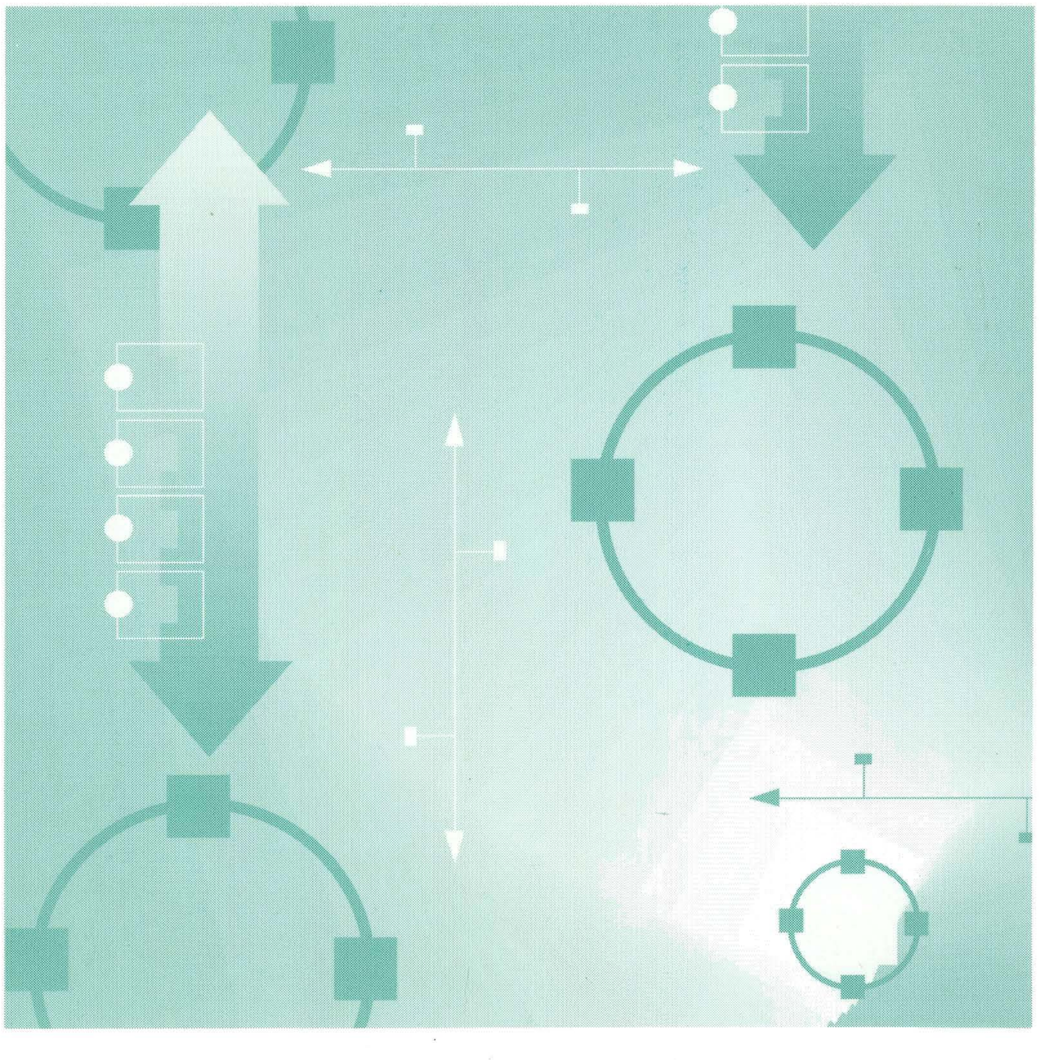
LAN Technical Reference

SC30-3587-00

IEEE 802.2 and NETBIOS Application Program Interfaces



IBM

LAN Technical Reference

SC30-3587-00

IEEE 802.2 and NETBIOS Application Program Interfaces

Note

Before using this information and the products it supports, be sure to read the general information under "Notices" on page xv.

