive up to this point in the self-testing: thus testing memory 0-64k addresses the shadow memory page tables (occupying only the upper 2k) and physical memory 0-62k. When paging is set active, the physical memory 62-64k (shadow overlap memory) is tested. Thus the main memory test checks 0-62k, the page tables, 64k-"top of memory", and then enables paging to test the 62-64k DRAM. Paging is disabled after this test.

Note that main memory DRAM has a "hole" between 3C000h and 3FFFFh, and should not be tested. A further "hole" may exist from 40000-50000h if the PC has been given 64kwords of the ND's memory space.

Assuming no decoder/addressing faults, this test procedure will detect any stuck-at fault in the main memory array (including parity bits) in the memory data register or in the read/write logic.

PC-ND Shared (dual-port) Memory Mapping Test

The dual-port test involves the ND checking a sequence of locations in the shared memory area, to find either the "normal" or the "diagnostic" four-word patterns which the PC has written into a pre-defined address area (called st-mail, see below) in secret memory. These patterns are made up so as to represent non-valid CPU instruction codes.

- The "normal" pattern indicates that the PC requires the ND to check the Auto-boot and ALD flags, to determine whether and from where it should boot normal program.
- The "diagnostic" pattern forms a request to enter a diagnostic mode that offers a Butterfly-110 version of OPCOM to the PC.

If the ND fails to find either of these patterns at the expected location, then the mapping of shared memory between the PC and ND has failed. In this situation, the ND searches through its memory in an attempt to find either of the key patterns, in any consecutive four-word area.

If the deposited pattern is then found (at the incorrect location), then this indicates that a mapping fault caused the PC memory window to point to the incorrect area of ND memory. However, it still identifies a "known good" four-word channel through which the PC can communicate with the ND.

The following responses to the possible outcomes of the search may arise:

- Neither pattern is found after a full search. In this case, error 22h is recorded in ERRLOG, the STS-FLAG is set to "error", and STOP invoked.
- Normal pattern is found at the correct location. This sets the STS-FLAG to "passed", and "normal boot up" is invoked.
- Diagnostic pattern is found at the correct location. This sets the STS-FLAG to "passed", and invokes the OPCOM Driver.
- Normal pattern is found at the wrong location. In this case, error code 20h is recorded in ERRLOG, the STS-FLAG is set to "error", and STOP invoked.
- Diagnostic pattern is found at the wrong location. In this case, error code 21h is recorded in ERRLOG, the STS-FLAG is set to "error", and the OPCOM Driver invoked.

The OPCOM Driver uses the "diagnostic OPCOM channel" known as "st-mail" - see section 7.6.2.5: Self-test mailbox.

7.6.2.3 Self-test status flag (STS-FLAG)

The STS-FLAG is used by the ND to indicate its self-test status to the PC. On start-up, it is set by the PC to NOT USED, and thereafter is set by the ND and checked by the PC.

It is a single 16-bit location in ND memory. It may have three states:

- NOT USED : 006Ch set by PC
- ERROR : 00B4h set by ND
- PASSED : 00D8h set by ND.

7.6.2.4 Self-test error log (ERRLOG)

ERRLOG is used by the ND to indicate the type of error identified by a self-test. It occupies one byte (8 bits) in NV-RAM (see Chapter 6 fig 6.29: Secret memory - NV-RAM address allocations).

Errors occurring in any of the self tests are recorded by copying the appropriate error code into ERRLOG and then flagging ERROR status in STS-FLAG. The PC can then check the STS-FLAG, and if necessary read the information in ERRLOG.

The table in fig 7.13 shows the error code values that may appear in ERRLOG.

Value (hex)	Description
00	no error
01-0D	ND-100 standard ND self-test error codes
10	secret memory address-in-address error
11	secret memory compliment-of-address-in-address err
12	existence test error
13	main memory compliment of data error
14	main memory parity error
15	main memory compliment of parity error
16	main memory restore data error
17	existence test error
18	main memory compliment of data error
19	main memory parity error
1A	main memory compliment of parity error
1B	main memory restore data error
1C	main memory compliment of data error
1D	main memory parity error
1E	main memory compliment of parity error
1F	main memory restore data error
20	mapping error - normal pattern found
21	mapping error - diagnostic pattern found
22	mapping error - pattern not found

Fig 7.13: ND self-test - ERRLOG codes

7.6.2.5 Self-test mailbox (st-mail)

"st-mail" is a 4-word area of secret memory as seen by the PC (for address information, see Chapter 6 fig 6.30: Secret memory - DRAM address allocations), into which it can deposit one of two patterns:

- normal : D1AEh E555h 4000h C062h
- diagnostic : D1AEh C7FFh C062h A800h

As part of its self-test (shared memory mapping), the ND searches for either one of these patterns.

Normal pattern If the ND self-test finds the "normal" pattern, then after successful completion of the self-tests, the ND checks the Auto-boot and ALD flags, to determine whether it should commence to load normal program, and if so, from where (i.e. which device) it should boot.

Diagnostic
OPCOM channel If the ND self-tests find the diagnostic pattern, this four-word self-test mailbox area is used as an OPCOM channel between the PC and ND. Diagnostic programs may then be loaded from the PC, through this known "good" channel, into previously-tested areas of ND memory, and then run. Note that in Diagnostics mode, OPCOM polls st-mail (or wherever the image of st-mail is found); no use is made of attention or configuration flags, watchdog timer, ND active levels, or IOX simulators, i.e. there is no use of secret memory other than the four consecutive st-mail locations. In this way, diagnostic tests may be run on the ND using only proved parts of the system.

When this OPCOM channel is used, the st-mail locations are used as follows:

	Viewed by ND	Viewed by PC
Lower byte word 0	RxD	TxD
Lower bit word 1	RxRdy (1)	TxRdy (0)
Lower byte word 2	TxD	RxD
Lower bit word 3	TxRdy (0)	RxRdy (1)

where RxD - Receive Data Holding Register
 TxD - Transmit Data Holding Register
 RxRdy - Receive Data Register Ready
 TxRdy - Transmit Data Register Ready

7.6.3 Diagnostic programs

7.6.3.1 **Butterfly diagnostic suite**

Program load To start the Butterfly Diagnostic program, insert the Butterfly-110 Diagnostics disk into the floppy drive and press CTRL+ALT+DEL.

When the program is loaded, type in the ND user-SYSTEM password. Once the password is entered correctly, the system presents the Butterfly Diagnostics Main Menu (fig 7.14).

Butterfly Diagnostic Program		Version xxx
Main Menu		
1 PC Diagnostics 2 PC-ND Interface Diagnostics 3 ND Diagnostics 4 PC Setup Utility 5 Exit		
Select option using ↑, ↓, and RETURN or by number or by name and RETURN		EXIT to terminate program HELP for help

Fig 7.14: Butterfly Diagnostics - main menu

Select option The display highlights the currently selected option. The lower part of this Main Menu explains how to move to any of the other options.

A Help message is associated with each option. Press the HELP key to bring up this message in a window on the Main Menu display.

Separate ND diagnostics Currently, the ND Diagnostics option is not implemented in this program. Instead, the ND Diagnostic tests are contained on a separate floppy disk. Program on this disk is loaded from OPCOM - see ND Diagnostics (section 7.6.3.5).

Exit

At any time, the current operation can be terminated by pressing EXIT (or ALT+F10) or selecting the Exit option on a menu. Program then returns to the previous menu.

If you are in the Main Menu and exit, the program stops and

Butterfly Diagnostic Program Terminated

is displayed.

Diagnostics test suite

Fig 7.15 lists all the test options available, and shows how to move through the diagnostic menu structure. Note that the third line of the menu displays for PC-ND Interface and ND Diagnostics tests indicates where you are in this menu structure.

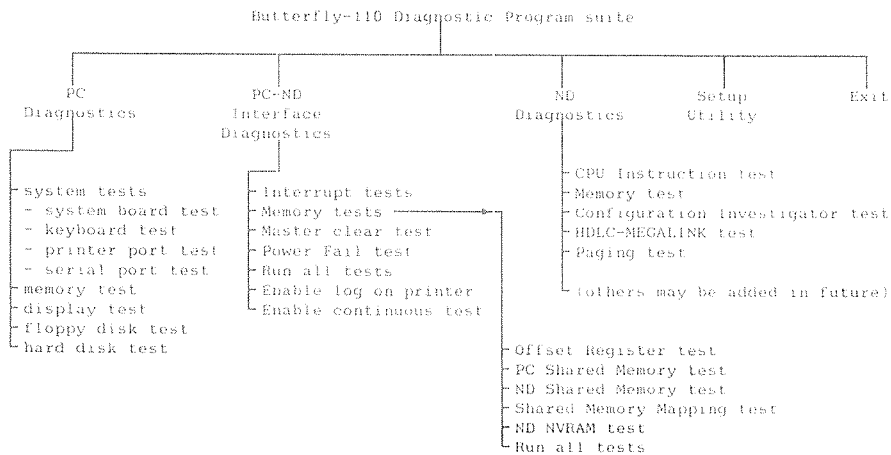


Fig 7.15: Diagnostics menu structure

The Help message window for each option gives a guide as to what features of Butterfly-110 are tested by that option.

7.6.3.2 Which diagnostic?

When deciding which diagnostic program to run, it is useful to consider the basic structure of Butterfly-110.

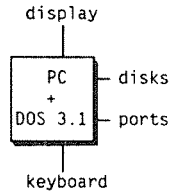


Fig 7.16: The PC block

The PC Diagnostics check that the PC hardware works correctly. They include tests on the disks, keyboard, display, and any printer that is installed.

The Ericsson WS286 PC runs under EIS DOS version 3.1. This is the Ericsson version of Microsoft's MS-DOS 3.1. All operations to load SINTRAN (or any other program) into the ND side, and for the ND to work with the keyboard, display, hard and floppy disks, printer and serial ports, and any additional expansion bus boards, rely on the PC and its DOS operating system software being fully functional (fig 7.16).

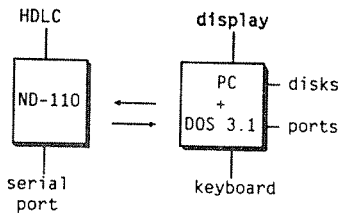


Fig 7.17: PC block and ND block

The ND-110PCX in Butterfly-110 runs alongside the PC's 80286 microprocessor (fig 7.17).

- In order to get the ND's SINTRAN operating system into the ND-110PCX, it must be taken from disk by the PC and passed across to the ND-110PCX. Until this is achieved, the ND-110PCX can do nothing useful.
- In order for the ND-110PCX and PC to operate together, they must be able to share data and pass specific tasks to each other.

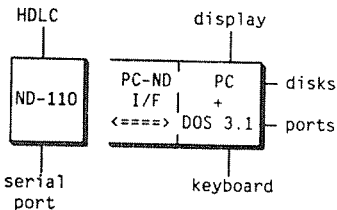
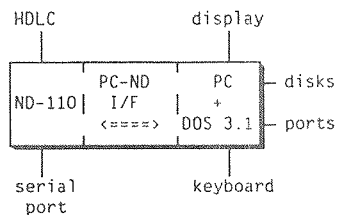


Fig 7.18: PC-ND interface

This communication is done over the PC-ND Interface (fig.7.18).

It is clear that no operation can be achieved by the ND110PCX unless this interface is fully functional. The PC-ND Interface diagnostics test this.



When the PC and the PC-ND Interface have been checked, the ND-110PCX itself may be checked, using the ND Diagnostics (fig 7.19). These tests include checks on the ND's HDLC, memory, instruction set and configuration.

Fig 7.19: Butterfly-110 structure

Note that whenever the ND-110PCX:

- takes input from the keyboard,
- outputs to the display,
- performs an operation with hard or floppy disk,
- uses any other port on the PC,

it relies on the PC and the PC-ND Interface being operational.

7.6.3.3 PC diagnostics

Fig 7.20 shows the Main Menu help window for PC Diagnostics.

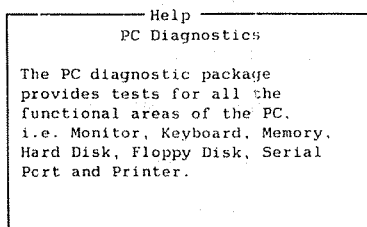


Fig 7.20: PC diagnostics - Help window

The PC side of Butterfly-110 is tested using the suite of test programs provided by Ericsson Information Systems (EIS) for their PC. Within Ericsson documentation, this suite is referred to as the Ericsson Maintenance Program. Parts of it are modified to comply with the operational characteristics of Butterfly-110.

PC Diag menu

On entering PC Diagnostics, you are offered the following options on the display (fig 7.21):

MAIN MAINTENANCE MENU	
System configuration	Error log menu
System test menu	Setup menu
keyboard test menu	Enter configuration mode
Color graphic video menu	Run configuration
Diskette drive menu	External command
Hard disk menu	
Exit	
Use arrows to select option <Enter> = Execute <H> = Help <digit> = Number of tests <Esc> = Back to previous menu <C> = Continuous testing	

Fig 7.21: PC Diagnostics - main menu

Select option	Select any option as indicated at the foot of the display. Each option has its own menu, which includes information on how to run the test(s).
HELP	To find out more about what an option does, move the cursor to that option and then press HELP. Information then appears in a "help" window on the display.
System test	<p>This diagnostic tests:</p> <ul style="list-style-type: none">• the main System Board in the PC.• the Keyboard <p>This checks the operation of the keyboard and the functions of the keys. It displays a picture of each key position. Press each key in turn to check that it inputs correctly to the system.</p> <ul style="list-style-type: none">• the printer port <p>This test checks the parallel port on the system board, and if a printer is connected, it outputs a sample printout. The correct printout test pattern is shown in fig 7.22, for comparison with any printout that you generate from this printer test.</p> <p>If a test printout appears to be faulty, the fault may be in the printer, or in its connecting cable, or on the system board. If the printer has a self-test switch, run the self-test to check whether the printer is functioning correctly. Refer to the Printer manual for details of its self-test facility.</p> <ul style="list-style-type: none">• the serial port. <p>This test checks the asynchronous communications line. To run this test, first fit the Test Terminator plug to the serial port. This plug loops the serial output back into the serial input, so enabling the PC to send characters and check that it then receives back the same characters. The Test Terminator plug is part of a Butterfly service engineer's kit.</p>
Memory test	<p>This test checks the PC memory, including the extra 128kbytes donated from the ND if this option is selected. The display indicates the memory area that is tested, in 64kbyte segments,</p> <p>i.e. segments 0 - 8 cover addresses 0 - 512k segments 9 - A cover addresses 512 - 640k.</p>

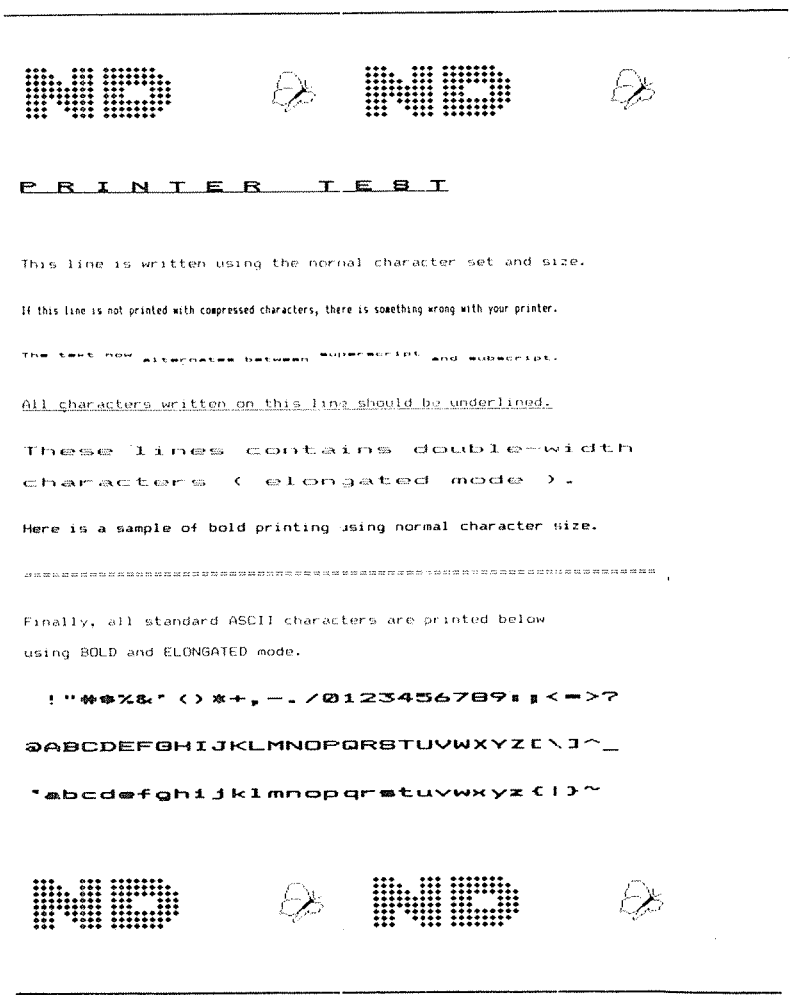


Fig 7.22: PC Diagnostic - printout from printer test

Display test

This "video" test checks both the Display Unit and the EGA Board.

It displays a test pattern on the screen. The test pattern shown depends on the display mode. Three examples are given here, for the Colour 80x25 mode (fig 7.23), the 40x25 mode (fig 7.24), and the Graphic 320x200 pixels mode (fig 7.25). These examples illustrate the kind of screen display you need to check.

Fig 7.23:
PC Diagnostic
- colour 80x25
display test

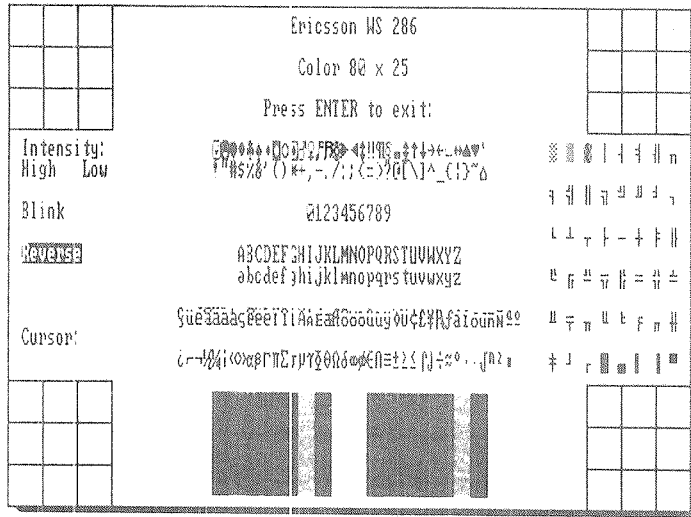


Fig 7.24:
PC Diagnostic
- 40x25
display test

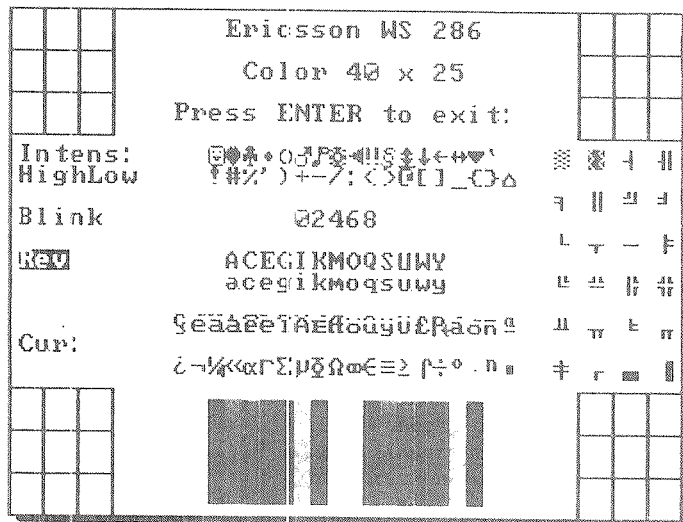


Fig 7.25:
PC Diagnostic
- graphic 320x200
display test



Floppy drive test

CAUTION

Tests involving writing to a floppy disk will destroy existing data

This "diskette drive" test checks the floppy disk control and read/write logic of the Disk Controller, and the operation of the Disk Drive. Three types of test may be selected:

- sequential read, or read/write
- random read, or read/write
- measurement (for service engineer's use only).

Having made a selection, a second menu is displayed, asking you to confirm or change the test parameters. Note that "Run test" specifies the number of test cycles to be done - this may be between 1 and 999, or C for continuous.

You need formatted floppy disks to run the read or read/write tests. Floppy disks used in read/write tests must be reformatted before being reused for normal data storage.

Hard disk test**CAUTION**

Tests involving write operations to disk may destroy existing data.

This test checks the hard disk parts of the Disk Controller, and the operation of the Winchester Disk Drive.

The Hard Disk menu offers six types of test. On selecting a test, a second menu is displayed, asking you to confirm or change the test parameters. These include the name of the Drive (C), the cylinder and sector address range to be tested, the test pattern to be used in "write" tests (this is entered as four hexadecimal characters), the number of retries to be made if an operation fails, and the number of test cycles to be performed (1-999, or C for continuous).

The tests are:

- sequential read

Each test cycle reads each cylinder in the range you specify, starting from the lowest cylinder up to the highest cylinder and then back again.

- sequential read/write

This test writes to the lowest cylinder, then writes to the lowest cylinder + 1, then reads from the lowest cylinder, then writes to the lowest cylinder + 2, then reads from the lowest cylinder + 1, and so on to the highest cylinder. It then works its way back again, writing the highest cylinder - 1, reading the highest cylinder, writing the highest cylinder - 2, reading the highest cylinder - 1, and so on, back to the lowest cylinder. The test repeats for the specified number of cycles.

- random read,

Each test cycle reads randomly chosen sectors on 100 randomly chosen tracks, within the cylinder/sector address range specified.

- random read/write

Each test cycle performs a write then read operation on randomly chosen sectors on 100 randomly chosen tracks. The write followed by read operations proceed in the manner described for sequential write/read testing, so that the disk heads have to move to a new cylinder address between each write and read operation.

- verify

This test performs the DOS command "verify track", between the lowest and highest entered cylinder addresses. It is equivalent to a read operation, except that no data is transferred into memory.

- max cylinder read/write.

This test writes the specified pattern to the last cylinder, then reads it back. Since the last cylinder is not used by DOS, this is the only "write" test which does not affect any DOS information that may already exist on the disk.

Configuration
mode

This option enables a user to combine a sequence of tests from any of the PC Diagnostics menus. It includes facility to set up test parameters.

Error log

The Error Log records error messages for each error condition that is detected. You can view, print, clear, or send the Error Log file to disk: the options available are presented in the Error Log menu.

7.6.3.4 PC-ND interface diagnostics

PC-ND environment As described in section 7.6.3.2 (Which diagnostic?), Butterfly-110 hardware comprises a ND-110PCX coexisting with an IBM-compatible PC-AT (30286 microprocessor). These two processors share the PC's I/O resources (screen, keyboard & mouse, floppy and hard disk, comms port). The ND requests use of these resources via its IOX instruction, using a "mailbox" communication protocol. The hardware of the mailbox comprises a shared-memory window, one interrupt line in each direction (PC → ND) and PC ← ND), and various status lines.

Communication between the PC and the ND relies on this mailbox interface being fully operational. Depending on the severity of a failure in the interface, the PC-ND power-up self-tests might be sufficient to detect and point to the cause of failure. Otherwise more comprehensive diagnostic tests are necessary, in which case a method that relies on only a partially tested interface is required for loading such tests across the suspect PC-ND interface.

HELP

The Help window for the PC-ND interface diagnostic is shown in fig 7.26.

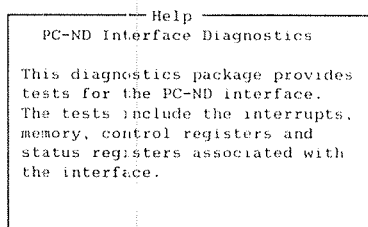


Fig 7.26: PC-ND Interface diagnostics - Help window

PC-ND I/F diag options menu

On selecting this option, the PC-ND Interface Diagnostics options menu is displayed (fig 7.27).

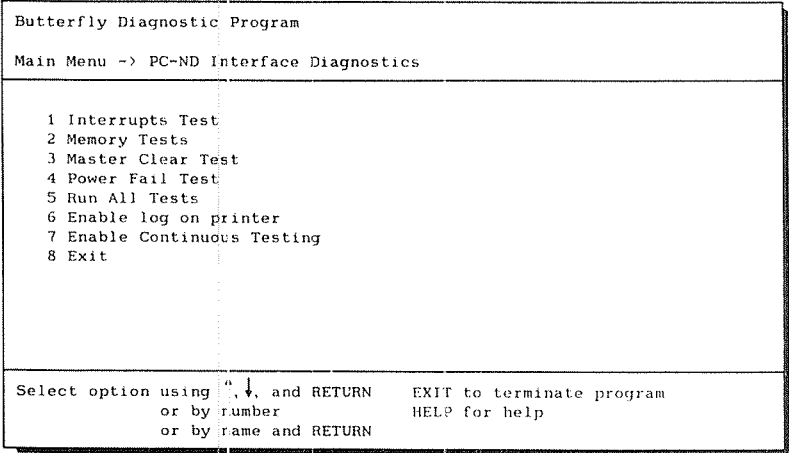


Fig 7.27: PC-ND Interface diagnostics - options menu

As usual, press the HELP key to obtain a summary description of what each test does. Test result(s) may be routed to a printer (if fitted). In all cases, the test results, and any error messages, are displayed on your screen (and printed if the "enable log on printer" option was selected).

Memory tests

In the case of the Memory tests, the system presents a further menu, which offers the following options:

- Offset Register test
- PC Shared Memory test
- ND Shared Memory test
- Shared Memory Mapping test
- ND NVRAM test
- Run all tests
- Exit

Again, each of these has its own Help window to indicate what each test does.

Run all tests

As before, the "run all tests" option runs each test in the menu sequentially, and presents a "Test Passed" or "Test Failed" summary of results at the end.

Tests summary	<p>The remainder of this section gives a brief functional description for each test in the PC-ND interface diagnostic.</p> <p>The general structure of these tests involves:</p> <ul style="list-style-type: none">• preparing the ND for diagnostic mode of operation• running the test• giving the test result.
Prepare in Diag	<p>This sets up the ND so that it is ready to run in diagnostic mode. As outlined in section 7.6.2 (Power-on self-test), this involves resetting the ND, writing a 4-word (in this case "diagnostic") pattern to secret memory, then releasing the ND to run its self-test program and monitoring this to completion.</p> <p>It:</p> <ul style="list-style-type: none">• generates a simulate powerfail (PCSIMF)• generates HMCL• resets ND-PC interrupt and enable memory (LKEY and Keys register)• points to a specific 64k address area in secret memory (LOFF and Offset register)• disables PC parity check• clears secret memory (and parity errors)• enables PC parity check• if necessary, clears ERRLOG (in NV-RAM)• writes the 4-word Diagnostic Pattern to secret memory• issues PCLETGO• monitors the STS-FLAG and ERRLOG over a timeout period, by repeatedly testing the STS-FLAG and ERRLOG. An error logged results in an error message.
Interrupts test	<p>This test is not available in the first release of the system. An appropriate system message indicates the current status of the software when this test option is entered.</p>

Memory testsOffset register1 Prepare the ND in DIAG Mode

This is described above.

2 Offset register read/write test

- checks that the register has no stuck bits
- checks that the register is not pattern-sensitive (by sweeping through the full 8-bit range 00-FFh).

3 Offset register mapping test

Provided ERFLOG indicates no standard self-test error, program proceeds by:

- loading ND code which will write an id word into offset 4000h of each 32kword block of ND memory.
- using the Offset register to map to each block in turn, checking it is reading the corresponding id word for each block.

PC shared memory test

The PC tests the secret memory DRAM, via the shared memory window. This test comprises two parts:

- "stuck at" faults

This uses the MATS+ test [ref. R NAIR, 1979, IEEE Trans Comp C-27 3 Mar, 258-261], for testing all "stuck at" faults in RAM of arbitrary wired logic behaviour and arbitrary decoder design.

- Coupling faults

This uses Suk & Reddy's Test B (ref D Suk & S Reddy, 1981, IEEE Trans Comp C-30, 12 Dec, 982-985).

ND shared memory test

1 Prepare the ND in DIAG Mode

This is described above.

2 Test memory

Provided ERRLOG indicates no standard self-test error or complete mapping-fail error, the program proceeds by the PC instructing the ND via Diag OPCOM, to test secret memory using microcoded memory tests. The test runs through addresses from start to st-mail, then proceeds from the end of st-mail to the top address in secret memory.

Shared memory mapping test

1 Prepare the ND in DIAG Mode

This is described above.

2 Memory mapping test

Provided ERRLOG indicates no standard self-test error or complete mapping-fail error, the program proceeds by loading ND code via the Diag OPCOM channel, which is then executed. The ND code places a pattern into secret memory DRAM. The PC then checks the pattern.

ND NV-RAM test

1 Prepare the ND in DIAG Mode

This is described above.

2 Read/write to a location

The program reads a location of NV-RAM, and stores the 1's-complement of the value it finds. It then writes to NV-RAM without first opening the location, and checks that the write operation failed. Then, it sets NVOPEN, writes to NV-RAM, and checks the write operation was successful. Finally, it restores the original value in NV-RAM.

Master clear test

1 Prepare the ND in DIAG Mode

This is described above.

2 If no standard self-test error

- checks ND is not in MCL (read Status register)
- generates HMCL
- checks that the ND is in MCL (read Status register)
- waits (on a timeout) for end of the ND master clear operation, then checks ND is not in MCL (read Status register).

Power fail test

1 Prepare the ND in DIAG Mode

This is described above.

2 If no standard self-test error

- checks the ND is not in PF state (read Status register)
- generates PCSIMF
- checks the ND is in "power fail" state (read Status register).

7.6.3.5 ND diagnostics

Separate disk These tests are not included in this release of the Butterfly Diagnostics disk. Instead, they must be loaded and run from the Butterfly-110 "ND Diagnostics" floppy disk.

Program load To load this program:

- ensure that the machine condition is "booted in Normal Mode". If in doubt, switch it OFF, then back ON so that a power-on reset is effected.
- select MS-DOS, then Butterfly Supervisor (C:SUPV)
- select the Console option, and enter Stop Mode (Alt+F2)
- set the Auto-boot flag to - (minus) (Alt+F4)
- master clear (Alt+F1)
- insert the Butterfly-110 ND Diagnostics floppy disk into the disk drive
- check that OPCOM is active, by pressing ↵ and noting that it results in a "hash" character prompt on the display
- type

1560& ↵

to load the test program monitor.

Information Press HELP to call up a display which lists the available monitor commands. Then, use the "list-files" command to display the available test files.

Test options The following tests are currently offered:

- Instruction
- Memory
- Configuration Investigator
- HDLC-MEGALINK.

As in all these test programs, the screen displays test results and any error messages. The tests themselves are described in the ND-100 Test Program Description (ND-30.005).

7.6.4 Butterfly Supervisor maintenance facilities

The Butterfly Supervisor Program provides facilities which enable some of the more specialised tasks on the ND side of Butterfly-110 to be performed easily. It covers:

- Console¹
- Telefix¹
- System Parameter Definition²
- Warm Start³
- Cold Start³
- Serial Port 1 and Modem Setup²

¹ test/diagnostic aids

² setup/configuration facilities

³ system restart facilities

The setup/configuration facilities (type ²) are described under "Installation tools" (section 7.4). The other Butterfly Supervisor facilities are described here.

The Butterfly Supervisor Program is stored on the hard disk. It runs in the PC under MS-DOS. However, it requires the ND USER-SYSTEM system password before it can be run.

Loading program To start the Butterfly Supervisor program, select MS-DOS from the System Manager and then type:

```
c> supv ↵
```

Then, type in the ND USER-SYSTEM password. Once this is entered correctly, the Butterfly Supervisor Program Main Menu is displayed (fig 7.28).

Butterfly Supervisor Program	Version x.x
Main Menu	
1 Console 2 Telefix 3 System Parameter Definition 4 Warm Start 5 Cold Start 6 Serial Port 1 and Modem Setup 7 Exit	
Select option using ↑, ↓, and RETURN or by number or by name and RETURN	EXIT to terminate program HELP for help

Fig 7.28: Butterfly Supervisor - main menu

Those options specifically related to Butterfly-110 maintenance tasks are:

- Console
- Warm Start
- Cold Start
- Telefix.

These are described in following sections.

Display colours

The display colours are fixed, except that when the selected option operates only on the ND side (e.g. Console, Telefix), display colours adopt those specified in the ND Display Manager menu. The fixed colours are:

top area	:	purple
options	:	green
Help window	:	yellow
bottom area	:	yellow
error messages	:	red

Error messages

Error messages are displayed on the 25th line of the display, immediately under the outlined area.

-
- Select option** The display highlights the currently selected option, in inverse video. The bottom area in this Main Menu explains how to move to any of the other options.
- A Help message is associated with each option. Press the HELP key to bring up this message in a window on the Main Menu display. It prompts you with the facilities provided by the highlighted option.
- Exit** At any time, the current operation can be terminated by pressing EXIT or selecting the Exit option on a menu. This returns you to the previous menu. If you are in the Main Menu and press EXIT or select the Exit option, the program stops and the message:
- Butterfly Supervisor Program Terminated
- is displayed.