First Edition (December 1993)

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

Forms for readers' comments appear at the front and back of this publication. If the forms have been removed, address your comments to:

Dept. E02
Design & Information Development
International Business Machines Corporation
PO Box 12195
Research Triangle Park, NC 27709-9990
USA

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Notices xv
Trademarks xv

Preface xvii
How This Manual Is Organized xvii
Related IBM Publications xviii
Guide to Information About the TCP/IP Interface xix
General Publications xix
Programming Publications xix
Guide to Information About the Open Data-Link Interface (ODI) xix
IBM Local Area Network OEMI xx

How This Manual Differs from the Fourth Edition of the *IBM Local Area Network Technical Reference* xxi

Chapter 1. LAN Overview and Interfaces 1-1
About This Chapter 1-2
Drivers and Programs That Provide the Application Program Interfaces 1-2
NDIS and non-NDIS Adapters 1-2
IBM Programs to Support LAN Adapters 1-2
Where to Find Information About IBM Adapter Interfaces 1-3
Introduction to the Networks 1-3
LAN Networks 1-3
The IBM Token-Ring Network 1-3
The IBM PC Network (Broadband) 1-4
The IBM PC Network (Baseband) 1-4
Ethernet Support 1-4
Related Software 1-4
OS/2 Communications Manager (OS/2 EE 1.3) 1-5
IEEE 802.2 Application Program Interface in OS/2 1-5
Differences Between the Dynamic Link Routine Interface and the Device Driver Interface 1-6
Why Use IEEE 802.2 or NETBIOS? 1-7
IBM LAN Application Program Interfaces 1-7
IEEE 802.2 Interface 1-9
The Direct Interface 1-9
The DLC Interface 1-9
The APPC/PC Interface 1-9
NETBIOS 1-10
Token-Ring Network Frame Definition 1-10
Routing Information Field 1-12
Routing Control Field 1-12
Route Designator Fields 1-15
Frame Format on the PC Network 1-16
Ethernet IEEE 802.3 Frame Format 1-17
Ethernet DIX Version 2.0 Frame Format 1-19

Chapter 2. Programming Conventions for the IEEE 802.2 Interface 2-1
About This Chapter 2-2
IBM Program Products for LAN Adapter Support 2-2

IEEE 802.2 Programming Conventions with DOS 2-3
Local Area Network Support Program (CCB1) Calling Conventions 2-3
Local Area Network Support Program (CCB1) Command Completion 2-3
Local Area Network Support Program (CCB1) Control Blocks 2-6
Programming Conventions with OS/2 2-7
The OS/2 DLR Interface 2-7
The OS/2 DD Interface 2-12
Control Blocks for All CCBs 2-19
CCB Field Explanations 2-19
Addressing 2-26
Adapter Addresses 2-26
Stations, SAPs, and IDs 2-27
SAP Assignments 2-29
DLC 2-30
Types of Service 2-31
Command Sequences 2-31
Link Station States 2-33
Timers 2-35
Guidelines for Selecting Parameter Values 2-36
Transmitting, Receiving, and Buffers 2-39
Buffer Pools 2-40
Receive Buffers 2-41
Transmit Buffers 2-46

Chapter 3. The Command Control Blocks 3-1
About This Chapter 3-3
Command Descriptions 3-4
BUFFER.FREE 3-4
BUFFER.GET 3-5
DIR.CLOSE.ADAPTER 3-6
DIR.CLOSE.DIRECT 3-8
DIR.DEFINE.MIF.ENVIRONMENT 3-9
DIR.INITIALIZE 3-11
DIR.INTERRUPT 3-16
DIR.MODIFY.OPEN.PARMS 3-16
DIR.OPEN.ADAPTER 3-17
DIR.OPEN.DIRECT 3-33
DIR.READ.LOG 3-34
DIR.RESET.MULT.GROUP.ADDRESS 3-37
DIR.RESTORE.OPEN.PARMS 3-37
DIR.SET.EXCEPTION.FLAGS 3-38
DIR.SET.FUNCTIONAL.ADDRESS 3-39
DIR.SET.GROUP.ADDRESS 3-41
DIR.SET.MULT.GROUP.ADDRESS 3-41
DIR.SET.USER.APPENDAGE 3-43
DIR.STATUS 3-44
DIR.TIMER.CANCEL 3-51
DIR.TIMER.CANCEL.GROUP 3-51
DIR.TIMER.SET 3-52
DLC.CLOSE.SAP 3-53
DLC.CLOSE.STATION 3-54
DLC.CONNECT.STATION 3-55
DLC.FLOW.CONTROL 3-56
DLC.MODIFY 3-57