7.6.4.1 Console facility

The Help window for "Console" lists the functions available (fig 7.29):

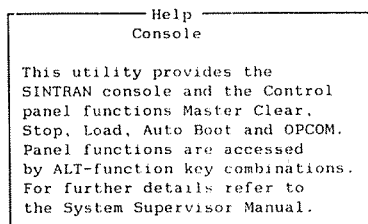


Fig 7.29: Butterfly Supervisor - Console option: Help window

As prompted in the lower part of the display, press RETURN to remove the help window and continue.

Options

The Console options are all ND Control Panel functions which are familiar to users of the Console facility in ND-100 systems. They are used as engineering diagnostic aids, to examine and if necessary modify operating and status information in the ND-110. They include ND's OPCOM facility, which uses a keyboard/display designated "terminal 1" for input and output.

A summary of the function of each facility is given here. For a detailed description of the use and meaning of each facility, refer to the Operator Interaction chapter of the ND-110 Functional description (ND-06.026), and the SINTRAN III System Supervisor Guide (ND-30.003).

Those familiar with the Console facility in ND-110 should note that when in the Console option:

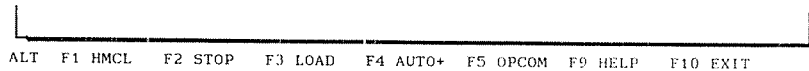
- Butterfly's keyboard and display operate as the "terminal 1" device
- although the functions available are the same as those in the ND-110 Control Panel, they are accessed in a different manner.

Selecting Console

When selecting the Console option, a test is made to see whether the "terminal 1" console is available. If not, the Butterfly Supervisor Program terminates with the message:

Unable to set console

When entering the Console option, a blank display is presented, except for the last (25th) line, which shows the console functions available, and how to select each one (fig 7.30):



```

ALT F1 HMCL F2 STOP F3 LOAD F4 AUTO+ F5 OPCOM F9 HELP F10 EXIT
  
```

Fig 7.30: Butterfly Supervisor - Console function select

The blank 24 lines are in the current background colour specified in the ND Display Manager - the usual default setting is blue. The last line of console functions is in red.

The current status of the ND-110 when in Console mode is shown in the right-hand corner of the last line, in white. It uses the following codes:

P	ND is in Power Fail mode
O	OPCOM is active
R	ND is in Run mode
S	ND is in Stop mode
M	ND is in Master Clear mode.

For example, "POSM" indicates that the ND is in Power Fail, in Stop mode, Master Clear is asserted and OPCOM is active.

ALT+F9: HELP

Press ALT with the F9 function key. The display then shows a summary description of console functions in its Help window (fig 7.31).

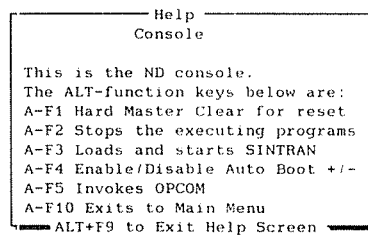


Fig 7.31: Butterfly Supervisor - Console Help option

ALT+F10: EXIT

Selecting Exit returns you to the Main Menu.

ALT+F1: HMCL

The Master Clear process involves resetting the ND-110PCX, then running the Self-tests which check the PC-ND interface and the ND-110PCX (but not the PC), and finally starts up SINTRAN if the ALD register and Auto-boot flag so define. If on the other hand the ALD register defines "stop", the system goes into Stop mode.

- If SINTRAN is started up successfully, the "SINTRAN running" message is displayed.
- If the Auto-boot flag tells the system to go into Stop mode, the OPCOM "hash" prompt is displayed, indicating that the ND's MOPC has control. It is this Microprogrammed Operator's Communication that allows direct operator control of the ND-110PCX, through Terminal-1.

The ALD register and Auto-boot flag are control functions in the System Parameter Definition option (see section 7.4.2). Note that the Console's AUTO option (ALT+F4) can also be used to change the Auto-boot flag.

Any error messages will be reported on the display. These take the form of an error number with a brief explanation of the error.

ALT+F2: STOP

This is selected to put the ND-110PCX into Stop mode. In this condition, all current programs stop and the MOPC (Microprogrammed OPERator's Communication) program runs. The "hash" prompt on the display indicates that the ND's MOPC has control. It is the MOPC program that allows direct operator control of the ND-110PCX, through Terminal-1. See also "OPCOM" below.

The ND can also be put into Stop mode by selecting the HMCL function when Auto-boot is disabled.

ALT+F3: LOAD

This function works only if the ND's MOPC is running. If so, selecting LOAD allows you to start an operation to load program from whatever device is specified in the ALD register, into the ND-110PCX. The loading operation then proceeds under control of the MOPC.

ALT+F4: AUTO

This is the Auto-boot flag. It is either "enabled" or "disabled". Besides being part of the System Parameter Definition option, this flag can be changed from one state to the other by selecting ALT0. The bottom line of the display shows the state of the existing setting:

enabled

disabled

AUTO+

AUTO-

ALT+F5: OPCOM

Selecting OPCOM starts the MOPC program without the need to first go into Stop mode. This allows other programs to run while MOPC is used from the console (terminal 1).

One new feature of OPCOM in Butterfly-110 is that input may be lowercase and uppercase (in previous versions of OPCOM, only uppercase was acceptable). The echoed output (on the terminal 1 display) is in uppercase.

Error messages Whilst using the Console option, error messages may originate from:

- the Butterfly Supervisor Program
- SINTRAN or programs running on the ND.

Error messages originating from the Butterfly Supervisor Program arise from a failure to provide OPCOM facility when requested. These are:

Unable to set OPCOM

Unable to set OPCOM for Stop

Unable to set OPCOM for Start

7.6.4.2 Warm Start and Cold Start

These facilities restart the ND side of Butterfly-110.

Why restart?

A restart may be necessary after finishing work using Console (e.g. OPCOM) or Telefix, or if it is suspected that the ND side of Butterfly-110 has been corrupted and needs to be re-initialised and restarted.

Essentially, a Warm Start is a much quicker and less comprehensive restart than a Cold Start. If restarting is necessary, it is therefore usual to do a Warm Start first, and if this is not successful, only then to do a Cold Start.

What happens

In Butterfly-110, both Warm Start and Cold Start initiate the equivalent of a ND master clear (HMCL in the Desk Supervisor Console facility), which initialises all the ND hardware into its starting condition. It then runs the ND self-tests. The essential difference between Warm and Cold Starts is in the way that SINTRAN is then loaded from disk into ND memory.

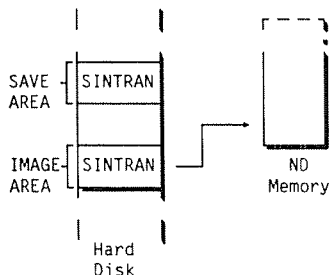


Fig 7.32: SINTRAN Warm Start

SINTRAN is stored on the hard disk in a SAVE AREA and an IMAGE AREA. The SAVE AREA stores the base copy of SINTRAN; this is only ever used to make a new copy of SINTRAN in the IMAGE area, although it may from time to time be patched to update with program modifications. The IMAGE AREA is a copy of the SAVE AREA that has been initialised and set up by the ND HENT-MODE file. This is described in detail in the SINTRAN III System Supervisor Guide (ND-30.003).

In a Warm Start (fig 7.32), the IMAGE AREA copy of SINTRAN is loaded into Butterfly-110's memory, and the LOAD-MODE file is run.

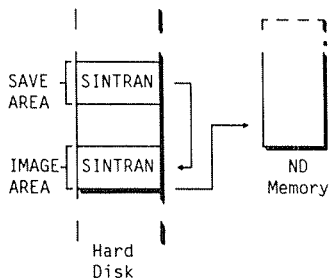


Fig 7.33: SINTRAN Cold Start

In a Cold Start (fig 7.33), the SAVE AREA on disk is copied in order to restore a completely new IMAGE AREA on the disk. This new IMAGE AREA is then loaded into ND-Memory. Following this, Butterfly-110's HENT-MODE file is executed to restore both the ND-Memory copy and IMAGE AREA copy to the correct (initialised and set up) condition.

Note that for Warm and Cold Starts, the Auto-load and Auto-boot settings in the System Parameter Option must be set so as to specify loading SINTRAN from disk.

Alternatives

A Warm Start is the equivalent of what happens in the ND side of Butterfly-110 when it is switched on. This process is described in flowchart form in section 7.6.2.1, and covers the sequence of events from the point where the ND is master-cleared, all PC self-tests and MS-DOS loading having been completed.

If SINTRAN is already running, a Warm or Cold Start can be initiated from the SINTRAN prompt, by using the commands:

@RESTART-SYSTEM

or

@COLD-START

A Cold-Start from the SINTRAN prompt requires that the Hent mode file is run manually afterwards (@MODE HENT-MODE:MODE,,)

Alternatively, if the system is already running the Butterfly Supervisor program, you can select the Warm Start from the Butterfly Supervisor Main Menu. Then, if the Warm Start is unsuccessful, select Cold Start to again attempt to load SINTRAN. If this also fails, check that the PC side is fully functional (see "PC equivalents", below). If it is, repeat the Warm Start, then if necessary the Cold Start. If this fails to load SINTRAN, run diagnostic tests to determine the problem area - see section 7.6.3.

PC equivalents

The PC has equivalents to ND's Warm and Cold Starts, which include starting up the ND side.

- CTRL+ALT+DEL (Warm Start for the PC)

When these keys are all pressed at the same time, the PC holds the ND side in master clear state while it reloads MS-DOS, then runs its PC-ND self-tests, and then it releases ND master clear. If ND program is in memory, the ND will restart from address 20, as if a power-fail has been seen and battery-backup was effective.

- CTRL+ALT+xx (xx is a free chosen key, set up in the PC's CMOS-RAM - see section 7.4.1 "Hard RESET")

This initiates the equivalent of a Butterfly-110 power-up start, but avoids switching the power off then back on.

If the ND keyboard driver is active (normally it is), it has programmed this key to be EXIT (code 6Ah).

Warm Start

On selecting Warm Start in the Supervisor program, the display is cleared and a Warm Start message line on the bottom of the screen indicates how the Warm Start is progressing:

Warm Start: Master Clearing ...

Warm Start: Self Testing ...

Warm Start: Loading SINTRAN ...

and finally

Warm Start: When ready, Press any key to
return to main menu

The normal SINTRAN messages also appear on the screen. Ultimately, the "SINTRAN running" and "system available" messages appear, indicating that SINTRAN is successfully loaded. Then, press any key to return to the Butterfly Supervisor Main Menu.

During the Warm Start, any errors that are detected are reported in the form of error messages on the screen. There are three sources of these messages:

- the Warm Start utility
- the Start-up program used for self-testing
- SINTRAN.

Any error messages from Warm Start appear on the same line as the progress messages.

Cold Start

On selecting Cold Start in the Supervisor program, the display is cleared a Cold Start message line on the bottom of the screen indicates how the Warm Start is progressing:

Cold Start: Master Clearing ...

Cold Start: Copying Save area to Image area ...

Cold Start: Self Testing ...

Cold Start: Preparing new SINTRAN Image ...
(<Esc> to terminate)

If not terminated by the user, the following message is displayed:

Cold Start: When ready, Press any key to
return to main menu

If terminated by the user, the message:

Cold Start: Terminated by user. Press any key
to return to main menu

appears.

The normal SINTRAN messages also appear on the screen. Ultimately, the "SINTRAN running" and "system available" messages appear, indicating that SINTRAN is successfully loaded. Then, press any key to return to the Butterfly Supervisor Main Menu.

During the Cold Start, any errors that are detected are reported in the form of error messages on the screen. There are three sources of these messages:

- the Cold Start utility
- the Start-up program used for self-testing
- SINTRAN.

Any error messages from Cold Start appear on the same line as the progress messages.

As usual, each option has its own Help display.

- **Originate** : Supervisor does the dialling to establish connection to the Telefix Service Centre.
- **Auto-answer**: Remote Service Centre dials the Butterfly modem. In Telefix Auto-answer mode, the modem and Butterfly Telefix program are ready to receive an incoming call from a Telefix Service Centre.
- **No Modem** : There is a direct connection between Butterfly and a Telefix computer - the distance between them is so short (<100m) that modems are unnecessary.
- **Tfx bd rate**: Defines the baud rate setting for the serial modem link between the Butterfly and the Telefix centre.

Originate Connection

Selecting this option allows dialling up a remote Telefix Service Centre directly from the Butterfly keyboard. On entering "originate", the program asks for the telephone number of the remote site (fig 7.36).

Enter the telephone number of the Telefix Service Centre modem for automatic dial-up	EXIT to Telefix Menu HELP for help
--	---------------------------------------

Fig 7.36: Telefix - originate connection

The Help window for this menu provides a prompt as to how the telephone number should be keyed into this display. The only acceptable characters are:

- **T** where Butterfly is connected to a telephone exchange which recognises the number dialled by the pitch of the tone; it generates
- **P** where the telephone exchange recognises the number dialled by the number of pulses (clicks) it generates
- numbers 0 to 9 # and *

- , which introduces a 1-second delay between the preceding number and the next. This caters for the requirement in some telephone exchanges where you have to wait for the next tone before continuing to dial. If necessary, several commas can be strung together to introduce an equivalent number of seconds delay.
- separators () - and SPACE, to punctuate the number and so make it easier to read on the display.

In the following example, the number is made up of an outside-line code to an exchange which recognises tone dialling, then the STD code (in brackets to make it easier to read) and number of the Remote Telefix modem, whose exchange recognises pulse dialling. A 2-second delay is introduced between dialling for the outside line and starting to dial the STD code. The space between the STD code and the final number makes it easier to read:

T9,, P(0234) 56789

Press ENTER to start dialling using the number entered. If the number contains any non-permitted characters, the system gives an audible warning. If this occurs, check that unacceptable character(s) are not included in the number, and correct it as necessary.

The system then tries to establish the telephone line connection. It displays the following progress messages:

Resetting Modem ...

Programming Modem ...

Dialling ... (<Esc> to terminate dial-out)

Connection Established

An error message is displayed if the system is unable to establish the telephone connection. Any such message appears beneath the box area on the screen.

If an error message is displayed, this attempt to establish the connection is terminated. You may then try again, or talk to the Remote Telefix site and perhaps ask them to attempt to establish the connection (with you in Auto-answer mode - see below).

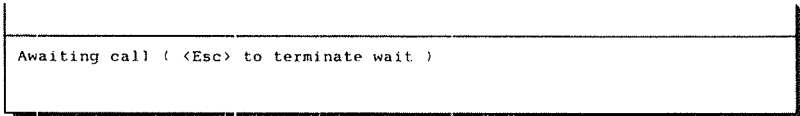
Once the Telefix connection is established, the display is cleared, ready for the Telefix session to proceed. See "Telefix session" below.

Auto-answer

Selecting this option allows the Remote Telefix Centre to start up a Telefix session with Butterfly.

On entering this option, the program displays a progress message for resetting the modem, then another message advising that the modem is being programmed to auto-answer incoming calls.

When the modem is successfully programmed, then the following screen is displayed (fig 7.37).



```
Awaiting call ( <Esc> to terminate wait )
```

Fig 7.37: Telefix - Auto-answer

The system then waits for the remote Telefix Service Centre to call.

When a call from the Telefix Centre arrives, the message:

```
Call arrived, attempting to connect ...
```

is displayed. Then, when the connection is established, the display is cleared, ready for the Telefix session to proceed. See "Telefix session" below.

If the attempt to establish a connection is unsuccessful, or is lost during a Telefix session, an appropriate error message is displayed on the bottom line of the screen. In this situation, contact the remote Telefix Centre to agree what further action to take.

No Modem

This option applies to the situation where a ND computer equipped with Telefix is sufficiently close to the Butterfly to allow a connection without using modems. In practice, the connecting cable should not be longer than 100 metres.

With the direct cable connection in place, the line connection is already established, so on selecting this option the display is cleared, ready for the Telefix session to proceed. See "Telefix session" below.

- Telefix baud rate** This option enables setting of the serial baud rate for the asynchronous communications link between the Butterfly and the Telefix centre. On selecting this option, the existing setting is displayed. The Help window shows the valid range of baud rate settings.
- Telefix session** The Help window associated with a Telefix session is available when in LOCAL. To obtain it (fig 7.38), press ALT+F9. It provides a brief reminder of how SOURCE and LISTEN should be used:

```

-- Help --
LOCAL AND LISTEN OFF: You control
console and Telefix can't listen.
LOCAL AND LISTEN ON: You control
console and Telefix can listen.
REMOTE AND LISTEN OFF: Console is
controlled by Telefix centre.
REMOTE AND LISTEN ON: Console is
controlled by Telefix and you can
listen in.
-- ALT F9 to Exit Help Screen --

```

Fig 7.38: Telefix - LISTEN On/Off

A feature of Telefix is that control of the operations in a Telefix session is exercised from the Butterfly end of the connection, using the SOURCE and LISTEN parameters:

- **SOURCE**
i.e. where input and output to/from the Butterfly originates. This may be LOCAL or REMOTE (fig.7.39):

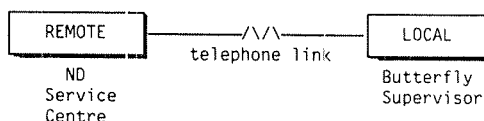


Fig 7.39: Telefix - Local and Remote

- **LISTEN.** This may be set to:
 - ON** (the one that is NOT the Source gets a copy of what the Butterfly outputs)
 - or
 - OFF** (the one that is NOT the Source is not allowed to see what the Butterfly outputs).

These two parameters can only be set up from the Butterfly keyboard, i.e. they cannot be set up remotely. In this way, the Remote location can be prevented from seeing any confidential files which are stored in the Butterfly machine.

After establishing a Telefix line connection, these two parameters automatically come up as REMOTE and ON (i.e. the ND Service Centre has control of the Telefix session).

At any point during the session, these settings Supervisor may be changed. Of course, this should not be done while the Remote location is transferring data or a program down the line to the Butterfly, otherwise the transfer will be incomplete.

If for any reason the Telefix connection is lost when in the REMOTE state, the message

Error: Connection lost. Returning to LOCAL - Listen OFF

appears. You must then contact the Remote Telefix Centre and agree how to continue.

CHAPTER 8
PARTS DATA

TABLE OF CONTENTS

Section	Page
8.1	SYSTEM UNIT 8-3
8.1.1	The PC 8-3
8.1.2	ND-110PCX CPU board 3401 8-4
8.1.3	ND-110PCX Local Bus board 3402 8-8
8.1.4	ND-110PCX Installation PROM board 3404 8-13
8.2	DISPLAY UNIT 8-13
8.3	KEYBOARD 8-13
8.4	MOUSE 8-13

8.1 SYSTEM UNIT

8.1.1 The PC

Parts data on the standard PC-AT System Unit is provided in the relevant supplier's manuals. These parts comprise the following items:

System Unit

- Ericsson PC-AT91
 - System board PMSYM 1195601
 - power supply PE 1063
- Enhanced graphics adaptor card
- Disk controller card
- Floppy disk
- Hard disk
- QuadPort communications controller

Display unit

Butterfly keyboard

Mouse

Certain components in the PC parts of Butterfly-110 may be sourced from separate suppliers. These include:

- Floppy Disk
- Hard Disk
- Enhanced Graphics Adaptor (EGA) card
- Disk Controller board
- Display unit
- Mouse.

8.1.2 ND-110PCX CPU board

Circuit schematic number : 3401. Assembled board part number : 324051.

Section Name & Compt Type	Pins	Qty	Spec	Ident	ND Part No
------------------------------	------	-----	------	-------	------------

Cycle Control 1

74F374	20	1		U 1	500751
PAL16R6 B	20	1	socketed (skt=500335)	U 2	
PAL16R6 A	20	1	socketed	U 3	
PAL16L8 A	20	1	socketed	U 4	
PAL16L8 A	20	1	socketed	U 5	
PAL16L8 A	20	1	socketed	U 6	
PAL16R4 A	20	1	socketed	U 7	
PAL16R4 A	20	1	socketed	U 8	
PAL16R4 A	20	1	socketed	U 9	
PAL16R4 A	20	1	socketed	U 10	
74F10	14	1		U 11	500706
74F00	14	1		U 12	500700
74F04	14	1		U 13	500704
74F02	14	1		U 14	500702
74F08	14	1		U 15	500708
74F20	14	1		U 16	500810
74S37	14	1		U 17	500377

IBD & Comm Decode

PAL16L8 B	20	1	socketed (skt=500335)	U 18	
PAL16L8 A2	20	1	socketed	U 19	
PAL16R4 A2	20	1	socketed	U 20	
PAL16R8 A2	20	1	socketed	U 21	
PAL16R8 A2	20	1	socketed	U 22	
PAL16R8 A2	20	1	socketed	U 23	
PAL16R8 A2	20	1	socketed	U 24	
74LS00	14	1		U 25	
74F74	14	1		U 26	500720
74F32	14	1		U 27	500713
74LS32	14	1		U 28	500249

Section Name & Compt Type	Pins	Qty	Spec	Ident	ND Part No
------------------------------	------	-----	------	-------	------------

Control Store

74F533	20	1		U 29	500760
74F533	20	1		U 30	500760
27C64 150nS	28	1	socketed	U 31	
27C64 150nS	28	1	socketed	U 32	
27C64 150nS	28	1	socketed	U 33	
27C64 150nS	28	1	socketed	U 34	
27C64 150nS	28	1	socketed	U 35	
27C64 150nS	28	1	socketed	U 36	
27C64 150nS	28	1	socketed	U 37	
27C64 150nS	28	1	socketed	U 38	

Memory Management System

74LS244	20	1		U 39	500288
74LS244	20	1		U 40	
74LS244	20	1		U 41	
74LS245	20	1		U 42	500289
74LS245	20	1		U 43	
74LS245	20	1		U 44	
74LS245	20	1		U 45	
74LS374	20	1		U 46	500293
74LS374	20	1		U 47	
IMS1421S-40	18	1		U 48	500491
IMS1421S-40	18	1		U 49	
IMS1421S-40	18	1	or 1423S-35	U 50	
IMS1421S-40	18	1		U 51	
IMS1421S-40	18	1	or AM 2169-40DC	U 52	
IMS1421S-40	18	1		U 53	
IMS1421S-40	18	1		U 54	
IMS1421S-40	18	1		U 55	
33 μ F 10V tant	8	4	0.2" +20%	C 3-6	503106
100 nF 50V cer	136	68	0.1" +80-20%	C 7-74	

Gate Arrays

MIC	120	1	socketed (500848)	U 56	516102
MAC	120	1	socketed (500848)	U 57	516103
ALU	144	1	socketed (500858)	U 58	516101

Address Latches

74F374	20	1		U 59	500751
74F374	20	1		U 60	
AM29821	24	1		U 61	500863

Section Name & Compt Type	Pins	Qty	Spec	Ident	ND Part No
------------------------------	------	-----	------	-------	------------

Register File

AM29841	24	1		U 62	500843
2149 55 ns	18	1		U 63	500491
2149 55 ns	18	1		U 64	
2149 55 ns	18	1		U 65	
2149 55 ns	18	1		U 66	

IDB Transceivers

74F245	20	1		U 67	500743
74F245	20	1		U 68	
74F245	20	1		U 69	
74F245	20	1		U 70	

ALU Jump

74F244	20	1		U 71	500742
74F244	20	1		U 72	
9x1k SIL 2%	10	1	parallel 2%	UR 1	

Print Status / Installation No.

74LS240	20	1		U 173	500280
9x10k SIL 2%	10	1	parallel 2%	UR 2	
2x8 pin conn.	16	1	0.1"	J 2	

Miscellaneous

74F04	14	1		U 76	500704
74F32	14	1		U 77	500713
74F00	14	1		U 78	500700
74LS273	20	1		U 79	500290
74ALS74	14	1		U 80	500620
74F08	14	1		U 81	500705
220R TR3 2%	2	1	0.3"	R 4	
330R TR3 2%	2	1	0.3"	R 5	

Console Interface

SCN2661 A	28	1		U 75	500884
-----------	----	---	--	------	--------

Section Name & Compt Type	Pins	Qty	Sspec	Ident	ND Part No
------------------------------	------	-----	-------	-------	------------

Task - LB Interface

74LS374	20	1		U 125	500293
74LS374	20	1		U 126	
74LS374	20	1		U 127	
74LS646	24	1		U 128	500605
74LS646	24	1		U 129	

Parity Error and Status

74LS374	20	1		U 161	500293
74LS374	20	1		U 162	
74LS374	20	1		U 163	

Miscellaneous Connectors

Note: To be mounted on solder side of board.

2x17pin conn.	34	0.1"	J 10
2x15pin conn.	30	0.1"	J 11
2x17pin conn.	34	0.1"	J 12
2x15pin conn.	30	0.1"	J 13

8.1.3 ND-110PCX Local Bus board 3402

Circuit schematic number : 3402. Assembled board part number : 324052.

Section Name & Compt. Type	Pins	Qty	Spec	Ident

<u>LB - IOB Interface</u>				
74ALS374	20	1		U 101 500664
74ALS374	20	1		U 102
74ALS374	20	1		U 103
74LS646	24	1		U 104 500605
74LS646	24	1		U 105
PAL16L8 A	20	1	socketed	U 106
74LS32	14	1		U 112 500249
74LS08	14	1		U 113 500246
74LS04	14	1		U 114 500270
74LS125A	14	1		U 115 500344
74LS32	14	1		U 117 500249
74ALS74	14	1		U 118 500620
74LS51	14	1		U 119 500250
9x1k SIL Pullup	10	1	parallel 2%	UR101

<u>HDLC Interface</u>				
AMZ8530A	40	1	CHANGED	U 108
AM9516	48	1		U 109 500886
26LS31	16	1		U 110 500296
26LS32	16	1		U 111 500297
74ALS74	14	1		U 120 500620
74ALS74	14	1		U 121
7x10k SIL	8	1	parallel 2%	UR109
1000pF 63V cerm	2	1	0.1" <+10%	C 179
1000pF 63V cerm	2	1	0.1" <+10%	C 180
1000pF 63V cerm	2	1	0.1" <+10%	C 181
DS14C88	14	1		U 195
DS14C89	14	1		U 196
DS14C89	14	1		U 197
74LS367	16	1		U 198
25 Pin D-conn.	25	1	rt L PCB mtg	J 104

Section Name & Compt Type	Pins	Qty	Spec	Ident	ND Part No
------------------------------	------	-----	------	-------	------------

Static Prom

NMC9817A 200ns	28	1	socketed or AM2817A-20 or MK48Z02B-20+SMARTSKT	BU122	500887
27C64 150ns	28	1		U 123	
27C64 150ns	28	1		U 124	

PC-LB Interface

74LS646	24	1		U 130	500605
74LS646	24	1		U 131	
74LS273	20	1		U 132	500290
74LS374	20	1		U 133	500664
74LS245	20	1		U 134	500289
74LS244	20	1		U 135	500288
74LS244	20	1		U 136	
74LS244	20	1		U 137	
PAL16L8 A2	20	1	socketed	U 138	
PAL16L8 A2	20	1	socketed	U 139	
74ALS74	14	1		U 140	500620
74LS32	14	1		U 141	500249
74LS04	14	1		U 142	500270
74LS02	14	1		U 143	500207
74LS125A	14	1		U 144	500344
74LS279A	16	1		U 145	
74LS08	14	1		U 199	500246
74LS138	16	1		U 200	500276
74LS374	20	1		U 201	500664
74LS32	14	1		U 202	500249
74LS32	14	1		U 203	
74LS374	20	1		U 204	500664
7X10k SIL	10	1	parallel 2%	UR110	
1NF 50V ceram	2	1	0.1" +20%	C 184	
31x2 Gold fingers				J 101	

Bus Control 2

AM29822	24	1		U 146	500606
AM29822	24	1		U 147	
PAL16R6 B	20	1	socketed	U 148	
PAL16L8 A	20	1	socketed	U 149	
74F02	14	1		U 150	500702
74F08	14	1		U 151	500705
74F109	16	1		U 152	500723

Section Name & Compt Type	Pins	Qty	Spec	Ident	ND Part No
------------------------------	------	-----	------	-------	------------

DRAM Memory and Parity

AM29828	24	1		U 154	500809
AM29828	24	1		U 155	
74F280	14	1		U 156	500747
74F280	14	1		U 157	
74LS245	20	1		U 158	500289
74LS245	20	1		U 159	
74LS374	20	1		U 160	500664
74F 04	14	1		U 164	500704
HM561003-12	33	1	Hiatchi	U 165	516034
HM561003-12	33	1	or Fujitsu MB85227-12	U 166	
HM561003-12	33	1	or OKI MSC41256YS9-12	U 167	
HM561003-12	33	1	either pinned or not.	U 168	
HM561003-12	33	1	If pinned use MOLEX	U 169	
HM561003-12	33	1	socket type 78810	U 170	
HM561003-12	33	1	(for 2 strips).	U 171	
HM561003-12	33	1		U 172	
4x33R SIL Discr	8	1	individual 2%	UR102	
4x33R SIL Discr	8	1	individual 2%	UR103	
4x33R SIL Discr	8	1	individual 2%	UR104	

Memory Control

HY5030-150R	14	1	or DDU7J-150	U 173	500910
74S37	14	1		U 174	
74ALS1008A	14	1		U 175	500665
74F 00	14	1		U 176	500700
74F 32	14	1		U 177	500713
PAL16L8B	20	1	socketed	U 205	
4x33R SIL Discr	8	1	individual 2%	UR105	

IOR / IOC

741S273	20	1		U 178	500290
AM29827	24	1		U 179	500889
LED Yellow	2	1	0.1"	L 101	502060
LED Green	2	1	0.1"	L 102	502058
LED Red	2	1	0.1"	L 103	502034
7x220R SIL Pullup	8	1	parallel 2%	UR107	

Section Name & Compt Type	Pins	Qty	Spec	Ident	ND Part No
------------------------------	------	-----	------	-------	------------

Oscillator

K1100A	14	1	39.3216MHz	X 101	519035
74ALS74	14	1		U 180	500620
74HCT132	14	1		U 181	500682
74HCT4040	14	1		U 182	500683
74S37	14	1		U 194	500377
7x10k SIL	8	1	parallel 2%	UR108	

Real-Time Clock

IQKF-26	2	1	32.768 kHz	X 1	519182
22pF 50V ceram	2	1	0.1" +80/-20%	C 182	
2-22 pF Var	2	1	Trimmer RS	C 183	
MM5368	8	1		U 183	500890
74ALS74	14	1		U 184	500654

Address Decode

74F32	14	1		U 185	500713
PAL20S10	24	1	socketed	U 186	
74F158	16	1		U 187	500781
4x33R SIL	8	1	individual 2%	UR106	

Priority Interrupt

74ALS74	14	1		U 188	500620
AM2913	20	1		U 190	500304
AM2913	20	1		U 191	500304

Powerfail / Clear

74HCT02	14	1		U 116	
74HCT10	14	1		U 153	500680
74HCT14	14	1		U 192	500681
180R TR3	2	1	0.3" 2%	R 101	
1k0 TR3	2	1	0.3" 2%	R 102	
2.7V zener	2	1		D 103	

Section Name & Compt Type	Pins	Qty	Spec	Ident	ND Part No
------------------------------	------	-----	------	-------	------------

Console Interface

75150		8	1		U 193	500240
2k2	TR4	2	1	0.4" 2%	R 103	
5k6	TR3	2	1	0.3" 2%	R 104	
2k2	TR4	2	1	0.4" 2%	R 105	
2k2	TR4	2	1	0.4" 2%	R 106	
1k0	TR3	2	1	0.3" 2%	R 107	
2k2	TR4	2	1	0.4" 2%	R 108	
2k2	TR4	2	1	0.4" 2%	R 109	
220R	TR4	2	1	0.4" 2%	R 110	
1k	TR3	2	1	0.3" 2%	R 111	
10k	TR3	2	1	0.3" 2%	R 112	
1N4148		2	1		D 101	502002
1N4148		2	1		D 102	502002
10nF ceram 50V		2	1	0.1" +20%	C 159	
1000pF ceram 50V		2	1	0.1" +20%	C 160	
10nF ceram 50V		2	1	0.1" +20%	C 161	
2x15 pin conn.		30	1	0.1"	J 103	

Miscellaneous Connectors

Note: To be mounted on solder side of board.

2x17skt	34		0.1"	J 110
2x15skt	30		0.1"	J 111
2x17skt	34		0.1"	J 112
2x15skt	30		0.1"	J 113

Power Supplies:

Main - Most of the board, only powered when operating.

Standby - Powered when computer down, for refresh of main memory.

Amp Spades		2	1/4" spade	J 106
Amp Spades		2	1/4" spade	J 107
33 μF 10V tant	4	2	0.2" +20%	C 101-102
33 μF 10V tant	8	4	0.2" +20%	C 103-106
33 μF 10V tant	2	1	0.2" +20%	C 162
3.3 μF 25V tant	4	2	0.2" +20%	C 157-158
100 nF ceram	20	10	0.1" +20%	C 107-116
100 nF ceram	80	40	0.1" +20%	C 117-156
100 nF ceram	80	40	0.1" +20%	C 163-178

8.1.4 ND-110PCX Installation PROM board 3404

Section Name & Compt Type	Pins	Qty	Spec	Ident	ND Part No
TBP18S30 SLOW	16	1	up to 5 more PROM fitted in sockets	U 201	
2x8skt	16	5	DIL socket for PROM	} U 202 ↓ U 206	

8.2 DISPLAY UNIT

There are two types of Display Unit:

- monochrome
- colour.

For parts information on either of these units, refer to the supplier's manuals.

8.3 KEYBOARD

Parts data to be added.

8.4 MOUSE

For parts information on this unit, refer to the supplier's manuals.

APPENDIX A
LOGIC DIAGRAMS

TABLE OF CONTENTS

3401 sheets 1, 2, 3, 4

ND-110PCX CPU

3402 sheets 1, 2, 3, 4

ND-110PCX LOCAL BUS

APPENDIX B
SIGNALS LIST

INTRODUCTION

The following pages present a list of the signal names used in the Butterfly ND-110PCX computer logic schematics:

- drawing number 3401: Butterfly ND-110PCX CPU,
- drawing number 3402: Butterfly ND-110PCX Local Bus.

Notation

The following notation applies:

- 1 Signals are listed in alphabetical order, with numeric characters appearing before alphabetics.

Any signal preceded with an asterisk (*) is a direct output from Control Store, before being "pipelined" into a latch.
- 2 The ACTIVE column indicates the active logic state of the signal:
 - 0 = low, <0.6V relative to system logical ground
 - 1 = high, >2.7V relative to system logical ground
 - + & - refer to serial interface lines.

Where two signal names are listed and have different logical active states, they should not automatically be assumed to be logical inversions of each other.

- 3 The DESCRIPTION entry explains the initials in the name, and where appropriate adds information regarding the signal function and context.
- 4 The SOURCE identifies the origin of the signal, in terms of where it is written on the logic schematics (not the source pin position of the chip). It is given in a short-form style as follows:

drawing number / page number / grid reference on page

E.g. signal TERM appears on drawing 3401
page 3
grid area G4

so source is entered as 1/3/G4.

<u>MNEMONIC</u>	<u>ACTIVE</u>	<u>DESCRIPTION</u>	<u>SOURCE</u>
*A(0-12)	0	unlatched μ code address bits 0-12	1/1/D2
AA(0-3)	1	A operand bits 0-3	1/1/D3
ACOND	0	ALU condition testing	1/1/D3
*ALUI(0-8)	1	ALU instruction bits (μ code bits 55-63)	1/2/G1
μ ALUI8	0	ALU instruction bits (μ code bits 55-63)	1/2/F2
*ALUM(0-1)	1	ALU mode bits 0-1 (μ code bits 44-45)	1/2/G2
APR	1	address present on local bus	2/2/C3
ALEX	0	PC address lines enable (memory cycle)	2/1/B4
ALEX	1	PC address lines enable (memory cycle)	2/1/C4
BA(0-3)	1	ALU B-operand bits 0-3	1/1/D3
BATNOK	1	battery not OK	2/4/B4
BDEST	1	ALU B address is destination	1/1/B2
BDRY	0	Bus-data ready	2/2/A5
*BIT(0-63)	0/1	μ code bits 0-63	1/2/A,B
*BIT(0-15)	1	μ code combined data field bits 0-15	1/2/G4
*BIT20	1	μ code bit 20	1/2/G4
BIT20	1	latched μ code bit 20	1/2/C3
BOSC	1	buffered OSC	2/4/B1
BPERR	0	bus memory parity error	2/2/G2
BRCLK	1	baud rate clock	2/4/B2
BRK	1	cycle break, conditional or trap	1/2/D4
BUSRDY	1	bus ready with memory address present	2/2/C3
BUSRDY	0	bus ready with memory address present	2/2/C3
C	+	HDLC TRXC buffer lines	2/3/A1
C	-	HDLC TRXC buffer lines	2/3/A2
*CA(0-9)	1	cache address bits 0-9	1/3/B3
CA(0-9)	1	latched cache address bits 0-9	1/3/C5
CACK	0	CPU acknowledge (CPU is granted I/O bus)	2/3/E4
CACLK	0	cache address clock	1/2/E5
CACLK	1	cache address clock	1/2/G4
CACT	0	CPU activity on bus	2/2/B5
CACT	1	CPU activity on bus	2/2/B5
CAS(0-3)	0	column address strobe bits 0-3 (mem addr)	2/2/D4
CBRK	0	conditional break in macroinstructions	1/2/D4
CBRK(1-2)	0	CBRK1 ored with CBRK2 to form CBRK	1/2/D4
CC(0-3)	0	nanocycle counter bits 0-3 (grey-coded)	1/2/E3
CC(2-3)	1	nanocycle counter bits 0-3 (grey coded)	1/2/E5
CD(0-15)	1	Cache data bits 0-15	1/1/B1
CFETCH	1	conditional fetch to MIC	1/2/D4
*CINSEL(0-1)	1	ALU carry-in select: μ code bits 46-47	1/2/G2
CLEAR	0	power is up OK (power clear)	2/4/B4
CLFF	0	clear ALU function flip-flops	1/2/C1
CLIRQ	0	clear interrupt request	1/2/C1
CLK1/CLK2		microcycle clock (one per microinstruction)	1/2/F4
CLOSE	0	close the PC write to NVRAM gate	2/1/C4

<u>MNEMONIC</u>	<u>ACTIVE</u>	<u>DESCRIPTION</u>	<u>SOURCE</u>
CLRHD1	0	DMA controller interrupt acknowledge	1/2/C2
CLRHD2	0	HDLC controller interrupt acknowledge	1/2/C2
CLRTI	0	clear real time interrupt	1/2/C2
COMINT	0	console interrupt (Tx or Rx)	1/1/B4
*COMM(0-4)	1	command bits 0-4 (μ code bits 32-36)	1/2/G3
COMM(0-4)	1	latched command bits 0-4	1/2/C3
COMSL	0	command slow microcycle (if not cache hit)	1/2/E2
CRQ	0	CPU request for bus	2/2/E4
CRQI	?	CPU request for bus	2/4/E3
CRY	1	ALU carry status bit	1/1/B2
CTS		HDLC CTS line	2/3/A4
CWRITE	0	CPU write to bus	2/2/E4
CWRITE	1	CPU write to bus	2/2/F5
D170	0	delay 120-165ns into bus cycle	2/2/C3
DACT	0	delayed CPU active on bus	2/2/D5
DCD		HDLC DCD line	2/3/A4
DDACT	0	delayed CPU doubly-active on bus	2/2/D5
DDRY	0	delayed DRY	2/2/B4
DDSTART	0	double-delayed start of bus cycle (DSTART)	2/2/D5
*DELAY	1	μ code bit 21	1/2/G4
DELOUT	0	memory cycle delay out (135ns)	2/2/C4
*DLY(0-1)	0	μ code delay bits 26-27 (not used)	1/2/G3
DIR		direction on current loop (console I/F)	2/4/G4
DISWHI	0	disable write for PC high byte	2/1/B4, 2/3/G3
DISWLO	0	disable write for PC low byte	2/1/B4, 2/3/G4
DLBTIO	1	direction from local bus to I/O bus	2/3/B4
DLBTPC	1	direction from local bus to PC	2/1/B4
DLBTORB	1	direction local bus to RASK internal bus	2/2/F5
DLY(0-1)	0	adds 52ns to the microcycle time	1/2/E3
DMAACK	1	DMA bus acknowledge (from DMA controller)	2/3/D4
DMAALE	0	DMA address latches enable	2/3/E5
DMAALE	1	DMA address latches enable	2/3/D2
DMACS	0	DMA chip select	2/3/C4
DMAWAIT	0	DMA wait	2/3/D3
DMINH	0	delayed memory inhibit	2/2/B4
DMOR	1	delayed memory out of range	2/2/D5
DOUBLE	1	extended mode memory addressing	1/1/B2
DPRIX	0	double precision (from MAC)	1/3/B3
DRAM	0	D-RAM selected	2/2/A2
DSR		HDLC DSR line	2/3/A4
DSR		console UART DSR line	2/4/G4
DSTART	1	delayed start of bus cycle	2/2/B4
DSTOP	1	delayed stop of bus cycle	1/2/C1
DTR		HDLC DTR line	2/3/A2
DVACC	0	delayed virtual access to memory	1/2/E2
DZD	1	double zero detect	1/2/F2
DZL	0	double zero latched	1/2/F2

<u>MNEMONIC</u>	<u>ACTIVE</u>	<u>DESCRIPTION</u>	<u>SOURCE</u>
*ECOND	0	enable conditional branching	1/2/G3
ECREQ	1	enable CPU request for local bus	1/2/D4
EDO	0	enable data out (ALU internal)	1/2/F1
EDOFF	0	enable extra MS-DOS memory offset onto LBA	2/1/D3
EGOFF	0	enable PC general offset register onto LBA	2/1/D3
EHADR	1	enable high address (memory timing)	2/2/B4
EIOBHI	0	enable I/O data bus transceivers high byte	2/3/B5
EIOBLO	0	enable I/O data bus transceivers low byte	2/3/B5
EIOR	0	enable input/output status register	1/2/F1
EIPL	0	enable input for page map lower byte	1/3/B1
EIPU	0	enable input for page map upper byte	1/3/B2
EIPUR	0	enable input to page map upper byte	1/3/B1
EMAP	0	enable map cycle (instr fetch or MAP instr)	1/2/C5
EMCL	0	enable master clear	2/4/E3
EMD	0	en data bet. local bus and cache data bus	1/2/F4
ENCD	0	enable cache data	1/2/F1
ENDOS	1	enable MS-DOS memory (PC extra 128 kbytes)	2/1/B5
ENDST	0	enable ND status to PC	2/1/D3
ENWRT	1	enable PC write to ND memory	2/1/F1
EOFF	0	enable general PC offset value onto PCD	2/1/D3
EORF	0	nanocycle is "D" to "F" (timing control)	1/2/E5
EPAN	0	enable panel type interrupts	1/2/F2
EPCHI	0	enable to PC the high byte of the local bus	2/1/B4
EPCI	0	enable peripheral comms interface (console)	1/2/C2
EPCL0	0	enable to PC the low byte of the local bus	2/1/B4
EPEA	0	enable parity error address register	1/2/F1
EPES	0	enable parity error status register	1/2/F1
EPGS	0	enable paging status register	1/2/F1
EPICV	0	enable priority interrupt controller vector	1/2/F2
EPMAP	0	enable map paging	1/3/B1
EPT	0	enable memory protect table	1/3/B1
EPTI	0	enter memory protect table information	1/3/B2
ERF	0	enable register file to IDB	1/2/G2
ERR	1	error (memory parity)	2/2/G2
ESTOF	0	enable IDB to FIDB (slow to fast)	1/3/B1
ETRP1	0	enable trap address vector 1	1/3/G4
ETRP2	0	enable trap address vector 2	1/3/G4
ETRP3	0	enable trap address vector 3: panel/int rq	1/3/G4
EWCA	0	enable write control store address	1/1/C3
EXTOSC	1	external oscillator as basic clock	2/4/B1
F11		ALU F register bit 11	1/1/B1
F15	1	ALU F register bit 15 (2's compl sign bit)	1/1/B2
F=0	1	ALU F register bit 0	1/1/B2
FETCH	0	macroinstruction fetch cycle	1/2/E2
FETCH	1	macroinstruction fetch cycle	1/2/D3
FIDB(0-15)	1	fast internal data bus bits 0-15	1/1/B1
FORM	0	signals a fetch in cycle control	1/2/C5

<u>MNEMONIC</u>	<u>ACTIVE</u>	<u>DESCRIPTION</u>	<u>SOURCE</u>
GOOFF	0	PC go offset address (64k block)	2/1/B3
GOST	0	PC go status	2/1/B3
GND		logical ground for console interface	2/4/G4
HDINT1	0	DMA controller interrupt	2/3/D1
HDINT2	0	HDLC interrupt	2/3/D1
HOLDRT	0	hold real time interrupts (console)	2/4/E5
I	+	HDLC CTS buffer lines	2/3/A3
I	-	HDLC CTS buffer lines	2/3/A3
I1	+	console receive data	2/4/G2
I2	-	console receive data	2/4/G2
ID5	0	interrupt detect 5: ALU Z indicator	1/1/F3
ID8	0	interrupt detect 8: memory parity error	2/4/E1
ID9	0	interrupt detect 9: memory out of range	2/4/E1
ID10	0	interrupt detect 10: power fail	2/4/B3
IDB(0-15)		internal data bus bits 0-15	
*IDBS(0-4)	1	IDB-source bits 0-4 (from µcode)	1/2/G2
IEPTON	1	interrupt enable PC to ND	1/1/F3
IE5	0	interrupt enable 5: ALU Z indicator	1/4/E3
IE8	1	interrupt enable 8: memory parity error	2/4/E3
IE9	1	interrupt enable 9: memory out of range	2/4/E3
IE10	1	interrupt enable 10: power fail	2/4/E3
IND	0	indirect memory access	1/2/D3
INOSCOFF	0	internal oscillator off	2/4/A1
INTRQ	0	interrupt request flip-flop status	1/2/E1
INVAL	0	invalid memory address	2/2/A2
INVALID	0	invalid memory address	2/2/B3
IOA bus	1	I/O address bus	2/3/C1
IOB(0-15)	1	I/O data bus bits 0-15	2/3/D1
IOBGNT	0	I/O bus grant	2/2/C4
IOBRD	0	I/O bus read	2/3/E5
IOBRD	1	I/O bus read	2/3/B5
IOCHCHK	0	I/O channel check	2/1/D4
IOCHRDY	0	I/O channel ready (PC)	2/1/C4
IODS	0	I/O data select (to DMA controller)	2/3/B4
IOMM	0	I/O bus memory mapped (CPU accesses I/O bus)	2/2/B2
IOREQ	0	I/O request	2/3/C5
IORQ	0	I/O bus request	2/2/B4
IOTIME	0	memory cycle I/O time	2/2/D3
IRQ	1	interrupt request	2/4/F1
IRQ5	1	interrupt request 5 (to the PC)	2/1/D2
JMP	0	used in ALU	1/2/C4
JMPA	0	update ALU during jump instr (CA/LA to FIDB)	1/2/F1

<u>MNEMONIC</u>	<u>ACTIVE</u>	<u>DESCRIPTION</u>	<u>SOURCE</u>
*LA(10-25)	1	mem logical address bits (1k page pointer)	1/3/B2
LA(10-25)	1	latched address bits (1k page pointer)	1/3/C3
LAPA	0	memory logical address to physical address	1/3/B1
LBA(0-23)	1	local bus address bits 0-23	
LBD(0-15)	1	local bus data bits 0-15	
LBTIO	0	local bus to I/O bus	2/3/B5
LBW	1	load bus write (D-RAM write data strobe)	1/4/B3
			2/1/B2
			2/3/F1
LCS	0	load control store after reset	1/1/C3
LCZ	0	loop counter zero	1/1/D3
LDDBR	0	load data bus register (in ALU)	1/2/E2
LDEXM	0	load "examine" register in MAC	1/2/C2
LDGPR	0	load GPR (register in ALU)	1/2/D1
LDIRV	0	load instruction register value into MIC	1/2/D2
LDLC	0	load loop counter (ucode cmdnd 17)	1/2/D2
LDPIL	0	load program level	1/2/D3
LIOA	1	latch local bus LBD for IOA bus	2/3/G3
LKEY	0	latch keys register (PC cmdnd byte to ND)	2/1/D3
LOFF	0	latch offset register	2/1/D3
		(PC window to ND address space)	
LOOP	0	loop bit from μ code (enables loop ctr)	1/2/G4
LPCBD	0	latch data local bus to PC	2/1/E2
LRIBD	1	latch local bus to cache data bus	2/2/F5
LST1	1		2/2/C4
LST2	1		2/2/C4
LSHADOW	1	latched shadow memory status	1/3/C3
L μ A(0-12)	1	latched μ code address	1/1/D2
LWCA	0	load write to control store address	1/1/C3
M/IO	0/1	DMA controller M/IO address line	2/3/D3
MACLK	1	microcontrol store address clock	1/2/G4
MADR(0-9)	0	D-RAM address bits 0-9	2/2/D1
MAP	0	map cycle (fetch macroinstruction)	1/2/F4
MCL	0	master clear (SWMCL, PCMCL, or EMCL)	2/4/E4
MCL	1	master clear (SWMCL, PCMCL, or EMCL)	2/4/E4
MCLK	1	map clock (latches logical address to mem)	1/2/G4
MCRQ	1	delayed CRQ	2/2/D5
MDLY(0-1)	0	modified delay 0 & 1	1/2/F3
MDRY	1	mega-delayed data ready	2/2/B4
MDRY	0	mega-delayed data ready	2/2/C5
MEMR	0	memory read by PC	2/1/A4
MEMW	0	memory write by PC	2/1/A5
MEMW	1	memory write by PC	2/1/A4
MINH	1	memory inhibited	2/4/B3
MIOREQ	1	delayed IOREQ	2/2/D5
*MIS(0-1)	1	μ code miscellaneous bits (0 and 1)	1/2/G2
MIS(0-1)	1	latched μ code miscellaneous bits (0 and 1)	1/2/C3
MPCREQ	1	delayed PCREQ	2/2/D5

<u>MNEMONIC</u>	<u>ACTIVE</u>	<u>DESCRIPTION</u>	<u>SOURCE</u>
MR	0	master reset	2/2/B4
MR	1	master reset	2/2/B4
MREQ	0	memory request	1/2/C5
MRFLCK	0	delayed RFCLK	2/2/D5
MSTART	0	latched memory cycle start	2/2/C4
NTOPINT	0	ND to PC interrupt	2/1/D2
NVOPEN	0	NV (static) RAM gate for writing from PC	2/1/B3
NRPTON	0	ND reset PC to ND interrupt	1/2/C1
NSNTOP	0	ND set ND to PC interrupt	1/2/C1
NSPTON	0	ND set PC to ND interrupt	1/2/C1
O1	+	console transmit data	2/4/G3
O2	-	console transmit data	2/4/G3
OPCOM	0	operator communication	2/4/C4
OSC	1	basic clock (internal source 40MHz nominal)	2/4/B1
OVF	1	overflow from ALU on arithmetic operation	1/1/B2
PAN	1	panel interrupt	2/4/F2
PANPAN	1	panel interrupt	1/2/E1
PARERR	0	memory parity error	2/2/E4
PARHI	0	memory parity bit for high byte	2/2/G1
PARHI	1	memory parity bit for high byte	2/2/F1
PARLO	0	memory parity bit for low byte	2/2/G2
PARLO	1	memory parity bit for low byte	2/2/F1
PCA(0-19)	1	PC address bus bits 0-19	2/1/A1
PCA18	0	PC address bus bit 18	2/1/B3
PCAEEN	1	PC address enable	2/1/A3
PCBUF	0	enable ND data connection with PCD bus	2/1/C3
PCD(0-7)	1	PC data bus bits 0-7	2/1/E3
PCFAIL	1	PC commands power fail condition to ND	2/1/B2
PCGNT	0	PC granted use of LBA	2/2/C4
PCGNT	1	PC granted use of LBA	2/2/C4
PCIOR	0	PC input/output read	2/1/A3
PCIOW	0	PC input/output write	2/1/A3
PCLETGO	0	PC releases power fail condition from ND	2/1/B3
PCMCL	0	PC master clear	2/1/B3
PCMEMR	0	PC memory read	2/1/A4
PCMEMW	0	PC memory write	2/1/A4
PCPRES	0	PC present	2/1/A3
PCR(0-2,15)	1	paging control register bits 0,1,2,15	1/3/B4
PCREQ	0	PC request for local bus	2/1/C4
PCRQ	0	PC memory present	2/2/B4
PCSIMF	0	PC simulates power fail	2/1/B2
PHI	1	parity bit for high byte read from D-RAM	2/2/F1

<u>MNEMONIC</u>	<u>ACTIVE</u>	<u>DESCRIPTION</u>	<u>SOURCE</u>
PIL(0-3)	1	priority interrupt level bits 0-3 (current register file level 0-15)	1/1/B3
PLO	1	parity bit for low byte read from DRAM	2/2/F1
PONI	1	paging on indicator	1/1/B2
POWFAIL	1	power fail	2/4/B3
POWSENSE	0	power sense	2/4/A3
PPN(10-25)	1	physical page number bits 10-25	1/3/D2
PRNTOP	0	PC reset ND to PC interrupt	2/1/F1
PROM	0	enable PROM memory area	2/2/B1
PSNTOP	0	PC set ND to PC interrupt	2/1/F2
PSPTON	0	PC set PC to ND interrupt	2/1/F2
PT(0-15)	1	memory protect table bits 0-15	1/3/F2
PTM	1	page table modified (ALU status reg bit 0)	1/1/B2
PTONINT	0	PC to ND interrupt	2/1/D3
PVIOL	1	page violation (page status reg bit 14)	1/3/G1
PXALT	0	indirect read and prefix	1/3/B3
PXM	1	prefix instruction and indirect read	1/3/B3
R	+	HDLC RXD lines	2/3/A2
R	-	HDLC RXD lines	2/3/A2
RAS(0-3)	1	memory row address strobe bits 0-3	2/2/C2
*RASEL(0-1)	1	Register A-operand select bits 0-1	1/2/G2
*RB(0-3)	1	B-operand from µcode bits 16-19	1/2/G4
RB(0-15)	0	ALU 16-bit output to MAC	1/1/B2
*RBSEL(0-1)	1	Register B-operand select bits 0-1	1/2/G2
READ	1	CPU to memory read cycle	1/2/D1
REF	0	refresh (memory)	2/2/C4
REFRQ	0	refresh request	2/2/A4
RESET DRV	1	reset drive (reset command from PC)	2/1/B2
RESRT	1	reset real time interrupts	2/4/E4
RESTR	1	restricted mode (active = no privileged instr allowed)	1/1/C3
RF(0-1)	1	register file address bits 0-1 (bank select)	1/1/D3
RFC	1	refresh clock	2/2/B4
RFCLK	1	refresh clock	2/4/B2
RINR	0	read installation register	1/2/F2
RIOK	0	(internal to trap system PAL)	1/3/G2
RPCI	0	read peripheral control interface (console)	1/2/C2
RRF	0	read register file	1/2/F2
RRQ	1	refresh request	2/2/B4
RSTDRV	0	reset drive (buffered reset from PC)	2/1/B2
RTS		UART RTS line	2/4/G4
RTS		HDLC RTS line	2/3/A2
RUN	1	ND is in RUN mode	2/4/B4
RXC		HDLC RXC line	2/3/A3
RXD		console RXD	2/4/G2
RXD		HDLC RXD	2/3/A3

<u>MNEMONIC</u>	<u>ACTIVE</u>	<u>DESCRIPTION</u>	<u>SOURCE</u>
S	+	HDLC RTXC lines	2/3/A3
S	-	HDLC RTXC lines	2/3/A3
SCACT	0	select CPU active	2/2/C4
SCACT	1	select CPU active	2/2/F5
SCCCE	0	system comms controller (HDLC) chip enable	2/3/C4
SCCRD	0	serial comms controller read	2/3/B5
SCCWR	0	serial comms controller write	2/3/B5
*SCOND	0	set condition for branching	1/2/G3
SECRET	1	secret memory area	1/1/F3
SEMRQ	0	semaphore request	1/2/C1
SGR	1	ALU sign: greater/equal	1/1/B2
SHADOW	1	shadow area of memory (paging tables)	1/3/B3
SHORT	1	command a short microcycle	1/2/C5
SIOC	0	strobe I/O control reg (ucode cmnd 7)	1/2/C2
SLBTIO	1	select local bus (LBD) to I/O bus (IOB)	2/3/B5
SLBTPC	1	select local bus to PC	2/1/E2
SLOWS	0	slow microcycle	1/2/D5
SPEA	0	strobe parity error address register	2/2/E4
SPES	0	strobe parity error status register	2/2/F4
*SST(0-1)	1	ALU status bits 0 & 1 (µcode bits 53-54)	1/2/G2
SSTART	0	start: resets Stop flip-flop (µcode cmnd 13)	1/2/D3
SSTOP	0	stop: sets Stop flip-flop (µcode cmnd 14)	1/2/D3
STAT	0	enable static memory area	2/2/B1
STOC	0	data from slow (IOB) to cache data (CD bus)	1/2/D1
STOF	0	data from slow (IOB) to fast (FIOB)	1/2/F2
STOP	1	ND is in STOP mode	2/4/B3
STRIKE	0	a write to NVRAM from PC is allowed	2/1/B4
SWELO	0	static memory write enable	2/2/D3
SWMCL	0	switch master clear	2/4/A4
SYSREQ	1	select PC(MS-DOS) or ND(SINTRAN) environment	2/1/D1
T	+	HDLC transmit lines	2/3/A1
T	-	HDLC transmit lines	2/3/A1
TCLK	1	trap clock	1/2/F4
TERM	0	terminal nanocycle (last in this microcycle)	1/2/G4
TEST	0	console test line	2/4/C4
TIMEPULS	0	memory row address strobe timing pulse	2/2/B4
TRAALD	0	enable auto-load descriptor	1/2/F1
TRAP	0	trap condition	1/2/F4
TRAP1	0	trap type 1 condition	1/3/G1
TRAP2	0	trap type 2 condition	1/3/G2
TRAP3	0	trap type 3 condition	1/3/G3
*TS3	1	true stack BIT 0)	1/2/G3
*TS4	1	true stack BIT 1) from µcode	1/2/G3
*TS5	1	true sequence BIT 0) bits 28-31	1/2/G3
*TS6	1	true sequence BIT 1)	1/2/G3
TSTART	1	delayed DDSTART	2/2/D5
TXC		HDLC TXC line	2/3/A3

<u>MNEMONIC</u>	<u>ACTIVE</u>	<u>DESCRIPTION</u>	<u>SOURCE</u>
TXD		UART TXD line	2/4/G3
TXD		UART TXD line	1/1/B4
TXD		HDCL TXD line	2/3/A2
T45	1	memory cycle time 45ns	2/2/C3
T60	1	memory cycle time 60ns	2/2/C3
T150	1	memory cycle time 150ns	2/2/C4
UP	0	loop counter count up	1/1/D3
VACC	0	virtual access to memory	1/3/B2
*VECT	0	branch microaddress is vectorized	1/2/G3
VEX	0	latched ucode bit 33	1/2/E2
WAIT1	0	wait for bus - read or write) microcycle	1/2/C5
WAIT2	0	wait for bus -read) control	1/2/C5
WEHI	0	write enable high byte	2/2/D3
WEHI	1	write enable high byte	2/2/D5
WELO	0	write enable low byte	2/2/D3
WELO	1	write enable low byte	2/2/E5
WMAP	0	write to memory map tables	1/3/B2
WP	0	write to P-reg address (abort pre-fetch)	1/1/B2
WRF	0	write to register file	1/2/D3
WRF	1	write to register file	1/2/D3
WRITE	0	CPU to memory write cycle	1/2/E2
WRITE	1	CPU to memory write cycle	1/4/A3
XCLK	1	microcycle clock (1 per microinstruction)	1/2/G3
XFETCH	1	used in ALU	1/2/C4
XOSC	1	buffered OSC	2/4/B1
*XRF3	1	µcode bit 50: extension reg file addr bit 3	1/2/G2
X21	1	HDLC X21 enabled	1/1/F2
Z	1	ALU "Z" indicator	1/1/B2
21OR22	1	PPN bits 21 or 22	1/4/B5

APPENDIX C
PC EXPANSION BUS

LIST OF ILLUSTRATIONS

Figure		Page
C.1	PC Expansion Bus connectors	C-3
C.2	Pin designations and signal names on Expansion Bus 62-way connector	C-4
C.3	Pin designations and signal names on Expansion Bus 36-way connector	C-5

The PC Expansion Bus is also known as the Input/Output (I/O) Channel. It comprises eight slot positions in the PC chassis. These accommodate expansion bus boards. Fig C.1 shows the connectors for these eight slot positions.

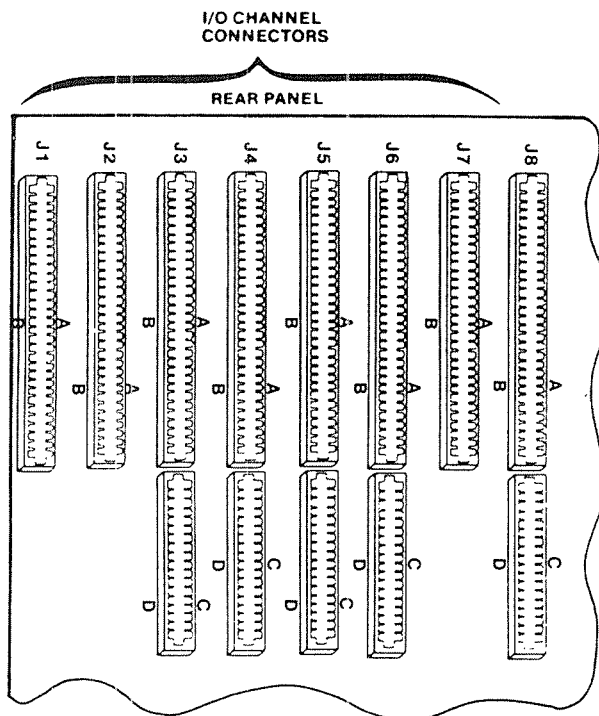


Fig C.1: PC Expansion Bus connectors

All Expansion Bus slot positions are fitted with a 62-way connector. Five positions are also fitted with a 36-way connector. The following pages summarise the signals on these connectors.

I/O Pin	Signal Name	I or O	I/O Pin	Signal Name	I or O
A 1	IOCHCK	I	B 1	GND	ground
A 2	D7	I/O	B 2	RSTDRV	0
A 3	D6	I/O	B 3	+5 Vdc	power
A 4	D5	I/O	B 4	IRQ9	I
A 5	D4	I/O	B 5	-5 Vdc	power
A 6	D3	I/O	B 6	DRQ2	I
A 7	D2	I/O	B 7	-12 Vdc	power
A 8	D1	I/O	B 8	SRDY (OWS)	I
A 9	D0	I/O	B 9	+12 Vdc	power
A 10	IOCHRDY	I	B 10	GND	ground
A 11	AEN	0	B 11	SMEMW	0
A 12	A19	I/O	B 12	SMEMR	0
A 13	A18	I/O	B 13	IOWC	I/O
A 14	A17	I/O	B 14	IORC	I/O
A 15	A16	I/O	B 15	DACK3	0
A 16	A15	I/O	B 16	DRQ3	I
A 17	A14	I/O	B 17	DACK1	0
A 18	A13	I/O	B 18	DRQ1	I
A 19	A12	I/O	B 19	MEMREF	I/O
A 20	A11	I/O	B 20	SYSCLK	0
A 21	A10	I/O	B 21	IRQ7	I
A 22	A9	I/O	B 22	IRQ6	I
A 23	A8	I/O	B 23	IRQ5	I
A 24	A7	I/O	B 24	IRQ4	I
A 25	A6	I/O	B 25	IRQ3	I
A 26	A5	I/O	B 26	DACK2	0
A 27	A4	I/O	B 27	TC	0
A 28	A3	I/O	B 28	BUSALE	0
A 29	A2	I/O	B 29	+5 Vdc	power
A 30	A1	I/O	B 30	840SC	0
A 31	A0	I/O	B 31	GND	ground

Fig C.2: Pin designations and signal names
on Expansion Bus 62-way connector

I/O Pin	Signal Name	I or O
C 1	IOBHE	I/O
C 2	P2A23	I/O
C 3	P2A22	I/O
C 4	P2A21	I/O
C 5	P2A20	I/O
C 6	P2A19	I/O
C 7	P2A18	I/O
C 8	P2A17	I/O
C 9	MEMR	I/O
C 10	MEMW	I/O
C 11	D8	I/O
C 12	D9	I/O
C 13	D10	I/O
C 14	D11	I/O
C 15	D12	I/O
C 16	D13	I/O
C 17	D14	I/O
C 18	D15	I/O

I/O Pin	Signal Name	I or O
D 1	FMEM	I
D 2	FIO	I
D 3	IRQ10	I
D 4	IRQ11	I
D 5	IRQ12	I
D 6	IRQ15	I
D 7	IRQ14	I
D 8	DACK0	O
D 9	DRQ0	I
D 10	DACK5	O
D 11	DRQ5	I
D 12	DACK6	O
D 13	DRQ6	I
D 14	DACK7	O
D 15	DRQ7	I
D 16	+5 Vdc	power
D 17	SECMAS	I
D 18	GND	ground

Fig C.3: Pin designations and signal names
on Expansion Bus 36-way connector

A0-A19	Address bits 0-19 are used to address memory and I/O devices within the system. These 20 address lines, together with P2A17-P2A23, allow access to up to 16Mbyte of memory space. A0-A19 are gated onto the system bus when BUSALE is high. They are latched on the falling edge of BUSALE. Addresses are generated from the microprocessor or the DMA Controller. They may also be driven by other microprocessors or DMA Controllers residing on the Expansion Bus.
P2A17-P2A23	These signals (unlatched) address memory and I/O devices within the system. They give the system addressing capability up to 16Mbyte. These signals are valid when BUSALE is high. They generate memory decodes for 1 wait-state memory cycles. They may be driven by other microprocessors or DMA Controllers residing on the Expansion Bus.
SYSCLK	This is a 6MHz system clock. It is a synchronous microprocessor cycle clock, with a cycle time of 167 ns.
RSTDRV	This is used to reset or initialise system logic, at power-on time or on recovery from a power-fail condition.
D0-D7 D8-D15	These signals provide the data bus bits 0-15 for the microprocessor, memory and I/O devices. D0 is the least significant bit. All 8-bit devices use D0-D7 (on the 62-way connector). D8-D15 are on the 36-way connector. It follows from fig D.1 that connector positions J3-J6 and J8 are best suited to operation for 16-bit devices.
BUSALE	"Bus address latch enable" is used to latch valid addresses and memory decodes from the microprocessor. When used with AEN, it is available to indicate a valid address is present. Microprocessor addresses should then be latched on the trailing edge of BUSALE. This signal is forced high during DMA cycles.
IOCHCK	"I/O channel check" provides the PC with parity (error) information regarding memory or devices on the Expansion Bus. When active, this signal indicates an uncorrectable system error.
IOCHRDY	"I/O channel ready" is sent low (not ready) by a memory or I/O device when it is required to lengthen memory or I/O cycles. A slow device should drive this signal low immediately on detecting its valid address and a Read or Write command. Machine cycles are extended by an integral number of clock cycles (167 ns). This signal should not be held low for longer than 2.5 μ s.

IRQ lines Interrupt requests 3-7, 9-12, 14-15 interrupt the PC. The interrupt requests are prioritised:

```

                IRQ 9      highest
                ↓
                IRQ 12
                IRQ 14
                IRQ 15
                IRQ 3
                ↓
                IRQ 7      lowest

```

The interrupt line must be held high until the microprocessor acknowledges the interrupt request. IRQ 13 is reserved by the PC system board. IRQ 8 is the PC's real-time clock.

IORC	"I/O Read command" instructs an I/O device to drive its data onto the data bus.
IOWC	"I/O Write command" instructs an I/O device to accept the data on the data bus.
SMEMR MEMR	These signals instruct the memory devices to drive data onto the data bus. SMEMR is active only when the memory decode is within the first 1Mbyte of address space. MEMR is active on all memory read cycles.
SMEMW MEMW	These signals instruct the memory devices to store the data present on the data bus. SMEMR is active only when the memory decode is within the first 1Mbyte of address space. MEMR is active on all memory read cycles.
DRQ0-3 DRQ5-7	DMA requests 0-3 and 5-7 are asynchronous requests, used by I/O devices to gain DMA service (or control of the system). DRQ0 has the highest priority, DRQ7 the lowest. A request is generated by sending a DRQ line high. This must be held high until the corresponding DACK (DMA acknowledge) has been issued by the PC.
DACK0-3 DACK5-7	DMA acknowledge 0-3 and 5-7 are issued by the PC to acknowledge receipt of corresponding DMA requests 0-3, 5-7.
AEN	"Address enable" is used to disable the microprocessor and I/O devices from the Expansion Bus, to enable DMA transfers to take place. When it is active, the DMA Controller has control of the address bus, and of the data-bus Read and Write command lines (memory and I/O).
MEMREF	This signal indicates a memory refresh cycle is active.
TC	"Terminal count" is a pulse which occurs when the number of transfers set up for a given DMA channel has taken place.

IOBHE	"Bus high enable" indicates a data transfer is taking place on the upper byte of the data bus (D8-D15). Sixteen-bit devices use IOBHE to condition data bus buffers tied to D8-D15.
SECMAST	This signal is used with a DRQ line, to gain control of the system. A processor or DMA controller on the Expansion Bus may issue a DRQ to a DMA channel in cascade mode. On receiving the DACK response, an I/O controller may take SECMAST low, which allows it to control the system address, data and control lines of the Expansion Bus. Following sending SECMAST low, an I/O controller must wait one system clock period before driving the address and data lines, and two clock periods before issuing a Read or Write command. If this signal is held low for longer than 15 μ s, system memory may be lost because of a lack of refresh.
FMEM (MEMCS16)	"Memory chip select 16" signals the PC that the present data transfer is a 1 wait-state, 16-bit memory cycle. It is derived from the decode of P2A17-P2A23.
FIO (I/OCS16)	"I/O chip select 16" signals the PC that the present data transfer is a 1 wait-state, 16-bit I/O cycle. It is derived from an I/O address decode.
840SC	This is a high speed clock (14.31818 MHz), with a 70 ns period, and a 1:1 mark:space ratio. It is not synchronised to the system clock.
SRDY (OWS)	The "zero wait state" signal tells the microprocessor that it can complete the current bus cycle without inserting any additional wait cycles. To run a memory cycle to a 16-bit device without wait cycles, SRDY is derived from an address decode gated with a Read or Write command. To run a memory cycle to an 8-bit device with a minimum of two wait states, SRDY should be driven active one system clock after the Read or Write command is active, gated with the address decode for the device.

APPENDIX D PAL SPECIFICATIONS

TABLE OF CONTENTS

<u>PAL</u>	<u>Version & Date</u>	<u>Page</u>
PILOT BUTTERFLY PAL 1 : COMMAND DECODE 1	A 3-JUN-86	D-3
PILOT BUTTERFLY PAL 2 : COMMAND DECODE 2	A 2-JUN-86	D-4
PILOT BUTTERFLY PAL 3 : COMMAND DECODE 3	A 2-JUN-86	D-5
PILOT BUTTERFLY PAL 4 : COMMAND DECODE 4	B 7-AUG-86	D-6
PILOT BUTTERFLY PAL 5 : COMMAND DECODE 5	A 2-JUN-86	D-7
PILOT BUTTERFLY PAL 6 : IDB SOURCE DECODE 1	A 2-JUN-86	D-8
PILOT BUTTERFLY PAL 7 : IDB SOURCE DECODE 2	A 2-JUN-86	D-9
PILOT BUTTERFLY PAL 8 : BREAK DETECT	A 2-JUN-86	D-10
PILOT BUTTERFLY PAL 9 : CYCLE TYPE DECODE	A 2-JUN-86	D-11
PILOT BUTTERFLY PAL 10: CYCLE LENGTH DECODE	A 2-JUN-86	D-12
PILOT BUTTERFLY PAL 11: CONDITION CONTROL	A 2-JUN-86	D-13
PILOT BUTTERFLY PAL 12: CLOCK GENERATION	A 3-JUN-86	D-14
PILOT BUTTERFLY PAL 13: MMU CONTROL	B 29-AUG-86	D-15
PILOT BUTTERFLY PAL 14: TRAP 1	A 2-JUN-86	D-16
PILOT BUTTERFLY PAL 15: TRAP 2	D 22-AUG-86	D-17
PILOT BUTTERFLY PAL 16: TRAP 3	A 3-JUN-86	D-18
PILOT BUTTERFLY PAL 17: PC I/F CONTROL	B 17-JUL-86	D-19
PILOT BUTTERFLY PAL 18: PC I/O ADDRESS DECODE	B 17-JUL-86	D-20
PILOT BUTTERFLY PAL 19: LOCAL BUS ADDRESS DECODER	B 17-JUL-86	D-21
PILOT BUTTERFLY PAL 20: LOCAL BUS ADDRESS ARBITER	B 24-JUL-86	D-23
PILOT BUTTERFLY PAL 21: LOCAL BUS CONTROL	D 9-FEB-87	D-26
PILOT BUTTERFLY PAL 22: I/O BUS CONTROL	E 14-FEB-87	D-27
PILOT BUTTERFLY PAL 23: MEMORY CONTROL	C 22-SEP-86	D-28

PAL16R8 (A2)

COMMAND DECODE 1

REG NO 41100

ROGER EDWARDS

PILOT BUTTERFLY PAL 1

(U23)

VERSION A

NORSK DATA LTD., NEWBURY

3-JUN-86

CLK /SSTOP M0 M1 C0 C1 C2 C3 C4 GND

/OE /SEMRQ /NSNTOP /CLIRQ /NSPTON /NRPTON NC /CLFF DSTOP VCC

 $CLFF := /C4 * C3 * C2 * C1 * /C0$ $CLIRQ := /C4 * /C3 * C2 * /C1 * /C0$ $SEMRQ := C4 * /C3 * /C2 * /C1 * C0 * /M1 * M0$
 $+ SEMRQ * /MREQ$ $NRPTON := /C4 * /C3 * C2 * /C1 * C0 * M1 * /M0$ $NSNTOP := /C4 * /C3 * C2 * /C1 * C0 * M1 * M0$ $NSPTON := /C4 * /C3 * C2 * C1 * /C0 * /M1 * /M0$ $/DSTOP := /SSTOP$ DESCRIPTION

Note: CLIRQ decodes as 04 but was also 14 (as per RASK 26-11-85) DSTOP added for trap.

If SEMRQ is to be used properly then the hold on term is needed (taken from RASK) and MREQ needs to be added to the PAL inputs.

PAL16R8 (A2)		ROGER EDWARDS
COMMAND DECODE 2	REG NO 41101	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 2	(U24) VERSION A	2-JUN-86

CLK /EORF M0 M1 C0 C1 C2 C3 C4 GND
 /OE NC /LDEXM /CLRHD2 /SIOC /RPCI /EPCI /CLRHD1 /CLRTI VCC

CLRTI := /C4 * C3 * C2 * /C1 * C0

SIOC := /C4 * /C3 * C2 * C1 * C0;

CLRHD1 := /C4 * /C3 * C2 * /C1 * C0 * /M1 * /M0

CLRHD2 := /C4 * /C3 * C2 * /C1 * C0 * /M1 * M0

EPCI := /C4 * C3 * /C2 * /C1 * C0
 + /C4 * C3 * /C2 * C1 * /C0

RPCI := /C4 * C3 * /C2 * /C1 * C0

LDEXM := C4 * /C3 * /C2 * /C1 * C0 * M1 * M0

PAL16R4 (A2)

COMMAND DECODE 3

REG NO 40700

ROGER EDWARDS

NORSK DATA LTD, NEWBURY

PILOT BUTTERFLY PAL 3

(U20)

VERSION A

2-JUN-86

CLK C0 C1 C2 C3 C4 M1 M0 LSHAD GND

/OE DDACT /ENCD /LDGPR NC READ NC /TRAP /STOC VCC

IF (VCC) STOC = /READ * /ENCD * /TRAP

+ /READ * /DDACT * /ENCD * /TRAP

+ LSHAD * /TRAP

/READ := /C4

+ C4 * /C3 * /C2 * C1 * /C0 * M1 * /M0

+ C4 * /C3 * /C2 * /C1

+ C4 * C3 * C1

+ C4 * C3 * C2 * /C1 * C0

LDGPR := /C4 * /C3 * /C2 * C1 * /C0

+ C4 * /C3 * /C2 * C1 * /C0 * M1 * /M0;

DESCRIPTION

Read inverted 12-11-85 for ESTOF generation.

PAL16R6 (A)		ROGER EDWARDS
COMMAND DECODE 4	REG NO 40800	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 4	(U3) VERSION B	7-AUG-86

CK /LDEXM MO M1 C0 C1 C2 C3 C4 GND
 /OE PONI /DVACC /VEX /COMSL /WRITE /FETCH /LDDBR IDB2 VCC

LDDBR := C4 * C3 * /C2 * /C1
 + C4 * C3 * /C1 * /C0
 + C4 * C2 * C1 * C0 * M1
 + C4 * /C3 * C1 * C0
 + C4 * /C3 * C2
 + C4 * /C3 * C1 * /M1
 + C4 * /C3 * C1 * M0

FETCH := C4 * /C3 * C1 * M1 * M0
 + C4 * /C3 * C1 * C0
 + C4 * /C3 * C2

WRITE := C4 * C3 * /C2 * C1
 + C4 * C3 * C2 * /C1 * C0

COMSL := C4 * /C3 * /C2 * /C1 * /C0
 + /C4 * /C3 * C2 * C1 * /C0 * /M0
 + C4 * C3 * C2 * C1 * C0 * M1

DVACC := /PONI
 + C4 * C3 * C2 * /C1 * M1 * M0 * /VEX
 + C4 * C3 * C2 * C1 * C0 * M1

VEX := LDEXM * IDB2
 + /LDEXM * VEX

DESCRIPTION

MAP command 22,2 removed from COMSL so that it is a long cycle.

keytop	The cap bearing the legend (engraved or two-shot moulded) describing the meaning/function of a key, and covering the keystation movement.
keystation	The physical position of a key in the keyboard.
μ	Abbreviation for micro. In measurements, this is $\times 10^{-6}$.
LAN	Local Area Network: a high-speed (c.10Mbit) local interconnection of computing equipment, sharing information and resources. Typically, the area covered would not exceed a few kilometres.
μ code	Abbreviation for microcode
least significant	In a binary value, this is the lowest-value bit.
LED	Light-Emitting Diode
m	abbreviation for milli, $\times 10^{-3}$.
M	abbreviation for Mega, $\times 10^6$.
MAC	Macroprogram Address Controller: gate array used in ND-110CX and ND-110PCX.
mailbox	The name given to an area of shared memory, through which the ND and PC sides of an OWS-110 and Teamstation communicate to request and control the functions they require of each other.
macroinstruction	A single software instruction written as part of a source language. In the ND-110PCX, each instruction is executed by a sequence of microinstructions.
menu	A screen display which is output from the computer, giving a related set of options to a user, in a user-friendly form. These options will form a part of an interactive task that the user is doing on the computer. The user selects his required option in the displayed menu (the precise way this is done varies from machine to machine) to proceed with this task.
MIC	Microprogram Instruction Controller: gate array used in ND-110CX and ND-110PCX
microcode	The name given to a set of microinstructions.
microinstruction	In ND computers, a 64-bit instruction word which acts directly on the central processing unit to control processor activity. A sequence of several microinstructions are required to execute a macroinstruction.

microprogram	The name given to a set of microinstructions which make up a complete program.
MMS	Memory Management System: that part of the ND computer which handles memory page addressing and memory (paging and ring) protection.
Mouse	A hand-held device connected to the computer, which controls the position and movement of a pointer (cursor) on a display screen. Moving the Mouse in any direction over a flat surface is translated into equivalent movement of the cursor on the display. Typically, up to three pushbuttons are included on the Mouse: these may be assigned functions which correspond to position control keys on the keyboard (e.g. HOME, MARK, RETURN).
monochrome crt	Used to refer to a single colour cathode ray tube (CRT) display. The displayed colour depends on the type of phosphor coating on the screen of the CRT. Various colours are possible, but white, green and orange are the most common.
module	a functionally self-contained body of software code
most significant	In a binary value, this is the highest-value bit.
modem	MODulator-DEModulator: this device modulates an electrical signal into a form suitable for transmission over a telephone line, and demodulates an incoming modulated signal.
monitor	The visual display unit
monitor call	In SINTRAN, the name assigned to a routine which supervises and controls a specific function that may be called upon to execute.
MOPC	In ND computers, <u>M</u> icroprogrammed <u>O</u> perator's <u>C</u> ommunication
MS-DOS	<u>M</u> icro <u>S</u> oft's <u>D</u> isk <u>O</u> perating <u>S</u> ystem - a Microsoft trademark.
multitasking	<p>In purist terms, a mode of operation which provides for concurrent performance, or interleaved execution, of two or more tasks.</p> <p>In OWS-110 and Teamstation, this term is used to refer to the operation of the program in the PC side of the machine which handles both normal "PC" activity and also the ND side's requests for I/O service.</p>
n	abbreviation for nano, $\times 10^{-9}$.
n-key rollover	In keyboards, the feature which caters for very fast typing speeds, whereby when a keyboard operator might press one, or two, or more (up to "n"), code-generating keys before releasing the previous key(s); in this situation, the keyboard still outputs the codes relevant to each of those keys, in the sequence they were keyed.

ND-110PCX	The version of the ND-110 computer which is designed and built specifically to run in the expansion card crate of an IBM-compatible PC.
ND-110CX	Also referred to as "RASK", this is the speeded-up design of the ND-100CX, using three gate arrays (MIC, MAC, ALU).
ND	Abbreviation for <u>Norsk Data</u> .
ND-I/O	Norsk Data Input / Output: refers to the ND HDLC facility in the OWS-110 and Teamstation.
ND Console	The facility provided for operator interaction with an ND computer. In OWS-110 and Teamstation, the ND-110PCX can use the ND-Butterfly terminal via OPCOM, as the ND Console.
network	A system of interconnection which provides for communication between equipment connected to the network (i.e. computers and related terminal devices; all of which are referred to as nodes in the network), for the purpose of transferring information (data) from any one node to any other or all other nodes, and for sharing resources (e.g. a database, output device, etc.).
NV-RAM	Non Volatile Random Access Memory
OPCOM	The ND facility by which an operator can communicate with microprogram. It involves running the "microprogrammed operator's communication" (MOPC) program under SINTRAN. The user interface to this is through an operator's console, which is referred to as "terminal-1". This device may be the ND computer's control panel (if fitted) or other keys+display device. In OWS-110 and Teamstation, the Butterfly-110 terminal itself may be used as "terminal-1".
OWS-10	The machine in the ND-Butterfly range which provides ND's Desk Top Manager (NOTIS-WP, communications to an ND Host computer, and file transfer between MS-DOS and SINTRAN), all on an IBM-PC-compatible machine. It includes two 360k/1.2Mbyte floppy disk drives, the ND/PC Keyboard with Mouse (and a port for connecting a Bar Code Reader), a parallel (Centronics) printer port, black & white monitor driven from a high resolution EGA (640x350 pixel) card, and a serial port for connection to a Host system.
OWS-11	The machine in the ND-Butterfly range which is the same as an OWS-10 except that it has a colour (not black & white) monitor.
OWS-12	The machine in the ND-Butterfly range which is the same as an OWS-11 except that one of its two 360k/1.2Mbyte floppy disk drives is replaced by a 40Mbyte Winchester disk.

OWS-110	The machine in the ND-Butterfly range which has an ND-110PCX computer fitted inside an OWS-12 machine. The ND Desk Top Manager software is replaced by a Butterfly version of SINTRAN III which runs all existing ND applications (NOTIS, SIBAS, UNIQUE, etc.), and which cooperates with the PC's MS-DOS operating system to share all the facilities in the machine.
parity	A check bit added to a set of bits (usually an 8-bit byte) to make the sum of all the bits (including the parity bit) always an ODD or EVEN value.
panel processor	In ND computers, this refers to the processor within the ND Control Panel.
partition (disk)	A sub-division of disk storage area. In OWS-110 and Teamstation, this refers to the 40Mbyte disk being partitioned to allocate 5Mbytes to MS-DOS and the remainder to SINTRAN.
PAL	Programmable Logic Array: an array of logic gates connected together by fusible links, in an integrated circuit chip. The PAL has several logic input and output lines. The fusible links may be "blown" (fused) to break connections as required, leaving an interconnected array of logic gates which implement a set of logic equations relating the logic inputs to the outputs.
parallel interface	An interface which provides for transfer of a group of binary data (e.g. a byte, a word) over a communications interface, in a simultaneous occurrence. This contrasts with the operation of a serial interface.
PC	Personal Computer.
PC-NDI/O	Personal Computer to Norsk Data computer Input/ Output.
PC-AT	Personal Computer - Advanced Technology: a common abbreviation for referring to a machine which is compatible with the IBM PC-AT.
PC expansion bus	The IBM-PC-AT-compatible signal bus and connections available for accommodating "PC-expansion" printed circuit cards into a compatible IBM-PC machine.
pixel	An abbreviation for PICTURE CELL. In computer graphics, it is the smallest element of a display surface that can be independently assigned video on/off, intensity, - and in colour monitors, colour.

pipeline	In ND computers, the method by which successive microinstruction words are read into a "pipeline" buffer register, to ensure that at the end of executing each microinstruction there is minimal processor delay before commencing to execute the next one.
PROM	Programmable Read-Only Memory.
print spooler	The spooling system for a printer.
PSSP	Print Spooler Service Program.
program	The complete set of code needed to perform a specified function or task on a computer.
RASK	A term sometimes used to refer to the ND-110 computer.
RAM	Random Access Memory
read	A "read" operation with memory or a storage device retrieves a copy of information from the addressed area, into the computer
RS-series	The series of standards produced by the EIA (Electrical Industries Association).
RS232C	Specifies the electrical signal characteristics, interface mechanical characteristics, and function description, for selected communication system configurations.
RS422A	Specifies the electrical characteristics of balanced voltage digital interface circuits.
screen	This refers to the visual display area of the monitor.
scan code	In a scan-type keyboard, the keyboard electronics continuously scans the matrix of key positions (keystations). When it finds a key pressed, it identifies that key by its scan position in the matrix - i.e. its scan code. It may send this scan code direct to the computer, or may process or translate it before sending.
service program	A program which performs a self-contained support function, called upon as and when required.
serial interface	An interface which provides for sequential transmission and/or reception of bits (BInary digiTs) of a byte (or word) over a communications link.

server	In a program, a utility which may be called in general support, to perform a specific task that is required from time to time, e.g. a print server.
secret memory	That area of ND-110PCX memory that is reserved for ND-PC communication, HDLC communication, and various other system functions. The PROM and NV-RAM memory areas are included in secret memory.
shared memory	In ND-110PCX, also referred to as Dual-Ported memory: the ND memory can be accessed by the PC side (under MS-DOS) as well as the ND side. It is thus "shared".
SINTRAN	The name given to the Operating System created by Norsk Data for use on ND computers.
spooling	<u>S</u> imultaneous <u>P</u> eripheral <u>O</u> peration <u>O</u> n <u>L</u> ine: the use of a separate buffer area to reduce processing delays when transferring data between the computer and a peripheral device (e.g. a printer).
st-mail	A 4-word "self-test mailbox" area of ND Secret Memory, used by the self-test routines during power-up, and for ND terminal-1 input/output when the Butterfly-110 keyboard & display unit are used as the terminal-1 device, and by ND diagnostic programs.
stand-alone	Operating as a self-contained system, not connected to any other system.
stack	A push-down area of temporary storage which has only one entry and exit point, and where data is handled on a "last-in first-out" basis, so that the next item to be retrieved is the one that was most recently stored
System Unit	The name given to the ND-Butterfly unit which houses the computer and disk systems.
terminal	A common term used to identify a keyboard (for input) and display monitor (for output) as a combined input/output device connected to a computer (directly or via a communications link).
terminal 1	See OPCOM.
Teamstation	The machine in the ND-Butterfly range which comprises an OWS-110 fitted with the 4-port Serial Communications Controller option. Up to four serial devices may be connected to these four ports on the Teamstation, to share its ND-computer facilities.

Telefix	Norsk Data's remote testing facility, which allows for connection, locally or across a telephone line, to a Telefix Service Centre, such that the Centre can load and run programs (including diagnostic tests) on a remote ND computer, and so perform remote fault diagnosis and checking.
trap	In a program, a trap is an unprogrammed conditional jump that is automatically activated by hardware detecting that condition.
typematic	In keyboards, a typematic key is one which repeats its coded output when held pressed. Typically, a typematic key outputs its code straight away when first pressed, then waits for approximately 0.5 seconds, and if still held pressed, then repeatedly outputs that same code at a rate of about 10-15 per second.
UART	Universal Asynchronous Receiver/Transmitter: an LSI (silicon technology term "Large Scale Integration") communications chip which provides the necessary data and control facilities to interface to an RS232 type of communications system.
UE	User Environment. This is the name given to Norsk Data's user interface to its NOTIS product range. It controls access to the computing system (restricting user facilities where required). It also offers various menus to the user (who then selects his required option(s) from these menus); on receiving a selection, it then performs all the necessary operations to implement that option, in such a way as to demand minimal further interaction with the user.
V24	Perhaps the most commonly-recognised of the series of CCITT V-series recommendations which govern data transmission over telephone circuits. It covers hardware interfacing regarding functions controlled, standard pin positions for connections, and electrical characteristics.
voltage loop	Serial communication in which the signals in the line connection are represented electrically as one of two voltage levels. In EIA (Electrical Industries Association) interfaces, these levels are nominally +12V and -12V, although a +/-3V change is the standard minimum. Contrasts with current loop.
VT100	A type of terminal which is recognised throughout the computer industry. It was introduced by the Digital Equipment Corporation (DEC), and is now emulated by many manufacturers.
warm start	In ND computers, this involves resetting the computer and then reloading the "image area" copy of main program, into the computer memory.

WAN	Wide Area Network: local networks some distance apart may be connected together through telephone or other communications networks, to set up a wide area network. ND's WAN system is called "COSMOS".
wand	A common term used to refer to the hand-held "pen" of a bar code reading facility.
watchdog timer	A timer which measures how long a task is allowed to take. It is started at the beginning of the task, and if it times out before the task completes, it signals that the task has taken too long.
Winchester disk	A magnetic disk storage device built with the magnetic disk and heads enclosed inside a sealed dust-free housing. This gives the disk heads and magnetic media increased protection over conventional air-filtration systems, against ingress of dust and other contaminants.
word	In ND computers, a 16-bit binary unit of information. The only exception to this is when referring to microinstructions, where words are 64-bits long.
write	A computer "write" operation involves depositing information into the addressed area of memory or a designated storage device.
X-series (X1-X29)	The series of standards produced by CCITT (Comite Consultatif International de Telephonie et de Telegraphie) which are recommendations to national communications authorities, to establish communications interfaces for user's Data Terminal Equipment (DTE) and modems or other Data Circuit Terminating Equipment (DCE).
X21	This deals with connection of terminals to modems, including automatic calling operations for switched circuits on synchronous Public Data Networks (PDNs). It is designed to replace earlier V24 and V25 interface recommendations.
X25	This recommends a protocol for Packet Switching Networks, supporting attachment of intelligent terminals as well as communication controllers and host computers to the network.

INDEX

ALU	4-26
autoload descriptor	4-47
Bar Code Reader	4-47 F-30
BIOS	3-11 6-62 6-69
BSM	3-33 6-68
bus structure (ND)	4-10
calendar	6-29 6-45
Cold Start	7-76 7-79
command decodes	4-34 5-13
communications 1-6 1-9 1-14 2-10 4-67 6-56 7-31 7-35	4-67
Computerlink	4-67
configuration	1-8 7-23 7-31 7-35
connections	1-7
console	2-15 4-70 6-30
control store	4-13
CPU	3-5 4-12
I/O registers	4-35
Traps	4-36
cycle	
control	4-19
timing	4-19
Display Unit	1-12
diagnostic	7-51
DMA	4-67
DRAM	4-40 6-100
file transfer	3-16 3-36 6-90
HDLC	4-67 4-73
upgrade	7-19
IBM-compatible	2-23 F-5 F-39
IDENT simulator	3-14 3-27 6-27
initialisation	2-9 6-59 7-20
installation number	2-15 4-37 7-17
instruction set	3-7
interrupt 2-13 3-10 3-24 3-26 3-32 4-56 6-23	6-27 6-44 6-61 6-65
I/O control	2-10 2-18
IOX simulator	3-13 3-30 6-19 6-21

keyboard 2-19 AppendixF
 communication F-32
 compatibility 2-23 6-56 6-72 F-5 F-6 F-41
 driver 6-77
 function keypad 2-25 F-19
 handler 3-32
 keystation F-27
 keytop F-6 F-27
 layout 2-20 F-6
 legends 2-20 F-6
 scan codes F-23 F-38
 template 2-26 F-20

level 16 3-13 3-27 6-21
 logic schematics AppendixA
 logic signals AppendixB

MAC 4-32
 mailbox 2-10 2-18 3-11 3-22 3-24 6-10
 master clear 4-64 7-30
 memory 2-16 4-38
 additional 1Mbyte 1-11 4-72
 addressing 2-14 4-42
 cycle 4-48
 management 2-14 3-20 4-38
 paging 2-14 4-42
 parity 4-54
 protection 2-14 4-42
 PC extended 2-16
 mapping 2-16 4-38
 upgrade 7-19
 MIC 4-14
 microcode
 bus 5-5
 mnemonics 5-27
 microinstruction
 word 5-11
 mnemonics 5-27
 microprogram
 word format 5-9
 writing 5-23
 Mouse 1-13 2-24 4-74 F-15 F-29
 monitor calls (SINTRAN) 6-40 6-82
 MON 144 (MAGTP) 6-82
 MS-DOS offset 4-48
 MS-DOS
 system calls 6-81
 structure 6-64
 multitasking 6-62 6-66

ND file services 6-81
 ND-I/O 4-41 4-67
 ND-PC communication ... 2-10 3-8 3-24 3-36 6-10 6-82 6-90
 NV-RAM 4-41 4-53 6-100

OPCOM 4-70 7-71
 oscillator 2-18 4-10

panel processor 4-70
 partition (disk) 7-20
 PAL specifications AppendixD
 parts data Chapter8
 PC 1-5 1-12 4-7
 expansion bus Appendix C
 interrupt 4-7 6-27 6-44 6-42 6-65
 memory 2-16 3-20
 software 6-62 6-64
 PC expansion bus AppendixC
 PCFSERV 3-17 3-37 6-81
 PC-NDI/O 3-14 3-26 4-64 6-41 6-66
 PC offset 2-18 4-47
 PC trap (filter) 3-17 6-94
 pipeline 4-13
 power fail 2-18 4-71
 print spooler 3-19 6-99
 PROM 4-40 6-100

real time clock 4-71
 register file 2-13 4-26
 RTNSIM 5-22

serial interface 4-67 F-18
 servers 3-19 3-37 6-81 6-91
 secret memory 2-16 4-41 4-46 6-100
 signal box 3-8 6-23 6-44
 signal names AppendixB
 system manager 3-14 6-70 6-76
 System start-up 2-9 7-44
 System Unit 1-8 2-6 2-7 7-6

terminal 1 2-15 4-70 7-71
 Telefix 7-80

UPSIM 5-21

Warm start 7-76 7-78
 watchdog timer 3-27 6-26
 Winchester disk 4-73 7-15

5.2 Communication with IBM PC-AT

5.2.1 Standard commands from DOS/BIOS

KBD ANSWER:		MEANING:			
BINARY		HEX	DEC	HEX	DEC
1 1 1 0	1 1 0 1	ED	237	FA	250
0 0 0 0	0 C N S	x	x	FA	250

SET/RESET MODE INDICATORS:

Set Mode on if bit=1

Set Mode off if bit=0

C=Capslock, N=Numlock

S=Scrolllock

1 1 1 0	1 1 1 0	EE	238	EE	238
---------	---------	----	-----	----	-----

ECHO:

Answer with EE when this test-command is received.

1 1 1 1	0 0 1 1	F3	243	FA	250
0 D D B	B A A A			FA	250

SET TYPEMATIC RATE/DELAY:

Repeat period=

$(8+A) \cdot (2 \cdot B) \cdot 0.00417 \text{ sec}$

Delay= $(1+D) \cdot 250 \text{ ms}$

1 1 1 1	0 1 0 0	F4	244	FA	250
---------	---------	----	-----	----	-----

ENABLE:

Start scanning of keys.

1 1 1 1	0 1 0 1	F5	245	FA	250
---------	---------	----	-----	----	-----

DEFAULT DISABLE:

Initialize KBD, but do not run selftest. Continue scanning if previously enabled.

1 1 1 1	0 1 1 0	F6	246	FA	250
---------	---------	----	-----	----	-----

SET DEFAULT:

Initialize KBD, but do not run selftest. Do not start scanning until enabled.

1 1 1 1	1 1 1 0	FE	254	PREVIOUS BYTE	
---------	---------	----	-----	------------------	--

RESEND:

Resend the previous byte due to errors detected by the MU.

1 1 1 1	1 1 1 1	FF	255	FA	250
---------	---------	----	-----	----	-----

RESET:

Initialize KBD with default values and execute selftest, BAT. Send a BAT completion code afterwards.

The following commands are NO-OPs and should only be answered by returning acknowledge (FA):

EF, F0, F1, F2, F7, F8, F9, FA, FB, FC, FD.

When a two-byte command is received, the keyboard is disabled (stops scanning) after the first byte. It is enabled again after the second byte is received, if it was previously enabled. If the second byte is not recognized as the correct parameter byte, the first byte is ignored.

5.2.2 Standard messages from keyboard

INFO FROM KBD:		MEANING:																	
BINARY		HEX	DEC																
<table border="1"><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>		0	0	0	0	0	0	0	0	00	00								
0	0	0	0																
0	0	0	0																
		OVERRUN: Ring buffer overrun is reported by sending 00 instead of the code that was lost.																	
<table border="1"><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>M</td><td>M</td><td>M</td></tr><tr><td>M</td><td>M</td><td>M</td><td>M</td></tr></table>		1	0	1	0	1	0	1	0	0	M	M	M	M	M	M	M	AA	170
1	0	1	0																
1	0	1	0																
0	M	M	M																
M	M	M	M																
		x	x																
		SELFTEST COMPLETE: After the selftest, AA is sent to say it is ready. The next byte indicates stuck keys: 00 means no keys are stuck. Otherwise, the Make code of the stuck key is sent. If several keys are stuck the code with highest value is sent.																	
<table border="1"><tr><td>0</td><td>M</td><td>M</td><td>M</td></tr><tr><td>M</td><td>M</td><td>M</td><td>M</td></tr></table>		0	M	M	M	M	M	M	M	x	x								
0	M	M	M																
M	M	M	M																
		KEY MAKE CODE: The make codes (01-7F) are the basic information sent from KBD to the System Unit. Two keys, for some reason, also have codes above 7F: (IBM trap for imitators?): <ul style="list-style-type: none">• Key F7 : Code 02 or 83,• Key SYS: Code 7F or 84.																	
<table border="1"><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>M</td><td>M</td><td>M</td></tr><tr><td>M</td><td>M</td><td>M</td><td>M</td></tr></table>		1	1	1	1	0	0	0	0	0	M	M	M	M	M	M	M	F0	240
1	1	1	1																
0	0	0	0																
0	M	M	M																
M	M	M	M																
		x	x																
		KEY BREAK CODES: The Break codes are the same as the Make codes, prefixed by F0.																	
<table border="1"><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr></table>		1	1	1	1	1	0	1	0	FA	250								
1	1	1	1																
1	0	1	0																
		ACK: Acknowledge reception of command from MU. If ACK is interrupted by a new command, ACK is not retransmitted.																	

1	1	1	0	1	1	1	0
---	---	---	---	---	---	---	---

EE 238

ECHO:

Acknowledge receipt of the test command Echo.

1	1	1	1	1	1	1	0
---	---	---	---	---	---	---	---

FE 254

RESEND:

Error message back to MU, meaning invalid command or parity error. No further action is expected by KBD.

The periodical keyboard test, reporting failures with the code FD, is not implemented.

5.2.3 Additions to standard commands to the keyboard

COMMAND FROM MU:

BINARY

HEX DEC

KBD ANSWER:

HEX DEC

MEANING:

1	1	1	0	0	1	0	0
0	0	c	C	n	N	s	S

E4 228

FA 250

SET/RESET LOCK MODE:
Set Mode if Even bit=1
Reset Mode if Odd bit=1
C/c=Capslock, N/n=Numlock
S/s=Scrolllock

0 E 0 E 0 E
d v d v d v

x x

FA 250

1	1	1	0	0	1	0	1
0	E	i	I	a	A	b	B

E5 229

FA 250

SET/RESET INDICATORS:
Set if Odd bit=1
Reset if Even bit=1
I/i=Ins, A/a=App, B/b=Busy, E=Error,
Blink for 10 sec and then turn off.

E 0 E 0 E 0 E
v d v d v d v

x x

FA 250

1	1	1	0	0	1	1	1
0	0	0	0	n	N	p	P

E7 231

FA 250

SET/RESET MODE
Set if Odd bit=1
Reset if Even bit=1
P/p=User Pushkeys, N/n= ND on line.

0 E 0 E
d v d v

x x

FA 250

1	1	1	0	1	0	0	0
0	D	D	D	D	D	D	D

E8 252

FA 250

SERIAL PORT OUTPUT DATA
The 7 bit ASCII code D...D is sent to the serial port output.

x x

FA 250

6 CONVERSION OF KEYSTROKE TO SYSTEM SCAN CODES AND BIOS OUTPUT

- The standard IBM key Scan Codes are 1 - 84.
- In BIOS the normal keystrokes are converted into ASCII codes 0 - 127.
- The special keystrokes are converted into "Extended ASCII", which have the ASCII 00 as prefix plus an extension number, 0x3 - 0x132.

Unused extension codes are:

1,2,4-14,26-29,39-43,51-58,69,70,74,76,78.

- Scan codes for NOTIS keys are 85 - 123.
- Extended ASCII codes for NOTIS keys are 150 - 232 (151,185 unused).
- Scan codes for mouse buttons are 124 -126.
- Extended ASCII codes for mouse buttons are 0x233 - 0x241.
- All numbers are DECIMAL.
- -- = Not defined.

6.1 Direct IBM PC equivalent keys

Key No.	Key Descr	ScanCode Base	ScanCode Shift	ScanCode Ctrl	ScanCode Alt	Ascii&Ex Base	Ascii&Ex Shift	Ascii&Ex Ctrl	Ascii&Ex Alt
1	Esc	01	01	01	01	27 x 01	27 x 01	27 x 01	--
16	Sys	84	84	84	84	0 x133 1 x133	0 x133 1 x133	0 x133 1 x133	0 x133 1 x133
20	UDK	70	70	70	70	Tog Slock 0 x169	Tog Slock 0 x170	Break --	Tog Slock --
21	Func	69	69	69	69	Tog Num 0 x172	Tog Num 0 x173	Pause --	Tog Num --
22	Print	55	55	55	55	0 x174	Prnt Scr	0 x114	0 x175
33	Caps	58	58	58	58				
34	\ FS	43	43	43	43	92 x 43	124 x 43	28 x 43	--
35	1 !	02	02	02	02	49 x 02	33 x 02	--	0 x120
36	2 @ NUL	03	03	03	03	50 x 03	64 x 03	0 x 03	0 x121
37	3 #	04	04	04	04	51 x 04	35 x 04	--	0 x122
38	4 \$	05	05	05	05	52 x 05	36 x 05	--	0 x123
39	5 %	06	06	06	06	53 x 06	37 x 06	--	0 x124
40	6 ^ RS	07	07	07	07	54 x 07	42 x 07	30 x 07	0 x125
41	7 &	08	08	08	08	55 x 08	38 x 08	--	0 x126
42	8 *	09	09	09	09	56 x 09	42 x 09	--	0 x127
43	9 (10	10	10	10	57 x 10	40 x 10	--	0 x128
44	0)	11	11	11	11	48 x 11	41 x 11	--	0 x129
45	-- US	12	12	12	12	45 x 12	95 x 12	31 x 12	0 x130
46	= +	13	13	13	13	61 x 13	43 x 13	--	0 x131
48	Backsp	14	14	14	14	8 x 14	8 x 14	127 x 14	--
58	← →	15	15	15	15	9 x 15	0 x 15	--	--
59	q Q DC1	16	16	16	16	113 x 16	81 x 16	17 x 16	0 x 16
60	w W ETB	17	17	17	17	119 x 17	87 x 17	23 x 17	0 x 17
61	e E ENQ	18	18	18	18	101 x 18	69 x 18	5 x 18	0 x 18
62	r R DC2	19	19	19	19	114 x 19	82 x 19	18 x 19	0 x 19
63	t T DC4	20	20	20	20	116 x 20	84 x 20	20 x 20	0 x 20
64	y Y EM	21	21	21	21	121 x 21	89 x 21	25 x 21	0 x 21
65	u U NAK	22	22	22	22	117 x 22	85 x 22	21 x 22	0 x 22
66	i I HT	23	23	23	23	105 x 23	73 x 23	9 x 23	0 x 23
67	o O SI	24	24	24	24	111 x 24	79 x 24	15 x 24	0 x 24
68	p P DLE	25	25	25	25	112 x 25	80 x 25	16 x 25	0 x 25
69	[(ESC	26	26	26	26	91 x 26	123 x 26	27 x 26	--
70]) GS	27	27	27	27	93 x 27	125 x 27	29 x 27	--
80	Ctrl	29	29	29	29				
81	a A SOH	30	30	30	30	97 x 30	65 x 30	1 x 30	0 x 30
82	s S DC3	31	31	31	31	115 x 31	83 x 31	19 x 31	0 x 31
83	d D EOT	32	32	32	32	100 x 32	68 x 32	4 x 32	0 x 32
84	f F ACK	33	33	33	33	102 x 33	70 x 33	6 x 33	0 x 33
85	g G BEL	34	34	34	34	103 x 34	71 x 34	7 x 34	0 x 34
86	h H BS	35	35	35	35	104 x 35	72 x 35	8 x 35	0 x 35
87	j J LF	36	36	36	36	106 x 36	74 x 36	10 x 36	0 x 36
88	k K VT	37	37	37	37	107 x 37	75 x 37	11 x 37	0 x 37
89	l L FF	38	38	38	38	108 x 38	76 x 38	12 x 38	0 x 38
90	; :	39	39	39	39	59 x 39	58 x 39	--	--
91	' "	40	40	40	40	39 x 40	34 x 40	--	--
92	~	41	41	41	41	96 x 41	126 x 41	--	--
93	Return	28	28	28	28	13 x 28	13 x 28	10 x 28	--
101	Alt	56	56	56	56				
102	L. Shift	42	42	42	42				
103	z Z SUB	44	44	44	44	122 x 44	90 x 44	26 x 44	0 x 44
104	x X CAN	45	45	45	45	120 x 45	88 x 45	24 x 45	0 x 45
105	c C ETX	46	46	46	46	99 x 46	67 x 46	3 x 46	0 x 46
106	v V SYN	47	47	47	47	118 x 47	86 x 47	22 x 47	0 x 47
107	b B STX	48	48	48	48	98 x 48	66 x 48	2 x 48	0 x 48
108	n N SO	49	49	49	49	110 x 49	78 x 49	14 x 49	0 x 49
109	m M CR	50	50	50	50	109 x 50	77 x 50	13 x 50	0 x 50
110	, <	51	51	51	51	44 x 51	50 x 51	--	--
111	. >	52	52	52	52	46 x 52	62 x 52	--	--
112	/ ?	53	53	53	53	47 x 53	63 x 53	--	--
113	R. Shift	54	54	54	54				

Key No.	Key Descr	ScanCode Base	ScanCode Shift	ScanCode Ctrl	ScanCode Alt	Ascii&Ex Base	Ascii&Ex Shift	Ascii&Ex Ctrl	Ascii&Ex Alt
122	Spacebar	57	57	57	57	32 x 57	32 x 57	32 x 57	32 x 57
28	F1	59	59	59	59	0 x 59	0 x 84	0 x104	0 x104
29	F2	60	60	60	60	0 x 60	0 x 85	0 x105	0 x105
30	F3	61	61	61	61	0 x 61	0 x 86	0 x106	0 x106
31	F4	62	62	62	62	0 x 62	0 x 87	0 x107	0 x107
32	F5	63	63	63	63	0 x 63	0 x 88	0 x108	0 x108
52	F6	64	64	64	64	0 x 64	0 x 89	0 x109	0 x109
53	F7	65	65	65	65	0 x 65	0 x 90	0 x110	0 x110
54	F8	66	66	66	66	0 x 66	0 x 91	0 x111	0 x111
55	F9	67	67	67	67	0 x 67	0 x 92	0 x102	0 x112
56	F10	68	68	68	68	0 x 68	0 x 93	0 x103	0 x113
100	-	74	74	74	74	45 x 74	45 x 74	45 x 74	-- ●
120	Enter	78	78	78	78	43 x 78	43 x 78	45 x 74	-- ●

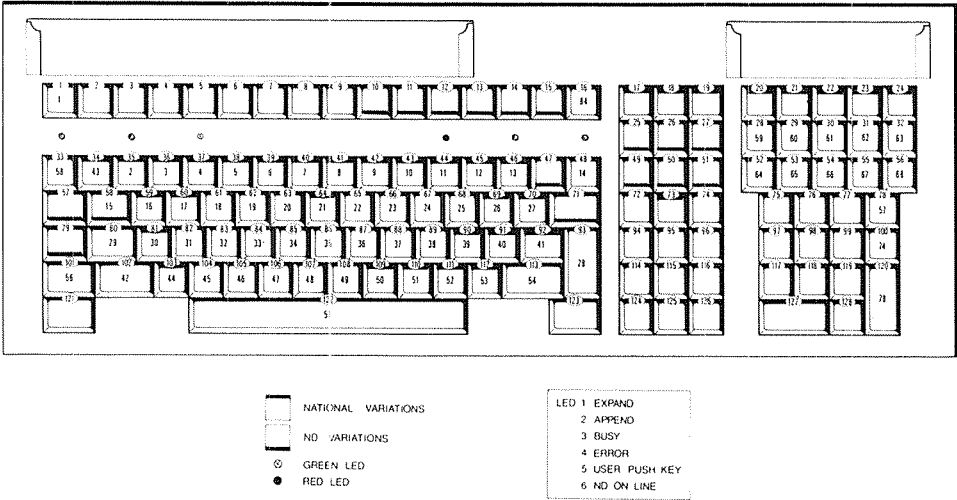
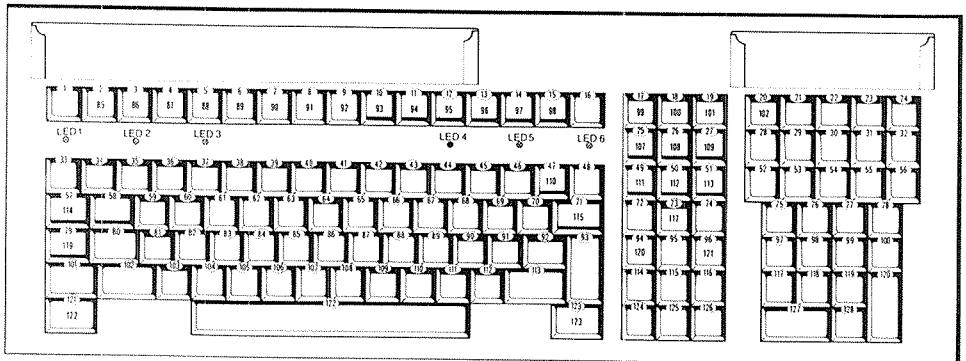


Fig F.19: Direct IBM PC equivalent keys

6.2 NOTIS-compatible keys

6.2	Key No.	Key Descri	ScanCode Base	ScanCode Shift	ScanCode Ctrl	ScanCode Alt	Ascii&Ex Base	Ascii&Ex Shift	Ascii&Ex Ctrl	Ascii&Ex Alt
	2	P1	85	85	85	85	0 x201	0 x202	0 x203	--
	3	P2	86	86	86	86	0 x204	0 x205	0 x206	--
	4	P3	87	87	87	87	0 x207	0 x208	0 x209	--
	5	P4	88	88	88	88	0 x210	0 x211	0 x212	--
	6	P5	89	89	89	89	0 x213	0 x214	0 x215	--
	7	P6	90	90	90	90	0 x216	0 x217	0 x218	--
	8	P7	91	91	91	91	0 x219	0 x220	0 x221	--
	9	P8	92	92	92	92	0 x222	0 x223	0 x224	--
	10	Mark	93	93	93	93	0 x150	0 x151	--	--
	11	Field	94	94	94	94	0 x152	0 x153	--	--
	12	Para	95	95	95	95	0 x154	0 x155	--	--
	13	Sent	96	96	96	96	0 x156	0 x157	--	--
	14	Word	97	97	97	97	0 x158	0 x159	--	--
	15	(Spare)	98	98	98	98	0 x160	0 x161	0 x162	--
	18	Copy	100	100	100	100	0 x165	0 x166	--	--
	19	Move	101	101	101	101	0 x167	0 x168	--	--
	23	Help	105	105	105	105	0 x176	0 x177	0 x178	-- ●
	24	Exit	106	106	106	106	0 x179	0 x180	--	-- ●
	25	Def tab	107	107	107	107	0 x181	0 x182	--	--
	26	Def bord	108	108	108	108	0 x183	0 x184	--	--
	27	Underlin	109	109	109	109	0 x185	0 x186	--	--
	47	Paragr	110	110	110	110	0 x187	0 x188	--	--
	49	Just l/r	111	111	111	111	0 x189	0 x190	--	--
	50	Just ar	112	112	112	112	0 x191	0 x192	--	--
	51	Just s/c	113	113	113	113	0 x193	0 x194	--	--
	57	Exp App	114	114	114	114	0 x195	0 x196	--	--
	71	Go To	115	115	115	115	0 x197	0 x198	0 x199	--
	73	Cancel	117	117	117	117	0 x237	0 x238	--	--
	78	Space	104	104	104	104	0 x200	0 x200	0 x200	-- ●
	78	Hyphen	119	119	119	119	0 x241	0 x242	--	--
	94	Prev	120	120	120	120	0 x243	0 x244	--	--
	96	Next	121	121	121	121	0 x245	0 x246	--	--
	121	(Spare)	122	122	122	122	0 x247	0 x248	0 x249	--
	122	(Spare)	123	123	123	123	0 x250	0 x251	0 x252	--



NATIONAL VARIATIONS
 ND VARIATIONS
 ⊗ GREEN LED
 ● RED LED

LED 1 EXPAND
 2 APPEND
 3 BUSY
 4 ERROR
 5 USER PUSH KEY
 6 NO ON LINE

Fig F.20: NOTIS-compatible keys

6.3 Keys not having same scan code in all states

In order to separate the number and cursor keys correctly, it is necessary to neutralize the effect of the NumLock state, and Left and Right shift keys, on some occasions. There are eight different states where special code sequences are necessary.

The code sequences are called X, Y, Z, and U, and their definition is as follows:

NumL stat	Left shft	Right shft	X	Y	Z	U
0	0	0	LM	LB	-	-
0	0	1	-	-	RB	RM
0	1	0	-	-	LB	LM
0	1	1	-	-	RB + LB	RM + LM
1	0	0	-	-	LM	LB
1	0	1	RB	RM	-	-
1	1	0	LB	LM	-	-
1	1	1	RB + LB	RM + LM	-	-

LM = Left Shift Make = 42

LB = Left Shift Break = 170

RM = Right Shift Make = 54

RB = Right Shift Break = 182

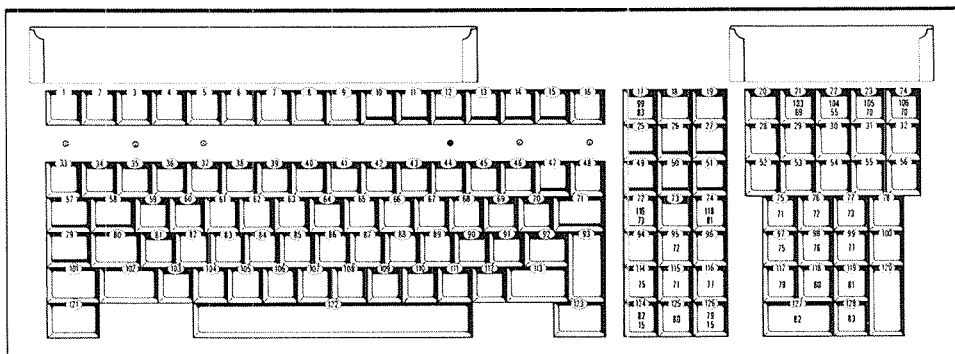
CM = Ctrl Make = 29

CB = Ctrl Break = 157

If the scan codes vary in the different states, and more than one State keys are pressed at the same time, the priority is:

- Highest = Alt,
- Next = Ctrl,
- Lowest = Shift.

Key No.	Key Descr	ScanCode Base	ScanCode Shift	ScanCode Ctrl	ScanCode Alt	Ascii&Ex Base	Ascii&Ex Shift	Ascii&Ex Ctrl	Ascii&Ex Alt
75	7	X+71 199+Y	X+71 199+Y	CB+X+71 199+Y+CM	71 199	55 x 71	55 x 71	55 x 71	7 x 00
76	8	X+72 200+Y	X+72 200+Y	CB+X+72 200+Y	72 200	56 x 72	56 x 72	56 x 72	8 x 00
77	9	X+73 201+Y	X+73 201+Y	CB+X+73 201+Y+CM	73 201	57 x 73	57 x 73	57 x 73	9 x 00
97	4	X+75 203+Y	X+75 203+Y	CB+X+75 203+Y+CM	75 203	52 x 75	52 x 75	52 x 75	4 x 00
98	5	X+76 204+Y	X+76 204+Y	CB+X+76 204+Y+CM	76 204	53 x 76	53 x 76	53 x 76	5 x 00
99	6	X+77 205+Y	X+77 205+Y	CB+X+77 205+Y+CM	77 205	54 x 77	54 x 77	54 x 77	6 x 00
117	1	X+79 207+Y	X+79 207+Y	CB+X+79 207+Y+CM	79 207	49 x 79	49 x 79	49 x 79	1 x 00
118	2	X+80 208+Y	X+80 208+Y	CB+X+80 208+Y+CM	80 208	50 x 80	50 x 80	50 x 80	2 x 00
119	3	X+81 209+Y	X+81 209+Y	CB+X+81 209+Y+CM	81 209	51 x 81	51 x 81	51 x 81	3 x 00
127	0	X+82 210+Y	X+82 210+Y	CB+X+82 210+Y+CM	82 210	48 x 82	48 x 82	48 x 82	0 x 00
128	.	X+83 211+Y	X+83 211+Y	CB+X+83 211+Y+CM	CB+83 211+CM	46 x 83	46 x 83	46 x 83	--
17	Delete	Z+83 211+U	99 227	83 211	83 211	0 x 83	0 x163	-- Reset	--
72	Pg up	Z+73 201+U	116 244	73 201	116 244	0 x 73	0 x239	0 x132	--
74	Pg dwn	Z+81 209+U	118 246	81 209	118 246	0 x 81	0 x240	0 x118	--
95	Up	Z+72 200+U	Z+72 200+U	72 200	--	0 x 72	0 x 72	--	--
114	Left	Z+75 203+U	Z+75 203+U	75 203	--	0 x 75	0 x 75	0 x115	--
115	Home	Z+71 199+U	Z+71 199+U	71 199	--	0 x 71	0 x 71	0 x119	--
116	Right	Z+77 205+U	Z+77 205+U	77 205	--	0 x 77	0 x 77	0 x116	--
124	⇧ Ins	Z+82 210+U	102 230	82 210	102 230	0 x 82	0 x 15	--	--
125	Down	Z+80 208+U	Z+80 208+U	80 208	--	0 x 80	0 x 80	--	--
126	⇩ End	Z+79 207+U	103 231	79 207	103 231	0 x 79	9 x 15	0 x117	--



□ NATIONAL VARIATIONS
 □ ND VARIATIONS
 ⊗ GREEN LED
 ● RED LED

LED 1: EXPAND
 2: APPEND
 3: BUSY
 4: ERROR
 5: USER PUSH KEY
 6: ND ON LINE

Fig F.21: Keys not having same scan code in all states

6.4 Mouse push buttons

The first line contains the Make (press) code, and the next line contains the Break (release) code.

First line contains Make code, next line contains the Break (release) code.

Key No.	Key Descr	ScanCode Base	ScanCode Shift	ScanCode Ctrl	ScanCode Alt	Ascii&Ex Base	Ascii&Ex Shift	Ascii&Ex Ctrl	Ascii&Ex Alt
SW1	Index	124	124	124	--	0 x225	0 x226	0 x227	--
		252	252	252		0 x234	0 x234	0 x234	--
SW2	Middle	125	125	125	--	0 x228	0 x229	0 x230	--
		253	253	253		0 x235	0 x235	0 x235	--
SW3	Ring	126	126	126	--	0 x231	0 x232	0 x233	--
		254	254	254		0 x236	0 x236	0 x236	--

7 SYSTEM TO "AT" SCAN CODE CONVERSION TABLES

In order to produce the correct scan code in the keyboard when it is attached to an AT System Unit, the System scan codes in section 6 must be converted to AT scan codes. The following table specifies the relation between System and AT scan codes, sorted in increasing SYSscan number order. The AT key numbers (as they appear in IBM documentation) are quoted for reference only.

ATkey DEC	ATscan HX	SYSScan DEC	ATkey DEC	ATscan HX	SYSScan DEC	ATkey DEC	ATscan HX	SYSScan DEC	ATkey DEC	ATscan HX	SYSScan DEC	ATkey DEC	ATscan HX	SYSScan DEC
90	76	118	01	1		53	41	65	33	51		R	5E	94
2	16	22	02	2		54	49	73	34	52		R	08	08
3	1E	30	03	3		55	4A	74	35	53		R	10	16
4	26	38	04	4		57	59	89	36	54		R	18	24
5	25	37	05	5		106	7C	124	37	55		R	20	32
6	2E	46	06	6		58	11	17	38	56		R	28	40
7	36	54	07	7		61	29	41	39	57		R	30	48
8	3D	61	08	8		64	58	88	3A	58		R	38	56
9	3E	62	09	9		70	05	05	3B	59		R	40	64
10	46	70	0A	10		65	06	06	3C	60		R	48	72
11	45	69	0B	11		71	04	04	3D	61		R	50	80
12	4E	78	0C	12		66	0C	12	3E	62		R	57	87
13	55	85	0D	13		72	03	03	3F	63		R	6F	111
15	66	102	0E	14		67	0B	11	40	64		R	13	19
16	0D	13	0F	15		73	02	02	41	65		R	19	25
17	15	21	10	16			83	131	41	65		R	39	57
18	1D	29	11	17		68	0A	10	42	66		R	51	81
19	24	36	12	18		74	01	01	43	67		R	53	83
20	2D	45	13	19		69	09	09	44	68		R	5C	92
21	2C	44	14	20		95	77	119	45	69		R	5F	95
22	35	53	15	21		100	7E	126	46	70		R	62	98
23	3C	60	16	22		91	6C	108	47	71		R	63	99
24	43	67	17	23		96	75	117	48	72		R	64	100
25	44	68	18	24		101	7D	125	49	73		R	65	101
26	4D	77	19	25		107	7B	123	4A	74		R	67	103
27	54	84	1A	26		92	6B	107	4B	75		R	68	104
28	5B	91	1B	27		97	73	115	4C	76		R	6A	106
43	5A	90	1C	28		102	74	116	4D	77		R	6D	109
30	14	20	1D	29		108	79	121	4E	78		R	6E	110
31	1C	28	1E	30		93	69	105	4F	79				
32	1B	27	1F	31		98	72	114	50	80				
33	23	35	20	32		103	7A	122	51	81				
34	2B	43	21	33		99	70	112	52	82				
35	34	52	22	34		104	71	113	53	83				
36	33	51	23	35		105	7F	127	54	84				
37	3B	59	24	36			84	132	54	84				
38	42	66	25	37		R	60	96	55	85				
39	4B	75	26	38		R	61	97	56	86				
40	4C	76	27	39		R	78	120	57	87				
41	52	82	28	30		R	07	07	58	88				
1	0E	14	29	41		R	0F	15	59	89				
44	12	18	2A	42		R	17	23	5A	90				
14	5D	93	2B	43		R	1F	31	5B	91				
46	1A	26	2C	44		R	27	39	5C	92				
47	22	34	2D	45		R	2F	47	5D	93				
48	21	33	2E	46		R	37	55	5E	94				
49	2A	42	2F	47		R	3F	63	5F	95				
50	32	50	30	48		R	47	71	60	96				
51	31	49	31	49		R	4F	79	61	97				
52	3A	58	32	50		R	56	86	62	98				

GLOSSARY

Appendix B contains a list of the logic signal names used in the Butterfly ND-110PCX logic diagrams, together with their meanings.

This Glossary complements this list, adding commonly used abbreviations and terms which appear in this manual and which have special meaning in Butterfly. Some of these terms are also common terms used throughout the computer industry, but are included here for completeness.

ALU	Arithmetic and Logic Unit.										
ANSI	American National Standards Institution										
ASCII	American National Standard Code for Information Interchange. This comprises a character set consisting of 7-bit coded characters (8 bits if a parity check bit is added). The character set includes the alphabetic and numeric characters, together with a selection of graphic and control characters										
baud	The unit describing the signalling speed across a serial communications link, expressing how many bits per second are being transferred across the link. There are standard multiples of baud rates offered in commercial, telephone and industrial environments, as follows: <table><tr><td>50 baud</td></tr><tr><td>110 ..</td></tr><tr><td>300 ..</td></tr><tr><td>600 ..</td></tr><tr><td>1200 ..</td></tr><tr><td>2400 ..</td></tr><tr><td>4800 ..</td></tr><tr><td>9600 ..</td></tr><tr><td>18200 ..</td></tr><tr><td>36400 ..</td></tr></table>	50 baud	110 ..	300 ..	600 ..	1200 ..	2400 ..	4800 ..	9600 ..	18200 ..	36400 ..
50 baud											
110 ..											
300 ..											
600 ..											
1200 ..											
2400 ..											
4800 ..											
9600 ..											
18200 ..											
36400 ..											
Bar Code Reader	An optical reading device that can read bar-coded characters and translate them into digital output suitable for input as character information, to a computing machine. The bar code is formed using an encoding system made up from vertical lines of differing width and spacing. There are several bar code encoding schemes in existence: the more common ones are 3 of 9 Code, Interleaved 2 of 5 Code, Code 11, Codabar, and UPC/EAN/JAN Codes.										
BIOS	Basic Input / Output System. This is the general name given to the software code (peripheral device drivers, etc.) which interfaces the operating system (software) with the hardware of the computer system. In Personal Computing, it has special recognition as IBM's BIOS in their IBM-PC systems. Compatible IBM-PC manufacturers either use the IBM BIOS under licence, or generate their own functionally compatible version of it.										
BSM	BIOS Sharing Module: this refers to software in the PC side of an OWS-110 and Teamstation, which controls the sharing of the BIOS between simultaneous requests for the same device, originating from more than one source (e.g. the ND and the PC).										

bus	One or more connector lines which form a functional entity. May have more than one source, and usually feeds several destinations. A bus may be a power supply line, or a group of several data/address/control lines. It provides a highway for distribution of the function it represents.
Butterfly	The range of ND-Butterfly products includes the OWS-10, OWS-11, OWS-12, OWS-110 and TeamStation.
Butterfly-110	The hardware which is common to an OWS-110 machine and a Teamstation machine.
Butterfly keyboard	The name given to the PC/NOTIS keyboard used in the ND-Butterfly product range. This keyboard provides compatibility with IBM-PC keyboard and also provides for all the ND-NOTIS functions, in the one keyboard.
Butterfly Supervisor	The name of the ND-utility program for OWS-110 and Teamstation, which offers set-up and maintenance facilities to the "supervisor" of a Butterfly installation. This person is also referred to as the Desk Supervisor: their rôle in an ND-Butterfly installation is to look after the system control features, and act as first-line support for users of the ND-Butterfly machines.
byte	In ND computers, a group of 8 consecutive bits (Binary digiTs).
calendar	A function in the computer which provides information on calendar time, in the form of year, month, day, time of day (hours, minutes and seconds).
cache	An area of fast-access local memory, between CPU and Main Memory, used to store the programs and data are most recently used by the computer, and so significantly reduce the time that the computer would otherwise need to access the same program/data in Main Memory.
capslock	The keyboard state in which the standard alphabetic keys are output in capital letters (i.e. upper case).
CCITT	Comite Consultatif International de Telephonie et de Telegraphie: an international committee set up under the International Telecommunications Union, to promote standards for the development of telephone, telegraph systems and data networks, and the environment for interworking between the different national networks in the world.
Centronics interface	Centronics is the name of the company which originated a popular parallel interface for data communication. This interface is commonly referred to as a "Centronics interface"

character	One of a set of graphic symbols and/or functional expressions, used to denote a single item in that set. In ND computers, each character is represented by a unique 8-bit pattern (byte).
cluster	A grouping of equipment, centred around a main item of equipment.
code (software)	The program statements which together form a program or a part of a program.
Computerlink	One type of ND communications facility.
Cold Start	In ND computers, this involves resetting the computer and then loading a copy of the "save area" of main program into the computer memory, also generating a new "image area" copy. See also "Warm Start".
colour display	Refers to a colour monitor. In ND-Butterfly, the colour monitor option offers a selection of 16 displayable colours from a palette of 64 available colours.
CPU	Central Processing Unit: the functional unit of computing hardware which fetches, interprets and executes instructions in a program.
current loop	serial communication in which the signals in the line connection are carried electrically as a 0/20mA current flow. Contrasts with voltage loop.
cycle	An interval of time in which one set of events is completed. The interval of time may vary for differing sets of events.
Desk Supervisor	The name occasionally given to the Supervisor of an ND-Butterfly installation - both in terms of the users and the facilities.
Display Unit	Same as visual display unit, or monitor.
diagnostic	Diagnostic programs provide the facility to check the operation of various parts of the system hardware, to detect fault conditions, and in doing this to point to probable causes of a detected fault.
DMA	Direct Memory Access.
DRAM	Dynamic Random Access Memory: the term "dynamic" implies that to retain its stored data, the storage device requires a memory refresh cycle within a specified interval.
EGA	Enhanced Graphics Adaptor - video driver board, as used in IBM-PC compatible machines.
EIS	Ericsson Information Systems.

FIFO	First-In First-Out: a queuing technique in which the next item to be retrieved is the item that has been in the queue for the longest time. Thus, data is read from a FIFO buffer in the same order as it was written into the buffer.
file	A unit of storage in a computer system, used to store related information. Files may be of different "types", this meaning they are used to store information of differing natures, for example :text, :data, :out, :plot, :clcb, etc.
FASP	File Access Service Program
logic gate array	A high density arrangement of logic gates on a single piece of silicon material. These logic gates may be linked to make up standard integrated circuit logic functions, and interconnected on the silicon to form complete functional circuits mounted on a single package.
HDLC	<u>H</u> igh <u>l</u> evel <u>D</u> ata <u>L</u> ink <u>C</u> ontrol. This is a class of Communications protocols which are bit-oriented rather than character-oriented. Within the HDLC class, the following protocols exist: Link Access Protocol (LAP), Balanced LAP (LAPB), and Synchronous Data Link Control (SDLC).
IBM-compatible	Compatible with the functionality of the IBM Personal Computer. This implies that it will function as if it were an IBM-PC, including being able to run any application program that will run on an IBM-PC.
IDENT	IDENT instruction in ND computers.
interrupt	A suspension of a program, caused by an event external to that program, in order to pass control to another routine which will service the cause of the interrupt. The interrupt is handled in such a way that when it has been dealt with (if appropriate, queued for later processing), the original program can resume from the point at which it was interrupted.
initialisation	The operation to reset to a known starting condition.
IOX	ND input/output instruction.
k	abbreviation for kilo, $\times 10^3$.
keyboard	<p>The equipment which provides arrangements of keys which operate as pushbuttons. A user presses keys to input characters and functions to a machine to which the keyboard is connected. Various lamps on the keyboard also provide output of status information to a keyboard user.</p> <p>Keyboards are often specific to a given type of machine (e.g. NOTIS kbd, VT-100 kbd, IBM-PC kbd, etc.). The ND-Butterfly keyboard provides dual compatibility with the ND and the PC environments, functioning as a ND-NOTIS keyboard and as an IBM-PC keyboard.</p>

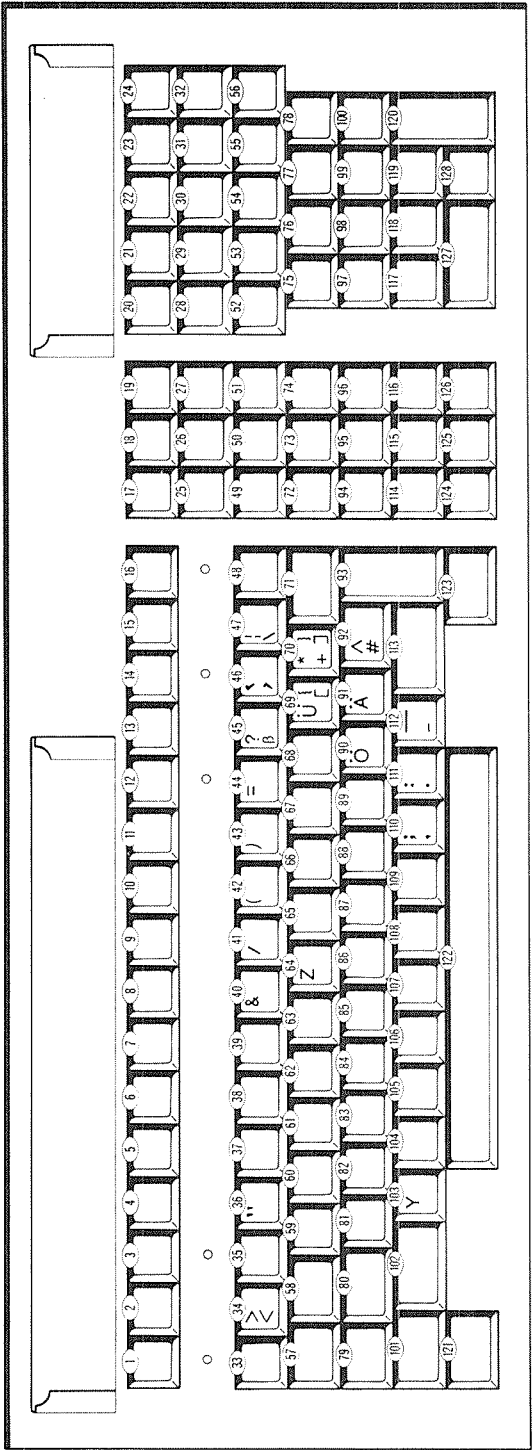


Fig F.7: Butterfly keyboard - German

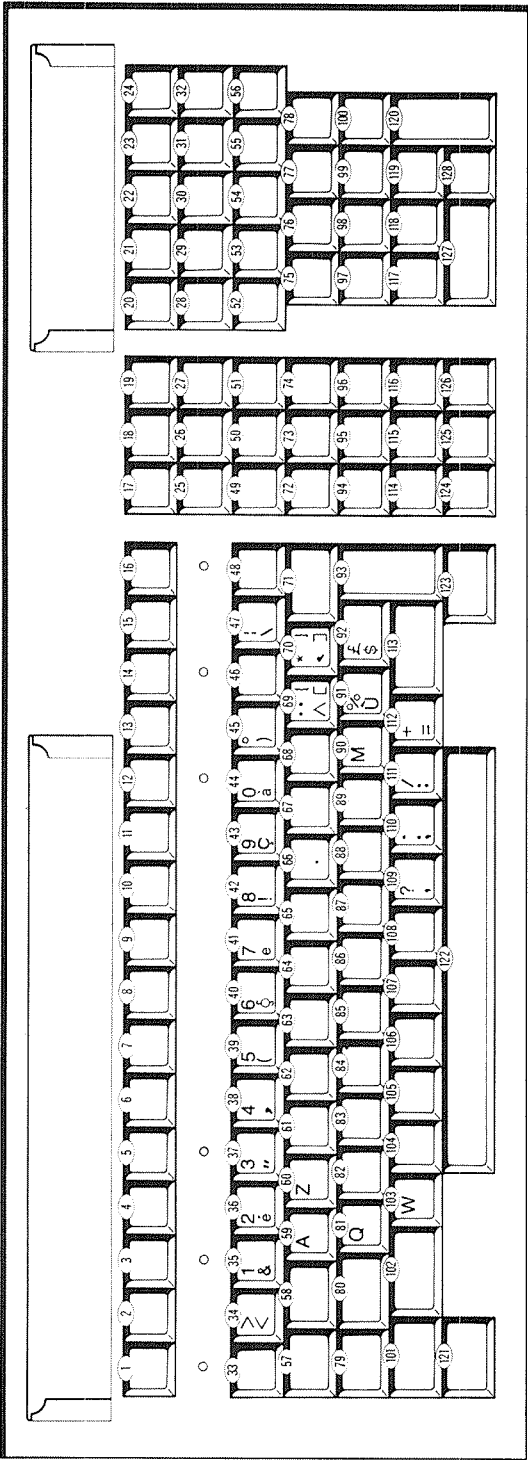


Fig F.8: Butterfly keyboard - French

2 NEW FEATURES

2.1 Mouse

In the middle of the rear edge of the keyboard (fig F.9) is a "mouse hole", for connection of an industry standard mouse. A "quadrature TTL" signal standard is assumed. The mouse is connected through a 9-pin D-connector.

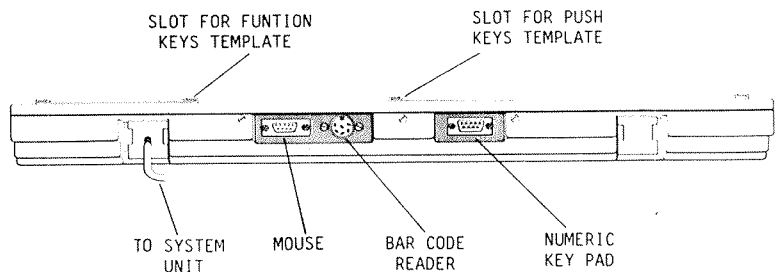


Fig F.9: Keyboard rear connectors

The Mouse transmits four position signals, X1, X2, Y1 and Y2. The X-direction is UP, and Y-direction is LEFT. It has three pushbuttons: SW1, SW2 and SW3. These are operated by the INDEX, MIDDLE and RING finger, respectively. The mouse program in Butterfly-110 may be configured to swap the functions of SW1 and SW3, so that left-handed persons may use their index finger to operate the most-used button - see section 2.1.3.

2.1.1 Mouse position

When the mouse is moved, the X and Y movements are registered as a number of pulses: 100 per inch. The higher the moving speed, the higher is the frequency of the pulse train. However the total number of pulses is the same, irrespective of whether the mouse movement is slow or fast. The maximum moving speed is about 10 inches per second, which gives a pulse frequency of approximately 1kHz.

The X1/Y1 and X2/Y2 signals are identical in frequency, but have different phases, to indicate up/left or down/right movements. All position information is incremental, and the keyboard does not keep track of any absolute positions. The maximum rate of input of position codes to the CPU from the keyboard is 35 displacements per second.

2.1.2 Position Coding

The Mouse relative movement is specified as an 9-bit binary number. Bits 0-6 contain the displacement. Bits 7-8 indicate four possible directions of movement, as follows:

Bit8	Bit7	Direction
0	0	+ X Up
0	1	- X Down
1	0	+ Y Left
1	1	- Y Right

These 9 bits are sent from the keyboard as a simulation of 5 keystrokes. First, a two-code header sequence ALT, INDEX is sent. Then follows three octal digits, represented by keystrokes 0-7 in the main typewriter area. The use of the mouse keys in the ALT state is reserved for left/right mode (see next section). Therefore ALT INDEX represents a unique code sequence.

For example, the movement of 59 steps to the left is represented as direction XY=10 and displacement = 0111011. This gives three octal characters 100, 111, 011, = 4, 7, 3. The scan codes sent to the computer in this case are ALT INDEX 4 7 3

$$= \text{AltMake} + \text{IndexMake} + \text{IndexBreak} + \text{AltBreak} + 4\text{Make} + 4\text{Break} + 7\text{Make} + 7\text{Break} + 3\text{Make} + 3\text{Break}$$

$$= 56 + 124 + 252 + 184 + 5 + 133 + 8 + 136 + 4 + 132$$

2.1.3 Push button coding

A mouse button is treated as any other key on the keyboard. The keys are assigned scan codes 124-126. The scan codes are not specified for three physical keys, but according to the three fingers INDEX, MIDDLE and RING.

Mouse Key	RIGHT HAND MODE		LEFT HAND MODE	
	Finger	Scan Code	Finger	Scan Code
#1	Index	124	Ring	126
#2	Middle	125	Middle	125
#3	Ring	126	Index	124

The ALT state is used to switch between left and right hand mode:

- ALT #1 means "set right hand mode". This is the default setting after "system initialize".
- ALT #2 means "set left hand mode".

ALT #1, #2 and #3 are therefore always suppressed in the keyboard. The application program need not know about the left/right swap arrangement of mouse buttons - that is local between the operator and the keyboard.

2.2 Serial Port / Bar Code Reader

serial interface is intended for a Bar Code Reader or a similar serial device. The connector - a DIN-type female - is marked in fig F.9.

2.2.1 Serial input

The Serial Read Line receives 7-bit ASCII codes (the 8th bit is always zero). They are converted into equivalent key strokes (scan codes), then sent to the CPU.

The number of possible codes received from Bar Code Readers are currently of the order of 40-50. However, the conversion table is prepared with 128 entries, so that Norsk Data can make modifications in the future.

The Header is the keystroke Ctrl @ (=NUL).

The Termination keystroke is CR (Return).

When the keyboard ring buffer is full, an XOF code is issued on the Serial Transmit Line. When the ring buffer can accept more keystrokes, an XON signal is transmitted.

2.2.2 Serial output

The connected device may receive control information from the CPU, via the keyboard. This is received in the form of 7-bit ASCII codes, preceded with command code E8 hex (Serial Output Port Data). These codes are sent forward by the keyboard on the Serial Transmit Line, without conversion.

2.2.3 Data format

The serial format is:

- 1 Start bit
- 8 Data bits (bits 0-6 = standard ASCII, bit7 = 0)
- 1 Odd Parity bit
- 1 Stop bit
- 1200 baud.

2.3 Separate function keypad

Provision is made for a separate Function Keypad which is either similiar to an IBM function pad or a numeric pad. Typical keypad layouts are shown in fig F.10. The connector for this - a 9-pin D-type male - is marked in fig F.9.

An extra key, tentatively called COMMAND is included, intended as a prefix key for command input from the pad.

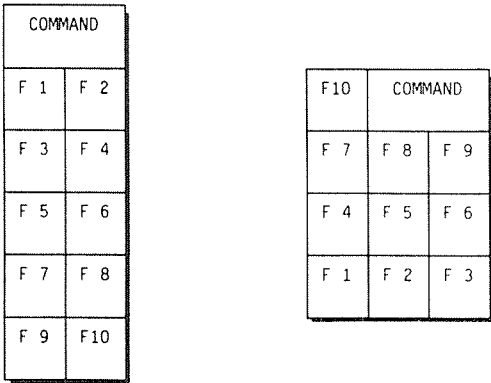


Fig F.10: Typical key layouts for external Function Keypad

The pad contains ten Function keys. These work in parallel with the existing Function keys on the keyboard.

The pad is connected by a 500mm cable to the back of the keyboard, and can be used on the left or right side. The pad may be attached to the side of the keyboard using strips of Velcro (nylon hook fabric).

2.4 Template card guides

Above the Function Keys are guide slots for retaining a template card. For each program/application, this template card (fig F.11) may be marked to show the meaning of each of the 10 function keys.

CTRL										INDEX	MIDDLE	RING
SHIFT												
BASE												

Fig F.11: Function Keys template

Over the Push-keys are guide slots for a similar template card (fig F.12), which may be marked by the user to identify each of the possible 8x3 predefined Push-key sequences. The templates can contain either the operators own defined keys, or preprogrammed key values for special application programs.

Fig F.12: Push-keys template

3 SOFTWARE SPECIFICATIONS

3.1 Initialization

The processor is initialized at power-up, or upon command from the System Unit. Initialization commands from PCs cause automatic hardware reset, while PC-ATs send a command byte.

The initialization consists of setting default values, and a self-test.

3.2 System Unit model

At initialization, the keyboard must find out which PC model it is connected to. The relevant models are:

IBM-compatible PC	-	abbreviated PC
IBM-compatible PC-AT	-	abbreviated AT

To distinguish the two models, two Model signals (DIP switches) are read at initialization time, to decide which is connected.

MOD1	MOD0	
OFF	OFF	= PC
OFF	ON	= AT

3.3 Interface signal standards

The interface signal standards are different for the three models. Although the data and clock signals are electrically similar, the timings and meanings are different.

The interface specifications should be obtained from the manufacturer of the relevant model.

3.4 Communication protocols

The communication protocols of the three models are different, with regard to commands and messages between Keyboard and System Unit, and to the codes for the different keystrokes. Section 5 of this Appendix cites each protocol, subdivided into three categories:

- a) standard commands and messages
- b) modifications of the standard messages
- c) additions to the standard commands.

In case of discrepancy between the information supplied by the PC model manufacturer, and ND's specifications of standard protocols, the model manufacturer's specification should be followed.

3.5 Self-test

The self-test performs the following checks (standard AT test is accepted):

- Verify checksum of the PROM.
- Test RAM, by storing the address value in address, then the inverted address value in address.
- Check if any keys are permanently stuck, and send information to the System Unit.
- Toggle all indicators, such that an observer can see they switch on and off.

If any errors are detected, the ERROR indicator is switched on for 10-30 seconds.

3.6 Initialization acknowledge

The keyboard must acknowledge initialization not later than 250ms after the command is given. If it fails to do so, the System Unit considers the keyboard absent. The acknowledge should be sent before the toggle-indicator part of self-test is finished. A problem arises where the self-test takes longer than 250ms, as is the case in IBM's AT keyboard. See section 5 of Appendix F.

3.7 Scan codes

The keys on a PC keyboard are numbered from 1 to 83, and the scan code for a key is equal to the key number. Olivetti also follows this scheme, in their M24. These codes are referred to as System Scan Codes.

On the AT, the keys are numbered differently, and the AT scan codes are translated into System Scan Codes, before they are delivered to BIOS. The AT scan codes are therefore only valid in the keyboard and on the serial lines into the System Unit.

This specification assumes that the principles of communicating through Make and Break codes are understood, and therefore in the lists of scan codes in section 6, only the PC Make codes (= System Scan codes) are quoted. However for some special sequences, the Break sequence is also specified.

The AT codes may be found from a conversion table from PC to AT Make codes, supplied in section 7.

3.8 Special case handling

A high degree of compatibility with IBM must exist without amendments to the BIOS. This requirement dictates that some manipulation must be done in the code conversion.

From a normal PC keyboard, each key has its own scan code. This is sent to the System Unit, whether or not the state keys (Shift, Ctrl or Alt) are pressed. In the current keyboard, different scan codes may be sent for different state keys. Also, sequences of scan codes may be sent to "undo" what an already-pressed state key has done.

3.8.1 Separation of Numeric Pad and Cursors

On the IBM keyboard, ASCII 0-9 are produced on the numeric pad in the shift state, or if Numlock is on. The use of shift if Numlock is on already, brings back the base state. Numlock is bistable in the same way as Capslock, and the effect of it is controlled in BIOS and not by the keyboard. Scrolllock has a similiar meaning for cursor movement, but the effect is much less dramatic.

The natural way of avoiding problems is to omit the Numlock key, and to precede number-pad keys with a Shift Make code, and end it with a Shift Break code. However, a problem arises if Numlock and Scrollock are omitted: the Microsoft Flight Simulator program, which is the current-accepted measure of IBM compatibility, skips the assigned meaning, and uses them as command keys.

3.8.2 Numlock

The Notis FUNC and HELP keys are also used as Numlock and Scrollock keys. Unfortunately, BIOS suppresses them, because they are only intended for internal housekeeping in the keyboard driver.

The FUNC and HELP keys must therefore send two different codes, one for NO use and one for IBM-compatible use:

FUNC= Func-Make + Numlock-Make ... Func-Break + Numlock-Break
HELP= Help-Make + Scrollock-Make ...
..... Help-Break + Scrollock-Break

The keyboard must record if the Func/Numlock key has been pressed an even or an odd number of times: if even, the Numlock state is off; if odd, the Numlock state is active. In addition to the Numlock state, the software must also account for the state of the two shift keys. The code sequences to be sent for the different state combinations are specified in section 6.

The Capslock and Scrollock states do not affect the codes for any of the keys. However, their states should be recorded, and shown on indicators.

3.8.3 Lock indicators

The Scrollock state is shown on a yellow indicator in the indicator area.

The Capslock state is shown on a green indicator on the CAPS key. A high-current LED should be used, to ensure that the green indication is clearly visible.

The Numlock state does not include an indicator, since a major effort to neutralize its effects has been made.

In principle, the keyboard must keep track of the lock status flags itself. However, some System Units send additional information regarding their status, in which case the flags must be updated accordingly.

- The PC sends no information.
- The M24 sends information about Capslock and Numlock.
- The AT sends information about Capslock, Numlock and Scrolllock.

The format of the Lock status from the System Unit is given in section 5.

3.9 Typematic action

All keys are typematic. This means that when a key is held pressed, the keyboard sends Make-codes continuously, until the key is released.

The typematic rate is fixed at 80ms. The delay before typematic action starts is 250ms.

When two or more keys are held down simultaneously, only the last key pressed repeats at the typematic rate. The typematic operation stops when the last key pressed is released, even if other keys are still held pressed.

An exception to this is the cursor keys. If two cursor keys are pressed simultaneously, continuous Make-codes for both keys are produced. This provides for future facility to move the cursor at 45 degree angles on the display, by pressing two adjacent cursor movement keys at the same time.

3.10 Ring Buffer

The Ring Buffer to the System Unit handles the asynchronous operation between the various filling processes and the sending routine. In the IBM and Olivetti models, the buffer is about 20 codes deep. However, it is advantageous for this to be larger, because the mouse and the serial port could exceed this number. (One shifted ASCII code produces 4 scan codes in the buffer). A 40-deep buffer is therefore preferred. A lower number may be acceptable, by negotiation with Norsk Data.

3.11 PROM size

To enable Norsk Data to make future modifications, the program is stored in a separate PROM chip (i.e. outside the processor), in a socket. An extra address line is required in the PCB layout, to allow for a future upgrade to 16kbyte PROMS.

4 ELECTRO-MECHANICAL SPECIFICATION

4.1 Keystation layout and keytop colours

Keystations The keyboard has 128 keystations. These are arranged as shown in fig F.13.

Keytop colours There are three keytop colours, as specified in fig F.14.

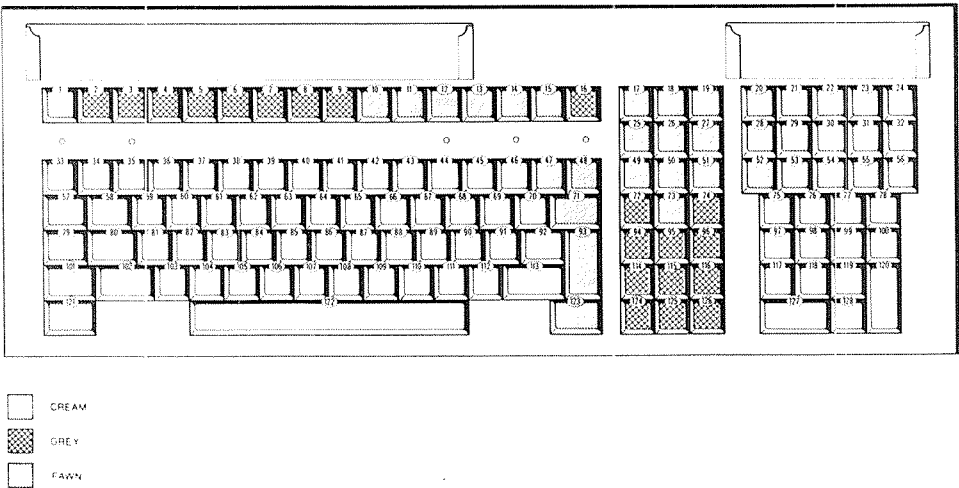


Fig F.14: Keytop colours

4.2 Function pad

The Function Pad consists of mechanical switches that can be scanned in an X-Y manner, by means of 4+3 signal lines. They produce the same scan codes as the keyboard Command and Function keys.

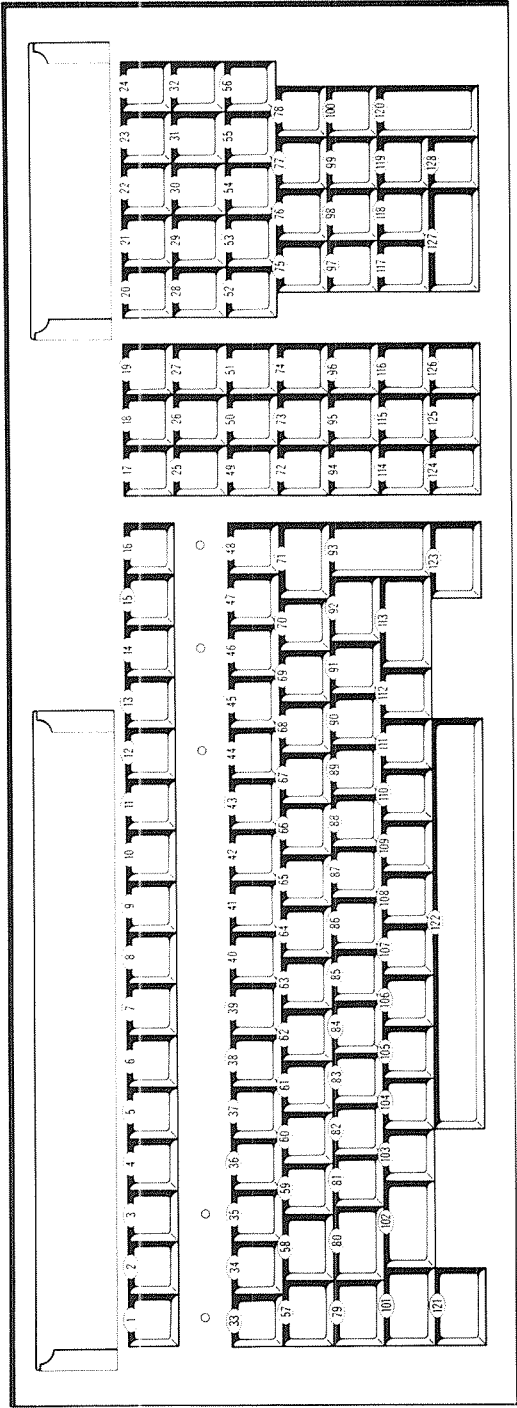


Fig F.13: Keystations and their reference numbers

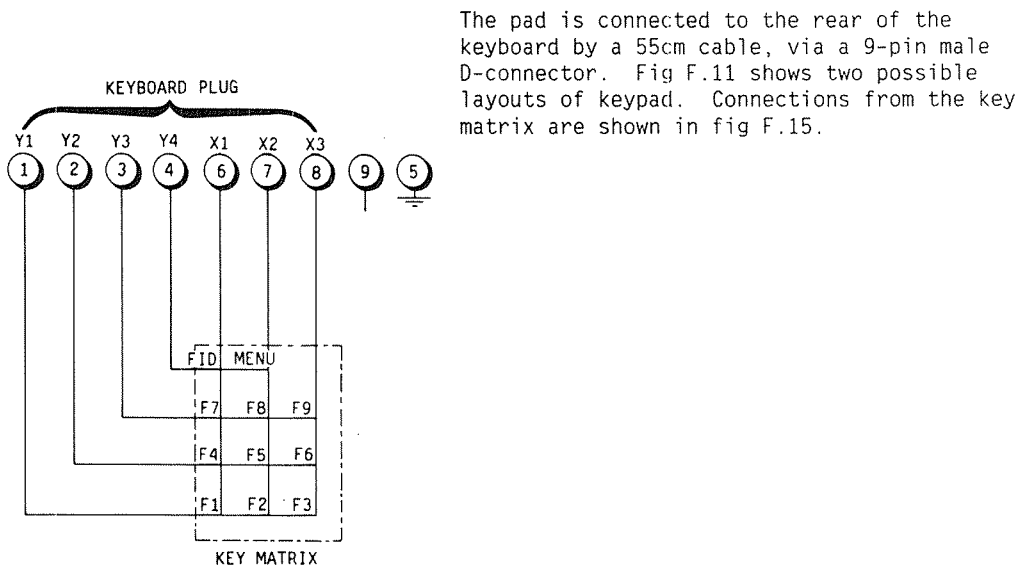


Fig F.15: Function Pad connections

4.3 Mouse port

The mouse port is a 9-pin female D-connector, on the rear of the keyboard. The X/Y interface signals and cable connections are shown in fig F.16.

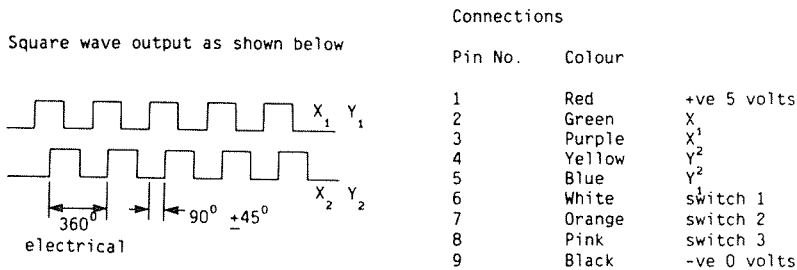


Fig F.16: Mouse interface

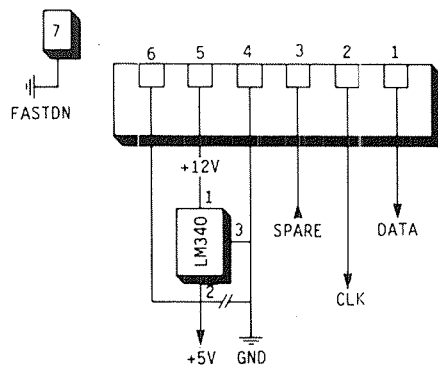
4.4 LED layout

Apart from CapsLock, all the LEDs are concentrated above the main key area (see fig F.13). The text is printed on a plastic strip that can be substituted for each national variant. The standard version is shown in fig F.1.

4.5 Bar Code Reader

The Bar Code Reader is not included in the standard system, but an interface for it is included in the keyboard, so that a Bar Code Reader unit may be connected at a later date. The motherboard includes a connector to link to the small bar code reader interface card. The card is accessed by removing the cover. A strain relief on the wand cable provides adequate support, to avoid strain on the card itself.

4.6 System Unit cable and connectors



The keyboard is attached to the System Unit by a 190cm coiled cable (same length as that in the IBM AT). This is shown in fig F.17. The cable is secured at the keyboard with a strain relief. The cable screen is grounded via an AMP spade terminal. A ferrite ring is threaded through the cable at the keyboard end to reduce EMI noise.

The 6-pin Berg connector accommodates both 5V and 12V supply rails when it is wired as shown in fig F.18.

Fig F.18: Berg connector to keyboard

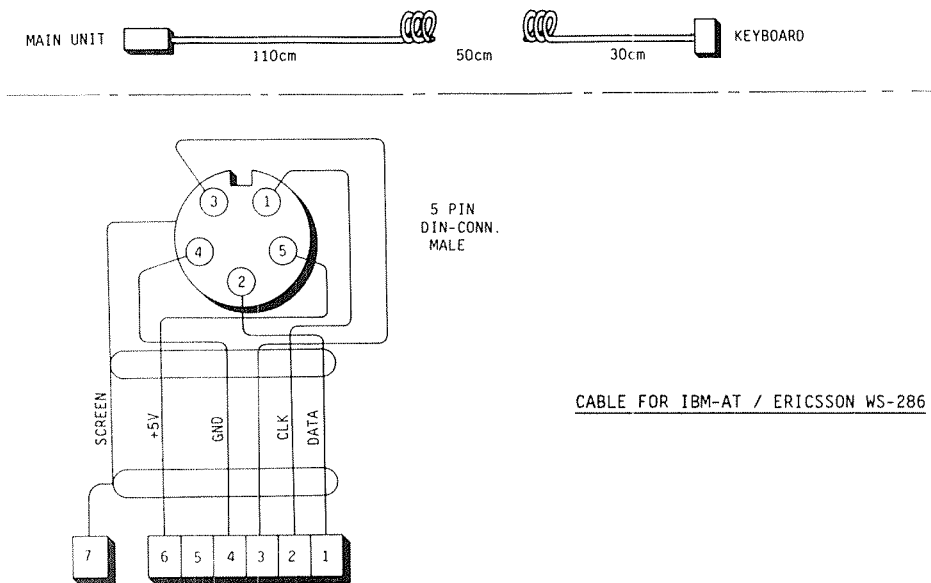


Fig F.17: Keyboard to System Unit cable

4.7 Other engineering specifications

- Key type: capacitor, linear pressure.
- Keyboard key rows profile: sculptured.
- Font for legends: Gorton.
- Keyboard enclosure colour: to be specified.
- Special keys: raised contour on keys 5 (in numeric keypad), F and J, to enable tactile location of numeric keypad and of hands position in main keypad areas.
- Clear lens on Capslock.
- Enclosure: low-profile, as per dimension drawing. Two internal struts, to support the keytop surface.
- Keytop profile: as per drawing.
- Vertical key movement: total - 3.0mm nominal
to operate - 1.0mm nominal.

5 KEYBOARD TO MAIN UNIT COMMUNICATION

5.1 Communication with Standard IBM PC

The initialize command consists of the System Unit pulling the Clock line low for at least 20ms. This resets the keyboard to default values, a self-test is performed, and a message is sent to the System Unit afterwards.

A standard PC cannot send other information to the keyboard.

NOTE

Implementation of the protocol used by the voice keyboard is preferred. The keyboard indicators can then be controlled from system program.

Standard messages from keyboard

<table><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>M</td><td>M</td><td>M</td></tr></table>	1	0	1	0	0	M	M	M	AA 170	SELF-TEST COMPLETE: After the self-test, AA is returned to indicate it is ready. The next byte identifies any stuck keys. 00 means no keys stuck. Otherwise, the Make code of the stuck key is sent. If several keys are stuck, the code with highest value is sent.
1	0	1	0							
0	M	M	M							
<table><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>M</td><td>M</td><td>M</td></tr></table>	1	0	1	0	0	M	M	M	x x	
1	0	1	0							
0	M	M	M							
<table><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	1	1	1	1	1	1	1	1	FF 255	OVERRUN: Ring buffer overrun is reported by sending FF instead of the code that was lost.
1	1	1	1							
1	1	1	1							

PAL16L8 (A)		ROGER EDWARDS
I/O BUS CONTROL	REG NO 41500	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 22	(U106) VERSION E	14-NOV-86

```

/IOMM CWRITE /IOBGNT /IODS /IOTIME IOREQ LBA4 NC NC GND
/LIOAO /SCCRD /SCCWR IOBRD /EIOBLO /EIOBHI SLBTIO /LBTIO DLBTIO VCC

```

$$\text{IF (VCC) /DLBTIO} = \text{IOBNT} * \text{/IOBRD} \\ + \text{IOMM} * \text{/CWRITE}$$

```
IF (VCC) LBTIO      = IOBGNT * IOBRD
                    + IOMM * CWRITE
```

$$\text{IF (VCC) /SLBTIO} = \text{IOTIME} * \text{IOBGNT} * \text{IOBRD} \\ + \text{IOMM} * \text{CWRITE}$$

```
IF (VCC) EIOBLO = IOMM ;PROG I/O READ OR WRITE
+ IOBGNT * /IOBRD * /LIOAO ;DMA BYTE WRITE
+ /IOMM * IOBRD * IODS * IOBGNT ;DMA WORD READ
+ EIOBLO * IODS * IOBRD * /IOMM ;HOLD ON IF DMA READ
;UNTIL IODS FINISHED
```

```
IF (VCC) EIOBHI = IOMM ;PROG I/O READ OR WRITE
+ IOBGMT * /IOBRD * LIOAO ;DMA BYTE WRITE
+ /IOMM * IOBRD * IODS * IOBGMT ;DMA WORD READ
+ EIOBHI * IODS * IOBRD * /IOMM ;HOLD ON IF DMA READ
;UNTIL IODS FINISHED
```

```
IF (IOMM) /IOBRD = CWRITE
```

```
IF (VCC) SCCWR = IODS * /IOBRD
                + IOMM * IOTIME * /LBA4 * /IOBRD      ;EXTENDED WHEN CPU WRITE
                + IOMM * SCCWR                          ;HOLD ON ONCE ITS ON
```

```
IF (VCC) SCCRD      = IODS * IOBRD
                    + IOMM * /LBA4 * IOBRD          ;EXTENDED WHEN CPU READ
                    * /CWRITE                        ;AVOID SPURIOUS READ IN A WRITE CYCLE
                    * /SCCWR                         ;TO AVOID READ WHEN CWRITE GOES OFF
```

DESCRIPTION

Note - IOBO (pin 8) and DMAALE (pin 9) not required as inputs any longer, but are connected on pilots.

B issue: EIOBLO AND EIOBHI modified during DMA Reads to produce Word Reads of a duration determined by IODS active. IOREQ no longer required - JTG
27/8/86

C issue: 7-NOV-86 Hold on TERM taken out of IOBRD now that faster SCC's used

D issue: 7-NOV-86 SCCWR delayed till IOTIME to give additional setup time for prog I/O writes to new SCC's.

E issue: 14-NOV-86 Hold on TERM added to EIOBLO & EIOBHI so that DMA Read Enable duration is controlled by IODS.

PAL16L8 (B)

MEMORY CONTROL

PILOT BUTTERFLY PAL 23

REG NO 42000

(U205) VERSION C

ROGER EDWARDS

NORSK DATA LTD, NEWBURY

22-SEP-86

/CAK SCACT DSTART TSTART /MSTART DELOUT CAO CA1 /IOMM GND
 BACK /TIMPULS IOAO IOA1 DMAALE /IOBGNT IOBO /LIOAO /DISWHI VCC

IF (VCC) TIMPULS = CAK * SCACT * TSTART * MSTART * DELOUT ;IO
 + /CAK * DSTART * MSTART * DELOUT ;NORMAL
 + /SCACT * DSTART * MSTART * DELOUT ;NORMAL

IF (VCC) /IOAO = /CAO * /IOMM * /BACK ;PASS
 + /IOAO * IOMM * /BACK ;HOLD
 + /CAO * /IOAO * /BACK ;HAZARD

IF (VCC) /IOA1 = /CA1 * /IOMM * /BACK ;PASS
 + /IOA1 * IOMM * /BACK ;HOLD
 + /CA1 * /IOA1 * /BACK ;HAZARD

IF (VCC) LIOAO = DMAALE * /IOBO ;PASS
 + /DMAALE * LIOAO ;HOLD
 + /IOBO * LIOAO ;HAZARD

IF (IOBGNT) DISWHI = /LIOAO

DESCRIPTION

B issue: 18-AUG-86: SCACT inverted to correct artwork error - JTG

C issue: 22-SEPT-86 LIOAO inverted with respect to IOBO to correct DMA byte writes.

APPENDIX E BUTTERFLY-110 LINK SETTINGS

TABLE OF CONTENTS

Section		Page
1	ND-110PCX LOCAL BUS BOARD (3402)	E-3
2	PC SYSTEM BOARD	E-4

LIST OF ILLUSTRATIONS

Figure		Page
E.1	ND-110PCX Local Bus board (3402) link settings	E-3
E.2	PC System board links	E-4

1 ND-110PCX LOCAL BUS BOARD (3402)

On the ND-110PCX there are two links to be checked.
They are both located on the Local Bus board.

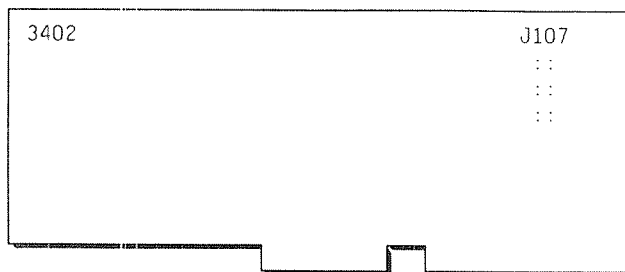


Fig E.1: ND-110PCX CPU board link settings

Main/Standby 5V	Pin 3 must be connected to pin 5. This connects the main 5V supply to standby-power 5V.
Extra DOS RAM	<p>Pin 1 may be linked to pin 2, to DISABLE the PC from being given 128kbytes of extra RAM in the ND DRAM area. With this link not connected, the PC sees this extra RAM in its address range 512 - 640kbyte.</p> <p>This link should only be connected if another memory board is installed which covers the same address range, or for test purposes.</p>
Test points	<p>On the other pins of J107, the following voltage test points are available:</p> <p>pin 3 : main +5V</p> <p>pin 5 : standby +5V</p> <p>pin 7 : +12V</p> <p>pins 6 & 8 : 0V (ground)</p>

2 PC SYSTEM BOARD

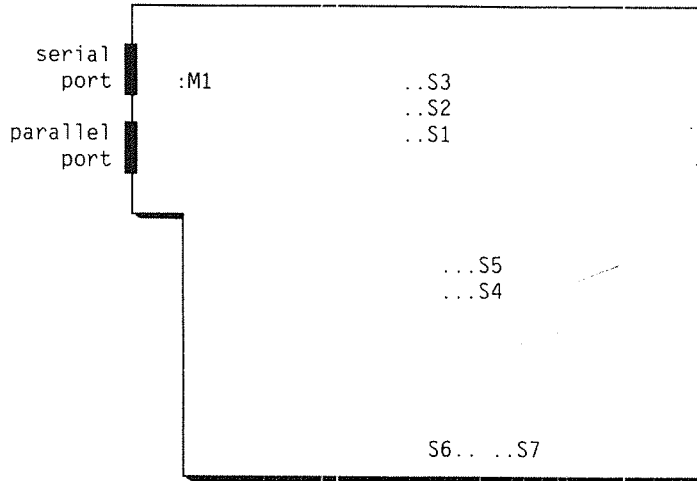


Fig E.2: PC System board links

Serial port

S1 enables the serial port to be COM1
S2 enables the serial port to be COM2.

If both are strapped or both are open, the serial port is disabled.

The standard setting is S1 connected

The only reason to change this setting is if an option board is fitted which uses the same serial port address (COM1).

Parallel port

S3 enables the parallel port to be LPT1.

The standard setting is S3 connected.

The only reason to change this setting is if an option board is fitted which uses the same parallel port address (LPT1).

Oscillator speed S4 and S5 select the speed of the oscillator for the 80286 processor.

 If pins 1 & 2 are linked, the speed setting is 8MHz.

 If pins 2 & 3 are linked, the speed setting is 6MHz.

S4 and S5 must always be set to be the same. The standard setting is for 8MHz.

Reasons to change it may be that a particular software package requires 6MHz, or fails to run satisfactorily at 8MHz, or that some optional hardware is installed that uses the same clock as the 80286 processor and 8MHz is too fast for it.

PROM size S6 defines the size of the PROMs

 S6 open specifies PROM type 27128

 S6 connected specifies PROM type 27256.

The standard setting is open.

Wait-states S7 defines the number of wait-states for memory accesses.

 S7 open specifies no wait-states

 S7 connected specifies one wait-state.

The standard setting is open.

Reasons to change it are to become more IBM-compatible, or to accommodate applications software that is more critical on timing and needs to be set to IBM-standard (which is one wait-state). In such instances, the speed setting may also need to be changed to 6MHz.

5V to serial port M1 enables +5V to pin 18 on the serial port connector. M1 is normally connected in Butterfly-110, since this enables use of the Facit current-loop adaptor.

 If equipment which does not like +5V on pin 18 is to be connected to the serial port, then link M1 should be removed. For most equipment, pin 18 on the serial port is not used.

APPENDIX F BUTTERFLY KEYBOARD SPECIFICATION

TABLE OF CONTENTS

Section		Page
1	INTRODUCTION	F-5
1.1	IBM compatibility	F-5
1.2	NOTIS compatibility	F-6
1.3	Standard keyboard layout and legends	F-6
2	NEW FEATURES	F-15
2.1	Mouse	F-15
2.1.1	Mouse position	F-15
2.1.2	Position coding	F-16
2.1.3	Push button coding	F-17
2.2	Serial port / Bar Code Reader	F-18
2.2.1	Serial input	F-18
2.2.2	Serial output	F-18
2.2.3	Data format	F-18
2.3	Separate function keypad	F-19
2.4	Template card guides	F-20
3	SOFTWARE SPECIFICATIONS	F-21
3.1	Initialization	F-21
3.2	System Unit model	F-21
3.3	Interface signal standards	F-21
3.4	Communication protocols	F-22
3.5	Self-test	F-22
3.6	Initialization acknowledge	F-22
3.7	Scan codes	F-23
3.8	Special case handling	F-23
3.8.1	Separation of numeric pad and cursors	F-23
3.8.2	Numlock	F-24
3.8.3	Lock indicators	F-24
3.9	Typematic action	F-25
3.10	Ring buffer	F-25
3.11	PROM Size	F-26
4	ELECTRO-MECHANICAL SPECIFICATION	F-27
4.1	Keystation layout and keytop colours	F-27
4.2	Function pad	F-27
4.3	Mouse port	F-29
4.4	LED layout	F-30
4.5	Bar Code Reader	F-30
4.6	System Unit cable and connectors	F-30
4.7	Other engineering specifications	F-31

5	KEYBOARD TO SYSTEM UNIT COMMUNICATION	F-32
5.1	Communication with standard IBM PC	F-32
5.2	Communication with IBM PC-AT	F-33
5.2.1	Standard commands from DOS/BIOS	F-33
5.2.2	Standard messages from keyboard	F-34
5.2.3	Additions to standard commands to the keyboard	F-35
6	CONVERSION OF KEYSTROKE TO SYSTEM SCAN CODES AND BIOS OUTPUT .	F-36
6.1	Direct IBM PC equivalent keys	F-37
6.2	NOTIS-compatible keys	F-39
6.3	Keys not having same scan codes in all states	F-40
6.4	Mouse push buttons	F-42
7	SYSTEM TO "AT" SCAN CODE CONVERSION TABLE	F-43

LIST OF ILLUSTRATIONS

Figure		Page
F.1	Butterfly keyboard: ND/IBM US-English (VT100)	F-7
F.2	Butterfly keyboard: 8 special PC keys (green legend)	F-8
F.3	Butterfly keyboard: ND/IBM International-English (ANSI)	F-9
F.4	Butterfly keyboard: Norwegian	F-10
F.5	Butterfly keyboard: Danish	F-11
F.6	Butterfly keyboard: Swedish	F-12
F.7	Butterfly keyboard: German	F-13
F.8	Butterfly keyboard: French	F-14
F.9	Keyboard rear connectors	F-15
F.10	Typical layouts for external Function Keypad	F-19
F.11	Function Keys template	F-20
F.12	Push Keys template	F-20
F.13	Keystations and their reference numbers	F-28
F.14	Keytop colours	F-27
F.15	Function pad connections	F-28
F.16	Mouse interface	F-29
F.17	Keyboard to System Unit cable	F-31
F.18	Berg connector to keyboard	F-30
F.19	Direct IBM PC equivalent keys	F-38
F.20	NOTIS-compatible keys	F-39
F.21	Keys not having same scan code in all states	F-41

1 INTRODUCTION

This specification defines a NOTIS keyboard which can be used on an IBM compatible PC. Some new features that are not required for compatibility reasons are also included. The specification includes a Mouse Port, a Bar Code Reader port and a Menu & Function pad.

The PC keyboard standard is fairly fixed with regards to key functions. However, indicators, electrical interface, and protocol in the communication with the main PC unit, vary between different models.

There are two mutually incompatible PC models on which ND wish to be able to use a Notis keyboard. They are:

- Original IBM-PC and clones.
- IBM PC-AT.

The new features are:

- Mouse interface in the keyboard.
- Bar Code Reader interface.
- Port for an external Menu/Function Pad.
- Guides for insertion of template cards adjacent to the Function and Push keys.

1.1 IBM Compatibility

It is essential that the keyboard works with the IBM-assigned keys, without changing the keyboard driver in the BIOS. The key strokes that are output by the driver are converted by a special keyboard handler before they are sent to the ND. It is in this handler that the keystrokes are converted to the same codes and Escape sequences that the NOTIS terminal produces.

However, a large number of the special NOTIS keys are not recognized by BIOS, and are therefore discarded. To allow these codes to pass, as well as enable programming of Push-keys, it is necessary to lift the keyboard driver out of BIOS and define the existence of the new keys.

At first sight, it may seem unnecessary to take the effort to be compatible with the existing BIOS if this is to be changed anyway. However, several programs that run on the IBM, make their own version of the keyboard driver, under the assumption that they receive original IBM Scan codes from the keyboard. For example, the Microsoft Flight Simulator program might not run if the greatest emphasis was not put on compatibility. In this specification, all the IBM key functions are therefore implemented with the correct scan code - even the apparently redundant Numlock key.

1.2 NOTIS compatibility

- Two NOTIS keys have been removed: LOCK and LOCAL. However LOCAL is substituted by SYS, which will serve the same functions plus a few more.
- Four new IBM keys have been introduced: ALT, F9, F10 and SYS.
- There are three other new yet-unassigned keys, mainly for NORTEXT typesetting systems, and for future graphics features in NOTIS.
- One key function has been redefined: SHIFT PRINT now means "Print Screen" instead of "Activate Formatter".

1.3 Standard keyboard layout and legends

Fig F.1 shows the keytop legends for an ND/IBM US-English (VT100) style keyboard. Some keys which have special significance for PC operations (fig F.2), are engraved in green on the front face of the keytop.

The differences from "VT100" for key-layout variants to suit ND/IBM International-English (ANSI), Norwegian, Danish, Swedish, German and French languages, are given in figs F.3 - F.8. In each case, figs F.1 - F.8 indicate the relative position and size of the legend on each keytop.

The basic version of keyboard firmware is that for the ND/IBM US-English keyboard. All other national variants are obtained from this, by changing keytops and using key-modifier software routines in the PC.

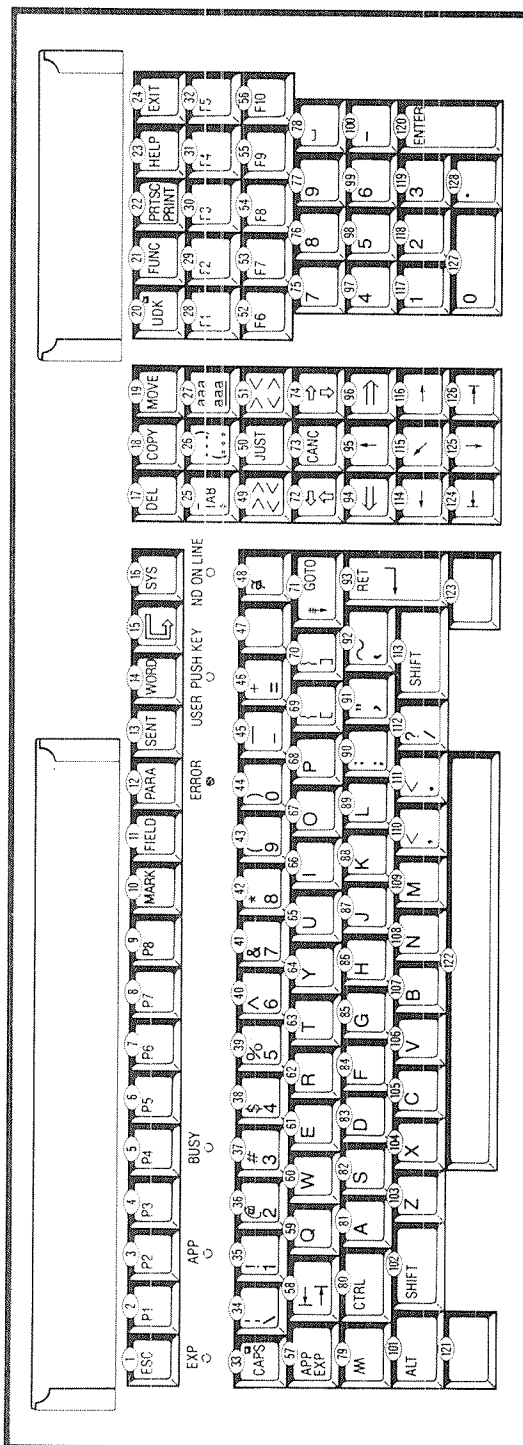


Fig F.1: Butterfly keyboard - ND/IBM US-English (VT100)

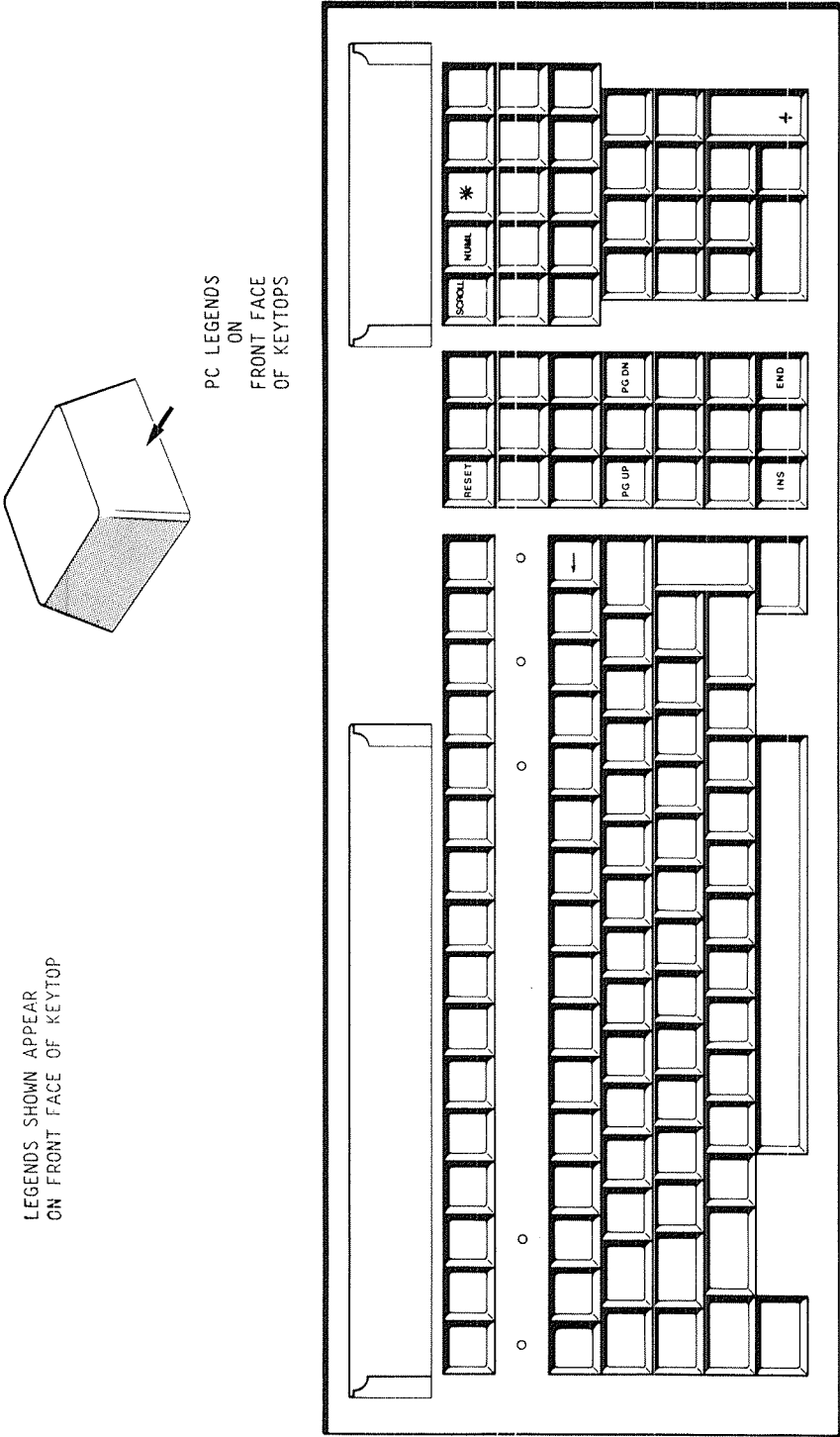


Fig F.2: Butterfly keyboard - special PC keys (green legend)

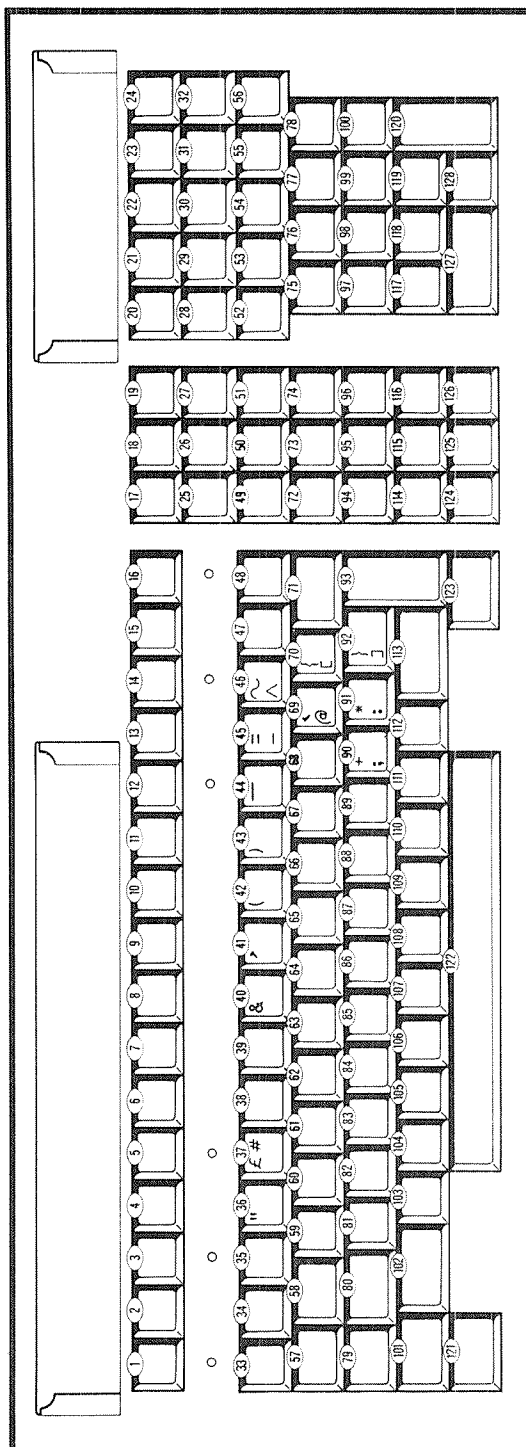


Fig F.3: Butterfly keyboard - ND/IBM International-English (ANSI)

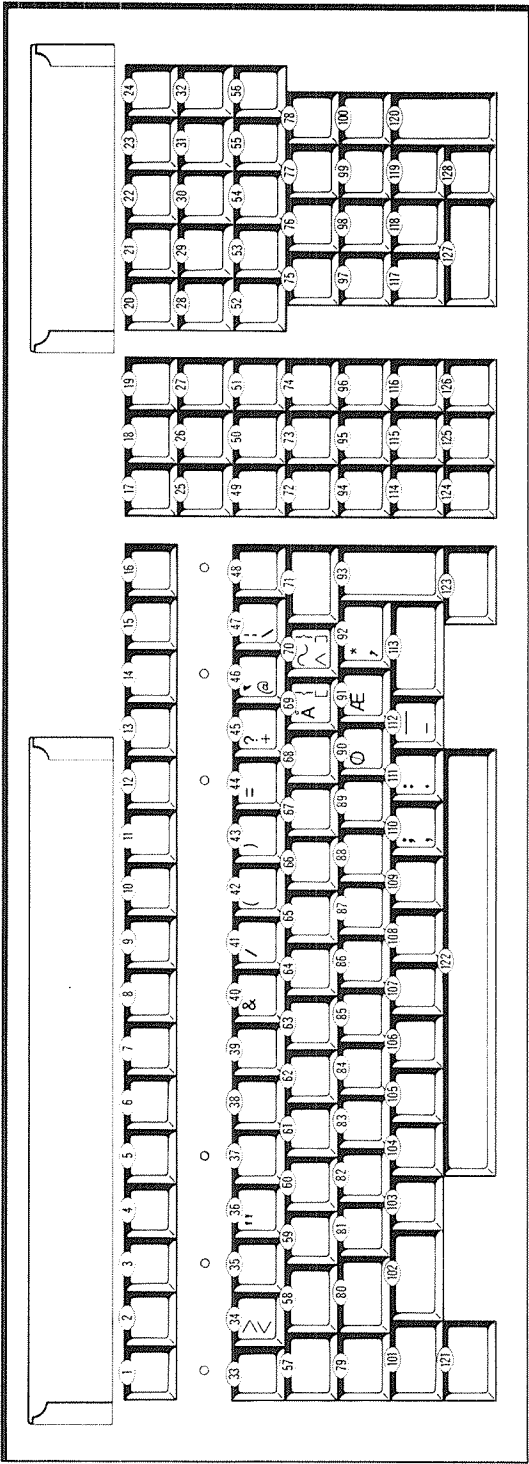


Fig F.4: Butterfly keyboard - Norwegian

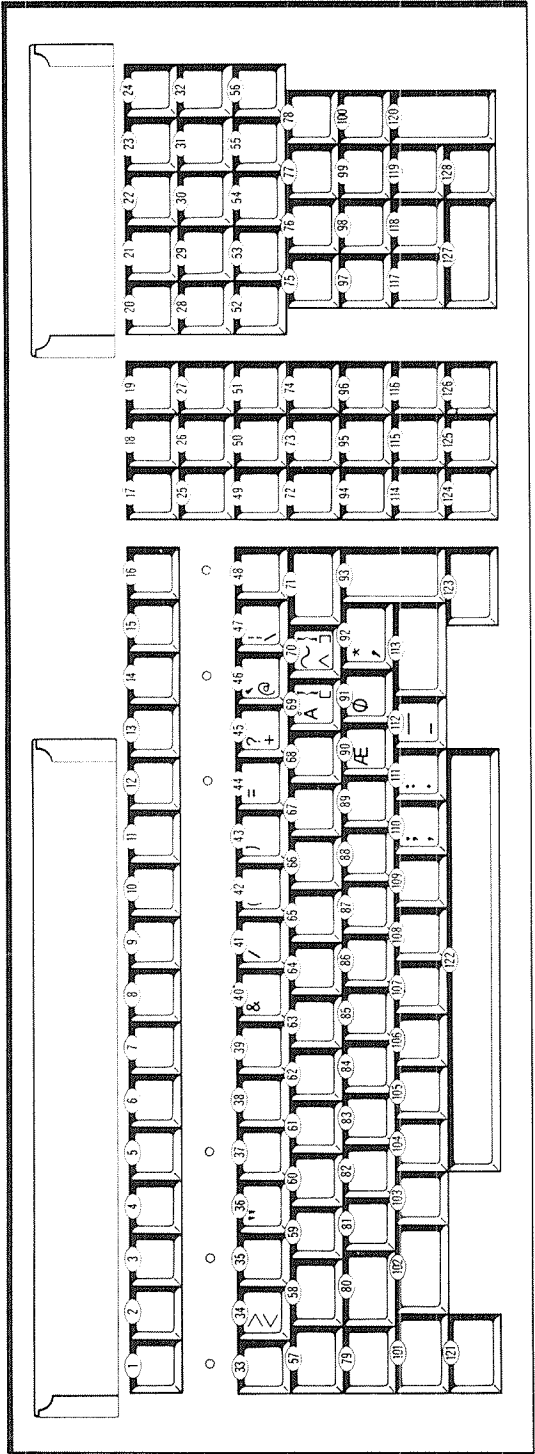


Fig F.5: Butterfly keyboard - Danish

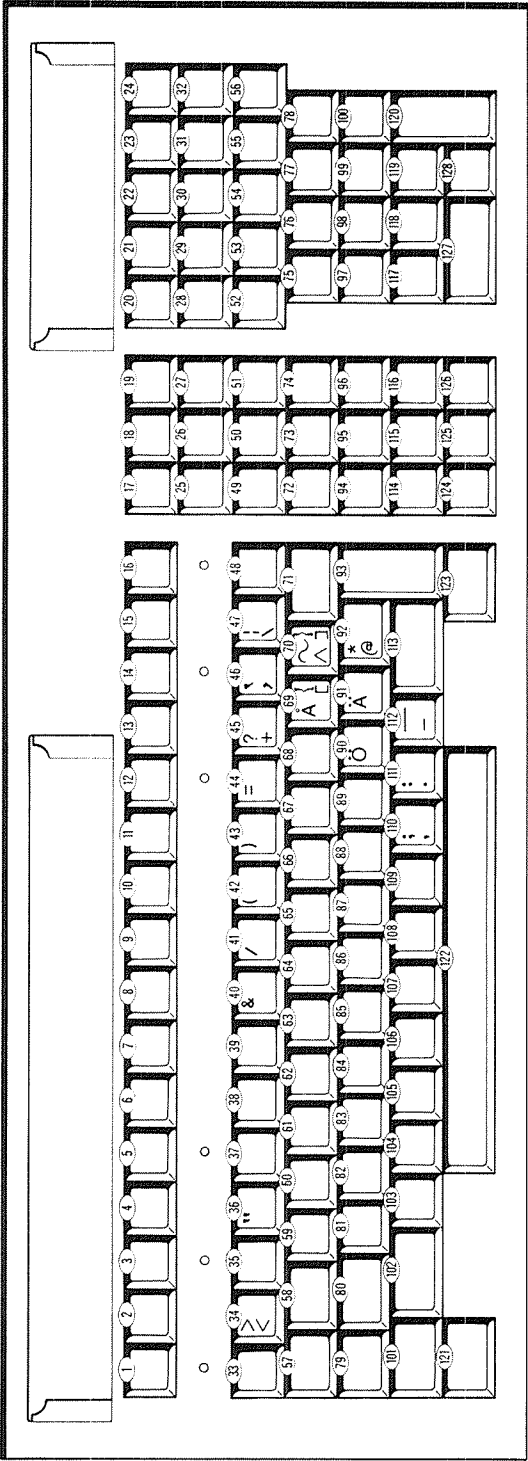


Fig F.6: Butterfly keyboard - Swedish

PAL16L8 (A2)

COMMAND DECODE 5

REG NO 40400

ROGER EDWARDS

NORSK DATA LTD, NEWBURY

PILOT BUTTERFLY PAL 5

(U19)

VERSION A

2-JUN-86

M0 M1 C0 C1 C2 C3 C4 MCLK NC GND

NC /LDIRV WRF /SSTOP /SSTART /LDPIL /IND NC /LDLC VCC

$$\text{IF (VCC) LDLC} = /C4 * C3 * C2 * C1 * C0$$

$$\begin{aligned} \text{IF (VCC) LDIRV} &= C4 * /C3 * /C0 * M1 * M0 * /MCLK \\ &+ C4 * /C3 * C1 * M1 * /MCLK \\ &+ C4 * /C3 * C1 * C0 * /MCLK \\ &+ C4 * /C3 * C2 * /MCLK \end{aligned}$$

$$\text{IF (VCC) IND} = C4 * /C3 * /C2 * C1 * /C0 * /M1$$

$$\text{IF (VCC) LDPIL} = /C4 * /C3 * /C2 * /C1 * C0$$

$$\text{IF (VCC) SSTART} = /C4 * C3 * /C2 * C1 * C0$$

$$\text{IF (VCC) SSTOP} = /C4 * C3 * C2 * /C1 * /C0$$

$$\begin{aligned} \text{IF (VCC) /WRF} &= C4 \\ &+ C3 \\ &+ C2 \\ &+ /C1 \\ &+ /C0 \\ &+ MCLK \end{aligned}$$
DESCRIPTION

WRF converted from /WRF - must gate (NAND) with /TERM decodes to 03.

PAL16R8 (A2)		ROGER EDWARDS
IDB SOURCE DECODE 1	REG NO 41001	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 6	(U21) VERSION A	2-JUN-86

CLK NC NC NC S0 S1 S2 S3 S4 GND
 /OE /TRAALD /EIOR /JMPA /EPEA /EPES /EPGS /ENCD /EDO VCC

EDO := /S2 * /S3 * /S4
 + /S0 * /S3 * /S4
 + /S1 * /S2 * S3 * /S4
 + /S0 * /S1 * S3 * /S4
 + /S0 * S1 * /S2 * /S3;

ENCD := S0 * /S1 * S2 * /S3 * S4;

EPGS := S0 * S1 * /S2 * /S3 * S4;

EPES := S0 * S1 * /S2 * S3 * /S4;

EPEA := /S0 * S1 * /S2 * S3 * /S4;

JMPA := /S0 * /S1 * S2 * S3 * S4;

EIOR := /S0 * S1 * S2 * S3 * /S4;

TRAALD := /S0 * S1 * S2 * /S3 * S4;

PAL16R8 (A2)

IDB SOURCE DECODE 2

REG NO 41000

ROGER EDWARDS

NORSK DATA LTD, NEWBURY

PILOT BUTTERFLY PAL 7

(U22)

VERSION A

2-JUN-86

CLK FO CSALUI8 /CLFF S0 S1 S2 S3 S4 GND

/OE /RRF /DZL DZD /RINR /STOF /ALUI8 /EPICV /EPAN VCC

ALUI8 := CSALUI8

DZL := FO

$$\begin{aligned} /DZD := & /ALUI8 * /DZD \\ & + /FO * /DZD \\ & + /DZL * /DZD \\ & + CLFF \end{aligned}$$

EPAN := S0 * S1 * S2 * /S3 * S4

EPICV := S0 * /S1 * /S2 * S3 * S4

$$\begin{aligned} STOF := & /S4 * S2 * S0 \\ & + /S4 * S3 * S1 \\ & + S4 * /S3 * S2 \\ & + S4 * /S2 * /S1 \\ & + S4 * /S2 * S0 \\ & + S3 * S1 * /S0 \\ & + S2 * /S1 * S0 \end{aligned}$$

RINR := S0 * /S1 * S2 * S3 * S4

RRF := S0 * /S1 * S2 * /S3 * /S4

PAL16L8 (A)

BREAK DETECT

PILOT BUTTERFLY PAL 8

REG NO 40200

(U5)

VERSION A

ROGER EDWARDS

NORSK DATA LTD, NEWBURY

2-JUN-86

NC FO F15 SGR CRY /WP MO M1 CO GND

NC /XFETCH C2 C3 C4 C1 /CBRK1 /CBRK2 /JMP VCC

IF (VCC) JMP = C4 * /C3 * /C2 * C1 * CO * /M1 * /MO * F15

+ C4 * /C3 * /C2 * C1 * CO * /M1 * MO * /F15

+ C4 * /C3 * /C2 * C1 * CO * M1 * /MO * FO

+ C4 * /C3 * /C2 * C1 * CO * M1 * MO * /FO

+ C4 * /C3 * C2 * C1 * /CO

+ C4 * /C3 * C2 * C1 * /M1

IF (VCC) XFETCH = C4 * /C3 * C2

+ C4 * /C3 * /C2 * C1 * /CO * M1 * MO * /WP

+ C4 * /C3 * /C2 * C1 * CO * /M1 * /MO * F15

+ C4 * /C3 * /C2 * C1 * CO * /M1 * MO * /F15

+ C4 * /C3 * /C2 * C1 * CO * M1 * /MO * FO

+ C4 * /C3 * /C2 * C1 * CO * M1 * MO * /FO

IF (VCC) CBRK2 = C4 * /C3 * C2 * /C1 * /CO * M1 * /MO * /CRY

+ C4 * /C3 * C2 * /C1 * /CO * M1 * MO * CRY

+ C4 * /C3 * C2 * /C1 * CO * /M1 * /MO * /F15

+ C4 * /C3 * C2 * /C1 * CO * /M1 * MO * F15

+ C4 * /C3 * C2 * /C1 * CO * M1 * /MO * /FO

+ C4 * /C3 * C2 * /C1 * CO * M1 * MO * FO

+ C4 * /C3 * /C2 * C1 * /CO * M1 * MO * WP

IF (VCC) CBRK1 = C4 * /C3 * /C2 * C1 * CO * /M1 * /MO * /F15

+ C4 * /C3 * /C2 * C1 * CO * /M1 * MO * F15

+ C4 * /C3 * /C2 * C1 * CO * M1 * /MO * /FO

+ C4 * /C3 * /C2 * C1 * CO * M1 * MO * FO

+ C4 * /C3 * C2 * /C1 * /CO * /M1 * /MO * /SGR

+ C4 * /C3 * C2 * /C1 * /CO * /M1 * MO * SGR

PAL16L8 (A)

CYCLE TYPE DECODE

REG NO 40201

ROGER EDWARDS

PILOT BUTTERFLY PAL 9

(U4)

VERSION A

NORSK DATA LTD, NEWBURY

2-JUN-86

C4 C3 C2 C1 C0 M1 M0 CMEN NC GND

LSHADOW /NC /EMAP /MREQ /WAIT2 /WAIT1 /FORM /COMSL SHORT VCC

```

IF (VCC) /SHORT =      /C3 * C2 * C1 * /C0 * /M0
                      + /C4 * C3 * /C2 * /C1 * /C0
                      + C4 * /C3 *      C1 *      M1
                      + C4 * /C3 * C2
                      + C4 * /C3 *      C1 * C0
                      + C4 *      C2 * C1 * C0 * M1
                      + C4 *      C2 * C1 * C0 * M0

```

```

IF (GND) COMSL = GND

```

```

IF (VCC) WAIT1 = /LSHADOW * C4 * /C3 * C1 * /M1
+ /LSHADOW * C4 * /C3 * C1 * M0
+ /LSHADOW * C4 * /C3 * C1 * C0
+ /LSHADOW * C4 * /C3 * C2
+ /LSHADOW * C4 * C3 * /C2
+ /LSHADOW * C4 * C3 * /C1

```

```

IF (VCC) WAIT2 = /LSHADOW * C4 * /C3 * C1 * /M1
+ /LSHADOW * C4 * /C3 * C1 * M0
+ /LSHADOW * C4 * /C3 * C1 * C0
+ /LSHADOW * C4 * /C3 * C2
+ /LSHADOW * C4 * C3 * /C2 * /C1
+ /LSHADOW * C4 * C3 * /C1 * /C0

```

```

IF (VCC) FORM = C4 * /C3 * /C2 * C1 * M1
+ C4 * /C3 * C1 * C0
+ C4 * /C3 * C2

```

```

IF (VCC) MREQ = C4 * /C3 * /C2 * C1 * /C0 * /M1
+ C4 * /C3 * /C2 * C1 * /C0 * M0
+ C4 * /C3 * C1 * C0
+ C4 * /C3 * C2
+ C4 * C3 * /C2
+ C4 * C3 * /C1

```

```

IF (VCC) EMAP = C4 * /C3 * /C2 * C1 * M1 * M0 * CMEN
+ C4 * /C3 * C1 * C0 * CMEN
+ C4 * /C3 * C2 * CMEN
+ C4 * /C3 * /C2 * C1 * /C0 * M1 * /M0

```

DESCRIPTION

CMEN changed to CC2 or CC3 to cover cycles e to p.

PAL16R6 (B)		ROGER EDWARDS
CYCLE LENGTH CONTROL	REG NO 40900	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 10	(U2) VERSION A	2-JUN-86

OSC SHORT /COMSL /FORM /WAIT1 /WAIT2 BRK /DLY1 /DLY0 GND
 /OE CACT /CACLK /TERM /CC3 /CC2 /CC1 /CC0 /EORF VCC

IF (VCC) EORF = /CC3 * CC2 * CC1 * CC0 * /BRK * /TERM
 + /CC3 * CC1 * /CC0 * /WAIT1 * /WAIT2 * /TERM

CC0 := /CC3 * /CC2 * /CC1 * /TERM
 + /CC3 * CC2 * CC1 * CC0 * /TERM
 + CC3 * CC2 * /CC1 * /TERM
 + CC3 * /CC2 * CC1 * /TERM
 + /CC3 * CC2 * CC1 * /CC0 * /TERM * /CACT
 + /CC3 * CC2 * /CC1 * CC0 * WAIT2 * CACT
 + BRK * /CC3 * CC2 * CC1

CC1 := /CC3 * /CC2 * CC0 * /TERM
 + CC1 * /CC0 * /TERM
 + CC3 * CC2 * CC0 * /TERM
 + /CC3 * CC2 * CC1 * CC0 * WAIT1 * /CACT * /TERM * /BRK

CC2 := /CC3 * CC1 * /CC0 * /TERM
 + CC2 * CC1 * CC0 * /TERM
 + CC2 * /CC1 * /TERM

CC3 := CC2 * /CC1 * /CC0 * /TERM
 + CC3 * CC0 * /TERM
 + CC3 * CC1 * /TERM

TERM := CC3 * CC2 * /CC1 * CC0 * SHORT * /BRK * /TERM
 + CC3 * /CC2 * CC1 * /CC0 * COMSL * /BRK
 + CC3 * /CC2 * /CC1 * /CC0
 + CC3 * /CC2 * /CC1 * CC0 * BRK

CACLK := /CC3 * /CC2 * CC1 * /CC0 * WAIT1 * /WAIT2
 + /CC3 * CC2 * /CC1 * /CC0 * WAIT1 * /WAIT2
 + CC3 * CC2 * CC1 * /CC0 * WAIT1 * /WAIT2

PAL16R4 (A)		ROGER EDWARDS
CONDITION CONTROL	REG NO 40600	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 11	(U7) VERSION A	2-JUN-86

CK NC NC /CSDLYO /CSDLY1 /MAP /ACOND /CSECOND /CSLOOP GND
/OE /MR CSDELAY /MDLY1 /NC /MDLYO /NC /DLYO /DLY1 VCC

IF (VCC) DLY1 = ACOND * CSECOND
+ ACOND * CSLOOP
+ MDLY1

IF (VCC) DLYO = MDLYO
+ CSDELAY * /DLY1

MDLYO := CSDLYO
+ MAP

MDLY1 := CSDLY1

PAL16L8 (A)		ROGER EDWARDS
CLOCK GENERATION	REG NO 40202	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 12	(U6) VERSION A	3-JUN-86

/EMAP /CC0 /CC1 /CC2 /CC3 /WAIT2 CACT /WRITE BRK GND
 /CBRK /TRAP /MACLK /MAP /MCLK /TERM TCLK /EMD NC VCC

IF (VCC) TRAP = BRK * /CBRK * CC1 * /CC0
 + BRK * /CBRK * CC3 * CC0
 + BRK * /CBRK * CC2

IF (VCC) /TCLK = BRK * /CBRK * /CC3 * /CC2 * CC1

IF (VCC) MCLK = TERM
 + WRITE * /CC3 * /CC2
 + WRITE * /CC3 * CC1 * /BRK
 + /CC3 * /CC2 * /CC1 * /CC0 (see note)

IF (VCC) MAP = EMAP * /CC3 * CC2 * CC0 * /CBRK * /MCLK * /TERM
 + EMAP * /CC3 * CC2 * /CC1 * /CBRK * /MCLK * /TERM
 + EMAP * CC3 * /CBRK * /MCLK * /TERM

IF (VCC) MACLK = TERM
 + MAP * /CC3 * CC2 * /CC1
 + BRK * /CBRK * /CC3 * CC2
 + /CC3 * /CC2 * /CC1 * /CC0 (see note)

IF (VCC) EMD = WAIT2 * EMD * CC2 * /CC1 * /BRK hold for reads
 + MAP * EMD * /CC2 * /CC3 hold for fetch
 + MAP * EMD * /BRK hold for fetch
 + CACT * /CC2 * /CC3 start term
 + CACT * /BRK start term

DESCRIPTION

EMD problem basic solution used. This might require changing.

TRAP - cycles d to o.

MCLK low during cycles g to j for write (WMAP) to shadow.

EMD changed - to include hold term for extending during read so catch clock on ALU and inverted - WAIT2 is reads and fetches and not LSHADOW.

MAP active for fo p cycles to cover all fetch.

MCLK & MACLK extended to cover nanocycle "a" as per RASK 16L84 2-9-85, but commented out until tried.

PAL16L8 (B)		ROGER EDWARDS
MMU CONTROL	REG NO 40300	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 13	(U18) VERSION B	29-AUG-86

LSHAD CAO DCUBLE /DVACC /WRITE /MREQ MCLK /FETCH /INTRQ GND
 /EMCL /EPMAP /VACC /EIPU /WMAP /VEX /EPTI /EPT /EIPL VCC

IF (VCC) EIPU = LSHAD * WRITE * DOUBLE
 + LSHAD * DOUBLE * CAO

IF (VCC) EIPL = LSHAD * WRITE
 + LSHAD * DOUBLE * CAO

IF (VCC) EPMAP = LSHAD * /WRITE
 + LSHAD * /DOUBLE
 + LSHAD * CAO
 + /DVACC * /LSHAD * MREQ

IF (VCC) EPT = /WRITE
 + /LSHAD
 + /DOUBLE
 + /CAO
 + EMCL

IF (VCC) EPTI = LSHAD * WRITE
 + LSHAD * /CAO
 + LSHAD * /DOUBLE * MREQ

IF (VCC) WMAP = LSHAD * WRITE * /MCLK

IF (VCC) VACC = /DVACC * /LSHAD * MREQ * /EMCL * /FETCH * /VEX
 + /DVACC * /LSHAD * MREQ * /EMCL * /INTRQ * /VEX

DESCRIPTION

B Iss: EMCL removed from EPTI and EPT, so MEM tests work in MACL.
 EMCL now redundant

PAL16R4 (A)

TRAP 1

REG NO 40500

ROGER EDWARDS

PILOT BUTTERFLY PAL 14

(U8)

VERSION A

NORSK DATA LTD, NEWBURY

2-JUN-86

TCLK /VACC /WRITE /IND /FETCH PT15 PT14 PT13 PT12 GND
 /OE /NC /NC /CD9 /CD8 /CD7 /CD6 /PVIOL /TRAP1 VCC

IF (VCC) TRAP1 = VACC * /PT15 * /PT14 * /PT13
 + VACC * WRITE * /IND * /FETCH * /PT15
 + VACC * /WRITE * IND * /FETCH * /PT14 * /PT13
 + VACC * /WRITE * /IND * FETCH * /PT13
 + VACC * /WRITE * /IND * /FETCH * /PT14
 + VACC * WRITE * PT15 * /PT12

IF (VCC) PVIOL = /PT15 * /PT14 * /PT13
 + WRITE * /IND * /FETCH * /PT15
 + /WRITE * IND * /FETCH * /PT14 * /PT13
 + /WRITE * /IND * FETCH * /PT13
 + /WRITE * /IND * /FETCH * /PT14

CD6 := PVIOL * PT15
 + PVIOL * PT14
 + PVIOL * PT13

CD7 := /PT15 * /PT14 * /PT13
 + /PVIOL * WRITE * PT15 * /PT12

CD8 := PVIOL

CD9 := VCC

PAL16R4 (A)

TRAP 2

REG NO 40502

ROGER EDWARDS

PILOT BUTTERFLY PAL 15

(U9)

VERSION D

NORSK DATA LTD, NEWBURY

22-AUG-86

TCLK /VACC /DPRIX PT11 PT10 PT9 PCR1 PCRO PT13 GND
 /OE /FETCH /NC /CD9 /CD8 /CD7 /CD6 /RIOK /TRAP2 VCC

```
IF (VCC) TRAP2 = VACC * DPRIX           ;S IV
+ VACC * /PT11                          ;page used
+ VACC * FETCH * PT13 * /RIOK           ;ring down
+ VACC * PT10 * /PCR1                   ;}
+ VACC * PT9 * /PCR1 * /PCRO            ;}ring violation
+ VACC * PT10 * PT9 * /PCRO             ;}
```

```
IF (VCC) RIOK = /PT10 * /PT9 * /PCR1 * /PCRO
+ /PT10 * PT9 * /PCR1 * PCRO
+ PT10 * /PT9 * PCR1 * /PCRO
+ PT10 * PT9 * PCR1 * PCRO
```

```
CD6 := PT10 * /PCR1 * PT11
+ PT9 * /PCR1 * /PCRO * PT11
+ PT10 * PT9 * /PCRO * PT11
+ /PT11
+ DPRIX * PT11 * RIOK
```

```
CD7 := /PT11
```

```
CD8 := PT10 * /PCR1 * PT11
+ PT9 * /PCR1 * /PCRO * PT11
+ PT10 * PT9 * /PCRO * PT11
+ FETCH * PT13 * /PT10 * PCR1 * PT11
+ FETCH * PT13 * /PT10 * /PT9 * PCRO * PT11
+ FETCH * PT13 * /PT9 * PCR1 * PCRO * PT11
+ DPRIX
```

```
CD9 := PT10 * /PCR1 * PT11
+ PT9 * /PCR1 * /PCRO * PT11
+ PT10 * PT9 * /PCRO * PT11
+ /PT11
+ FETCH * PT13 * /PT10 * PCR1
+ FETCH * PT13 * /PT10 * /PT9 * PCRO
+ FETCH * PT13 * /PT9 * PCR1 * PCRO
```

DESCRIPTION

RIOK removed from TRAP2 term 2 page used

B Issue: RIOK removed from pt11 terms in CD6,7 & 9

C Issue: PT11 added to CD8 to allow page used to have priority over ring down.

D Issue: PT11 added to ring viol terms in CD 6, 8 and 9.

PAL16R4 (A)		ROGER EDWARDS
TRAP 3	REG NO 40501	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 16	(U10) VERSION A	3-JUN-86

TCLK DSTOP /INTRQ NC DSTOP NC /FETCH NC PANPAN GND
/OE NC NC /CD9 /CD8 /CD7 /CD6 NC /TRAP3 VCC

IF (VCC) TRAP3 = FETCH * INTRQ

CD6 := FETCH * INTRQ * PANPAN * /DSTOP

CD7 := GND

CD8 := GND

CD9 := GND

DESCRIPTION

V5 - DSTOP will change vector from 16 to 17 to disable panel request while in single step.

PAL16L8 (A2)		ROGER EDWARDS
PC I/F CONTROL	REG NO 41600	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 17	(U139) VERSION B	17-JUL-86

PCA19 PCA18 PCA17 PCA16 PCA0 /MEMR MEMW /PCGNT /STAT GND
 /STRIKE /DISWHI /ALEX ENDOS ENWRT DLBTPC /EPCLO /EPCHI /DISWLO VCC

IF (VCC) ALEX = PCA19 * PCA18 * /PCA17 * PCA16 * MEMW * ENWRT
 + PCA19 * PCA18 * /PCA17 * PCA16 * MEMR
 + PCA19 * /PCA18 * /PCA17 * MEMW * ENDOS
 + PCA19 * /PCA18 * /PCA17 * MEMR * ENDOS

IF (VCC) EPCHI = PCA0 * MEMR
 + PCA0 * MEMW * PCGNT

IF (VCC) EPCLO = /PCA0 * MEMR
 + /PCA0 * MEMW * PCGNT

IF (VCC) /DLBTPC = /MEMR * PCGNT

IF (PCGNT) DISWLO = PCA0
 + /MEMW
 + STAT * /STRIKE

IF (PCGNT) DISWHI = /PCA0
 + /MEMW
 + STAT * /STRIKE

DESCRIPTION

Note: T160 is not used in this PAL.

PAL16L8 (A2)
 PC I/O ADDRESS CONTROL REG NO 41300 ROGER EDWARDS
 PILOT BUTTERFLY PAL 18 (U138) VERSION B NORSK DATA LTD, NEWBURY
 17-JUL-86

PCAO PCA1 PCA2 PCA3 PCA4 PCA5 PCA6 PCA7 PCA8 GND
 PCA9 /PCBUF /PCIOW /PCIOR PCAEN NC /ALEX /PCPRES /ENIO VCC

ENIO = /PCPRES
 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOW * /PCA2 * /PCA1 * PCA0 ;PCSIMF

 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOW * /PCA2 * PCA1 ;NVOPEN & PCMCL

 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOW * PCA2 ;LDOFF,G00FF,GOST & PCLETG0

 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOR * PCA2 * /PCA1 * PCA0 ;G00FF

 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOR * PCA2 * PCA1 * /PCA0 ;GOST

PCBUF = ALEX
 + /PCPRES
 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOW * /PCA2 * /PCA1 * PCA0 ;PCSIMF

 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOW * /PCA2 * PCA1 ;NVOPEN & PCMCL

 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOW * PCA2 ;LDOFF,G00FF,GOST & PCLETG0

 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOR * PCA2 * /PCA1 * PCA0 ;G00FF

 + /PCAEN * /PCA9 * PCA8 * /PCA7 * PCA6 * PCA5 * /PCA4 * /PCA3
 * PCIOR * PCA2 * PCA1 * /PCA0 ;GOST

DESCRIPTION

Base address is 160 hex.
 PCBUF added to enable PC I/O buffer during memory accesses and I/O in the range
 16X hex - JTG 9/12/85.

PAL20S10		ROGER EDWARDS
LOCAL BUS ADDRESS DECODER	REG NO 41800	NORSK DATA LTD., NEWBURY
PILOT BUTTERFLY PAL 19	(U186) VERSION B	17-JUL-86

```
"device declaration
    b20s1019 device    'P20S10'; " normal speed
```

```

"inputs
LBA13,LBA14,LBA15,LBA16,LBA17      pin 11,10,9,8,7;
LBA18,LBA19,LBA20,LBA21             pin 6,5,4,3;
LBA23,SCACTO,REFO                     pin 1,13,2;
ENDCS                                  pin 15;

```

```

"outputs
    PROM0,STAT0,IOMMO                pin 23,22,21;
    RAS01,RAS21,RAS11,RAS01         pin 14,19,18,17;
    DRAM0,INVAL0                     pin 16,20;

```

```
"constant declarations
    H,L,X      = 1,0,.X.;
    PROM,STAT,IOMM      :=!PROMO,!STATO,!IOMMO;
    RAS3,RAS2,RAS1,RAS0 :=RAS31,RAS21,RAS11,RAS01;
    DRAM,INVAL,SCACT :=!DRAMO,!INVALO,!SCACTO;
    REF        :=!REFO;
    enable DOS memory =ENDOS;
```

```
"define addresses
    Addr = [LBA23,X,LBA21,LBA20,LBA19,LBA18,LBA17,LBA16,
            LBA15,LBA14,LBA13,X,X,X,X,X,X,X,X,X,X,X,X];
```

equations

```
RASC = (Addr >= ^h000000) & (Addr <= ^h03BFFF)
& (Addr >= ^h80C000) & (Addr <= ^h80FFFF);

RAS1 = (Addr >= ^h040000) & (Addr <= ^h04FFFF) &
!(enable DOS memory & SCACT)
& (Addr >= ^h050000) & (Addr <= ^h07FFFF);

RAS2 = (Addr >= ^h080000) & (Addr <= ^h0BFFFF);

RAS3 = (Addr >= ^h0C0000) & (Addr <= ^h0FFFFF);

PROM = (Addr >= ^h800000) & (Addr <= ^h801FFF);

STAT = (Addr >= ^h808000) & (Addr <= ^h8087FF);
```

```

IOMM = (Addr >= `h80A000) & (Addr <= `h80BFFF);

DRAM = (Addr >= `h000000) & (Addr <= `h03BFFF) & !REF
      & (Addr >= `h040000) & (Addr <= `h04FFFF) & !REF &
      !(enable DOS memory & SCACT)
      & (Addr >= `h050000) & (Addr <= `h0FFFFF) & !REF
      & (Addr >= `h80C000) & (Addr <= `h80FFFF) & !REF;

INVAL = (Addr >= `h03C000) & (Addr <= `h03FFFF) & !REF
      & (Addr >= `h040000) & (Addr <= `h04FFFF) & !REF &
      enable DOS memory & SCACT
      & (Addr >= `h100000) & (Addr <= `h7FFFFF) & !REF
      & (Addr >= `h802000) & (Addr <= `h807FFF) & !REF
      & (Addr >= `h810000) & (Addr <= `hFFFFF) & !REF
      & (Addr >= `h808800) & (Addr <= `h809FFF) & !REF;

```

test vectors

```

([Addr, REFO, SCACT0] -> [RAS31, RAS21, RAS11, RAS01,
    PROMO, STATO, IOMMO, DRAMO, INVAL0])

```

note enable DOS memory not tested....

Address	R S	RAS	P S I D I
	E C	3 2 1 0	R T O R N
	F A		O A M A V
	C		M T M M A

```

[ `h000000, H, H] -> [L, L, L, H, H, H, H, L, H]; "bank 0 DRAM
[ `h000000, L, H] -> [L, L, L, H, H, H, H, H, H]; "bank 0 DRAM in REF
[ `h038000, H, H] -> [L, L, L, H, H, H, H, L, H];
[ `h035000, H, H] -> [L, L, L, L, H, H, H, H, L, H];
[ `h027000, H, H] -> [L, L, L, L, H, H, H, H, L, H];
[ `h010000, H, H] -> [L, L, L, L, H, H, H, H, L, H];
[ `h03C000, H, H] -> [L, L, L, L, L, H, H, H, H, L]; "don't exist
[ `h070000, H, H] -> [L, L, H, L, L, H, H, H, H, L, H]; "bank 1 DRAM
[ `h070000, L, H] -> [L, L, H, L, L, H, H, H, H, H, H]; "bank 1 DRAM in REF
[ `h087000, H, H] -> [L, H, L, L, L, H, H, H, H, L, H]; "bank 2 DRAM
[ `h087000, L, H] -> [L, H, L, L, L, H, H, H, H, H, H]; "bank 2 DRAM in REF
[ `h0FFFFF, H, H] -> [H, L, L, L, L, H, H, H, H, L, H]; "bank 3 DRAM
[ `h0FFFFF, L, H] -> [H, L, L, L, L, H, H, H, H, H, H]; "bank 3 DRAM in REF
[ `h1F0000, H, H] -> [L, L, L, L, L, H, H, H, H, H, L]; "don't exist
[ `h1F0000, L, H] -> [L, L, L, L, L, H, H, H, H, H, H]; "no inval as REF
[ `h20F000, H, H] -> [L, L, L, L, L, L, H, H, H, H, L, H];
[ `h800A00, H, H] -> [L, L, L, L, L, L, H, H, H, H, H, H]; "PROM
[ `h808000, H, H] -> [L, L, L, L, L, H, L, H, H, H, H]; "STAT
[ `h80A000, H, L] -> [L, L, L, L, L, H, H, L, H, H, H]; "mapped I/O
[ `h80F000, H, H] -> [L, L, L, L, H, H, H, H, H, L, H]; "bank 0 addr conv
[ `h81F000, H, H] -> [L, L, L, L, L, H, H, H, H, H, L]; "don't exist
[ `h82000F, H, H] -> [L, L, L, L, L, L, H, H, H, H, H, L];
[ `h84AF00, H, H] -> [L, L, L, L, L, H, H, H, H, H, L];
[ `h8800F0, H, H] -> [L, L, L, L, L, H, H, H, H, H, L];

```


PAL16R8 (B)		ROGER EDWARDS
LOCAL BUS ADDRESS ARBITER	REG NO 41700	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 20	(U148) VERSION B	24-JUL-86

```

"device declaration
    bf16r820 device 'P16R8'; " B PAL

"inputs
    OSC1,OE0                pin 1,11;
    PCRQ0,RRQ1,RFC1         pin 2,3,4;
    SEMRQ0,IORQ0,MCRQ1      pin 5,6,7;
    MDRY0,DMINHO            pin 9,8;

"outputs
    SCACT0,PCGNT0           pin 18,17;
    IOBGNT0,REFO            pin 16,15;
    MSTART0                 pin 19;

"feedback terms
    lst1,lst2                pin 14,13;

"constant declarations
    H,L,X,C = 1,0,.X.,.C.;
    rfclk,Srq,dry,minh       = RFC1,!SEMRQ0,!MDRY0,!DMINHO;
    Prq,Rrq,Irq,Crq         = !PCRQ0,RRQ1,!IORQ0,MCRQ1;
    Cat,Pgt,Igt,Ref         = !SCACT0,!PCGNT0,!IOBGNT0,!REFO;
    mstart                   = !MSTART0;

"define last accesses
    LST = [lst1,lst2];
    none,cpu,pc,iob = 3,1,0,2;

"macros
    HoldOn macro (s) {?s & !dry & !minh};
    OKforRq macro (s) {?s & !dry & !minh};
    HiPriRq macro {(Rrq & !rfclk)} ; "refresh has highest priority
    HoldOnRef macro {(Ref & !dry)};

equations

    Ref := HoldOnRef
        £ OKforRq(Rrq) &
          !Cat & !Pgt & !Igt & "nothing else active
          (!rfclk & !Srq £ "late - hi priority
          !Crq & !Irq & !Prq) "early - only if no other reqs
        £ Rrq & minh; "keep going if accesses stopped

    Cat := HoldOn(Cat)
        £ OKforRq(Crq) &
          !Ref & !Pgt & !Igt & "nothing else active
          (!HiPriRq &
          (!Irq & (!Prq £ (LST == pc)) £
          (LST == iob) £ (LST == none)) £
          Srq);

```

```

Pgt := HoldOn(Pgt)
      £ OKforRq(Prq) &
        !Ref & !Cat & !Igt &      "nothing else active
        !HiPriRq &
        (!Crq & (!Irq £ (LST == iob) £ (LST == none)) £
        (LST == cpu));

Igt := HoldOn(Igt)
      £ OKforRq(Irq) &
        !Ref & !Cat & !Pgt &      "nothing else active
        !HiPriRq &
        (!Prq & (!Crq £ (LST == pc) £ (LST == cpu))£
        (LST == pc));

mstart := Ref £ Cat £ Pgt £ Igt;

!lst2 :=      Pgt & dry
      £ Igt & dry
      £ !lst2 & !Cat & !Pgt & !Igt & !Ref;  "lock on

!lst1 :=      Pgt & dry
      £ Cat & dry
      £ !lst1 & !Cat & !Pgt & !Igt & !Ref;  "lock on

```

test vectors

```

([OSC1,minh,Srq,Crq,Prq,Irq,Rrq,rclk,dry] ->
 [Cat,Pgt,Igt,Ref,LST,mstart])
"  c m S C P I R r d      C P I R      m
"  l i r r r r r r f r      a g g e LST s
"  k n q q q q q c y      t t t f      t
"      h              k              a

[ C,H,X,X,X,X,L,X,X ]->[L,L,L,L, X ,L ]; "1
[ C,H,H,H,H,H,H,H,L ]->[L,L,L,H,none,L ];
[ C,H,H,H,H,H,L,L,L ]->[L,L,L,H,none,H ];
[ C,H,L,L,L,L,L,L,H ]->[L,L,L,L,none,H ]; "4
[ C,L,L,H,L,L,L,H,L ]->[H,L,L,L,none,L ];
[ C,L,L,L,L,L,L,H,L ]->[H,L,L,L,none,H ];
[ C,L,L,L,L,L,L,L,H ]->[L,L,L,L,cpu ,H ]; "7
[ C,L,L,H,H,H,H,H,L ]->[L,H,L,L,cpu ,L ]; "
[ C,L,L,H,L,H,H,H,L ]->[L,H,L,L,none,H ]; "
[ C,H,L,H,L,H,L,H,H ]->[L,L,L,L,pc ,H ]; "10
[ C,H,L,H,L,H,L,H,L ]->[L,L,L,L,pc ,L ]; "
[ C,L,L,H,L,H,H,H,L ]->[L,L,H,L,pc ,L ]; "
[ C,L,L,L,L,L,L,L,H,L ]->[L,L,H,L,none,H ]; "13
[ C,L,L,H,H,H,H,L,H ]->[L,L,L,L,iob ,H ]; "
[ C,L,L,H,H,H,H,L,L ]->[L,L,L,H,iob ,L ]; "
[ C,L,L,H,H,H,L,H,L ]->[L,L,L,H,none,H ]; "16
[ C,L,L,H,H,H,L,H,H ]->[L,L,L,L,none,H ]; "
[ C,L,L,L,H,H,L,H,L ]->[L,H,L,L,none,L ]; "
[ C,L,L,L,L,H,L,H,L ]->[L,H,L,L,none,H ]; "19
[ C,L,L,L,L,H,L,H,H ]->[L,L,L,L,pc ,H ]; "
[ C,L,L,L,L,H,L,H,L ]->[L,L,H,L,pc ,L ]; "
[ C,L,L,L,L,L,L,H,H ]->[L,L,L,L,iob ,H ]; "22
[ C,L,L,H,L,H,L,H,L ]->[H,L,L,L,iob ,L ]; "
[ C,L,L,L,L,L,L,H,H ]->[L,L,L,L,cpu ,H ]; "
[ C,L,L,H,H,L,L,H,L ]->[L,H,L,L,cpu ,L ]; "25
[ C,L,L,L,L,L,L,H,H ]->[L,L,L,L,pc ,H ]; "
[ C,L,L,H,H,L,L,H,L ]->[H,L,L,L,pc ,L ]; "
[ C,L,L,L,L,L,L,H,H ]->[L,L,L,L,cpu ,H ]; "28
[ C,L,L,L,H,L,L,H,L ]->[L,H,L,L,cpu ,L ]; "
[ C,L,L,L,L,L,L,H,H ]->[L,L,L,L,pc ,H ]; "
[ C,L,L,H,H,H,L,H,L ]->[L,L,H,L,pc ,L ]; "31
[ C,L,L,L,L,H,L,H,H ]->[L,L,L,L,iob ,H ]; "
[ C,L,L,L,L,H,L,H,L ]->[L,L,H,L,iob ,L ]; "
[ C,L,L,L,L,L,L,H,H ]->[L,L,L,L,iob ,H ]; "34
[ C,L,L,H,H,H,L,H,L ]->[H,L,L,L,iob ,L ]; "
[ C,L,L,H,L,L,L,H,H ]->[L,L,L,L,cpu ,H ]; "
[ C,L,L,H,L,L,L,H,L ]->[H,L,L,L,cpu ,L ]; "37
[ C,L,L,L,L,L,L,H,L ]->[H,L,L,L,none,H ]; "
[ C,L,L,L,L,L,L,H,H ]->[L,L,L,L,cpu ,H ]; "
[ C,L,L,L,L,L,L,H,L ]->[L,L,L,L,cpu ,L ]; "40

```

PAL16L8 (A)		ROGER EDWARDS
LOCAL BUS CONTROL	REG NO 41400	NORSK DATA LTD, NEWBURY
PILOT BUTTERFLY PAL 21	(U149) VERSION D	09-FEB-87

ECREQ /APR /WRITE /EPEA /SCACT ENWRT /CACT BDRY ERR GND
 /INVAL /SPEA T60 BLOCK /EDOFF /PCERR /CWRITE /PARERR CRQ VCC

IF (VCC) CWRITE = WRITE * ECREQ
 + CWRITE * /BDRY
 + CWRITE * SCACT
 + CWRITE * CACT

IF (VCC) /CRQ = ECREQ
 + CWRITE * /CACT

IF (VCC) SPEA = /APR + BLOCK + EPEA

IF (VCC) PARERR = ERR * /T60 * BDRY * /INVAL * /EDOFF

IF (VCC) /BLOCK = /BLOCK * T60
 + /BLOCK * /BDRY
 + /BLOCK * /INVAL * /PARERR
 + /BLOCK * EDOFF
 + EPEA

IF (VCC) PCERR = ERR * ENWRT
 + ERR * EDOFF

DESCRIPTION

CACT and SCACT inverted in CWRITE to bring in line with RASK and to extend CWRITE in order to remove glitches on CD bus during STD instruction

BLOCK inverted & used to generate SPES via D170 externally. PARERR cleaned up using T60 and BDRY internally and parity is not generated when an invalid address is signalled.

Note: on Pilot machines, pin 6 is connected to DRY but not used.

B issue - APR inverted.

C issue - /BDRY hold on TERM removed from CREQ (as in RASK) to prevent spurious EMDs.

D issue - PCERR is parity error to PC. It occurs if writes to PC are allowed or if an access to enabled MS-DOS-only memory occurs. Overcomes power-up when window for Seg "D" should look like PROM and not return parity errors. It must still be gated with PCGNT outside PAL. Parity errors in MS-DOS-only area are not reported to ND (PARERR & BLOCK).



SEND US YOUR COMMENTS!

Are you frustrated because of unclear information in our manuals? Do you have trouble finding things?

Please let us know if you:

- find errors
- cannot understand information
- cannot find information
- find needless information.

Do you think we could improve our manuals by rearranging the contents? You could also tell us if you like the manual.

Send to:

Norsk Data A.S
Documentation Department
P.O. Box 25 BOGERUD
N - 0621 OSLO 6 - Norway

NOTE!

This form is primarily for documentation errors. Software and system errors should be reported on Customer System Reports.

Manual Name: _____

Manual number: _____

Which version of the product are you using? _____

What problems do you have? (use extra pages if needed) _____

Do you have suggestions for improving this manual? _____

Your name: _____ Date: _____

Company: _____ Position: _____

Address: _____

What are you using this manual for? _____