DLC.OPEN.SAP 3-60
DLC.OPEN.STATION 3-66
DLC.REALLOCATE 3-70
DLC.RESET 3-71
DLC.SET.THRESHOLD 3-73
DLC.STATISTICS 3-74
PDT.TRACE.OFF 3-76
PDT.TRACE.ON 3-77
PURGE.RESOURCES 3-85
READ 3-86
READ.CANCEL 3-94
RECEIVE 3-95
RECEIVE.CANCEL 3-102
RECEIVE.MODIFY 3-103
TRANSMIT.DIR.FRAME 3-107
TRANSMIT.I.FRAME 3-108
TRANSMIT.TEST.CMD 3-109
TRANSMIT.UI.FRAME 3-109
TRANSMIT.XID.CMD 3-109
TRANSMIT.XID.RESP.FINAL 3-110
TRANSMIT.XID.RESP.NOT.FINAL 3-110

Chapter 4. NETBIOS 4-1
About This Chapter 4-2
NETBIOS Overview 4-2
The Network Control Block 4-3
NCB Field Explanations 4-3
NETBIOS Operational Parameters 4-8
Local Area Network Support Program Old Parameters 4-8
Local Area Network Support Program New Parameters 4-13
NETBIOS 3.0 (OS/2 EE) 4-13
NETBIOS 4.0 4-13
NETBIOS DLC Timers 4-14
NETBIOS Calling Conventions Using the Local Area Network Support Program 4-14
NETBIOS Calling Conventions Using the Dynamic Link Routine Interface 4-14
NETBIOS Calling Conventions Using the Device Driver Interface 4-14
NCB Completion with Wait Type Commands 4-15
NCB Completion with No-Wait Type Commands 4-15
NETBIOS Command Descriptions 4-15
NCB.ADD.GROUP.NAME 4-15
NCB.ADD.NAME 4-16
NCB.CALL 4-16
NCB.CANCEL 4-17
NCB.CHAIN.SEND 4-18
NCB.CHAIN.SEND.NO.ACK 4-19
NCB.DELETE.NAME 4-20
NCB.FIND.NAME 4-21
NCB.HANG.UP 4-22
NCB.LAN.STATUS.ALERT 4-23
NCB.LISTEN 4-24
NCB.RECEIVE 4-25
NCB.RECEIVE.ANY 4-26
NCB.RECEIVE.BROADCAST.DATAGRAM 4-27

NCB.RECEIVE.DATAGRAM 4-27
NCB.RESET 4-28
NCB.SEND 4-32
NCB.SEND.BROADCAST.DATAGRAM 4-33
NCB.SEND.DATAGRAM 4-34
NCB.SEND.NO.ACK 4-34
NCB.SESSION.STATUS 4-35
NCB.STATUS 4-37
NCB.TRACE 4-41
NCB.UNLINK 4-45

Chapter 5. The NETBIOS Frames Protocol 5-1
About This Chapter 5-3
Assumptions 5-3
Related Documents 5-3
Terms and Definitions 5-3
NETBIOS Commands 5-4
General Information on Remote Name Directory 5-4
New NCB Commands 5-5
Header Format Overview 5-5
NETBIOS Header 5-6
NETBIOS Frame Summary 5-7
NETBIOS Frame Formats and Descriptions 5-11
ADD_GROUP_NAME_QUERY 5-11
ADD_NAME_QUERY 5-12
NAME_IN_CONFLICT 5-13
STATUS_QUERY 5-14
TERMINATE_TRACE 5-16
DATAGRAM 5-16
DATAGRAM_BROADCAST 5-17
NAME_QUERY 5-18
ADD_NAME_RESPONSE 5-20
NAME_RECOGNIZED 5-21
STATUS_RESPONSE 5-22
TERMINATE_TRACE 5-24
DATA_ACK 5-24
DATA_FIRST_MIDDLE 5-25
DATA_ONLY_LAST 5-28
SESSION_CONFIRM 5-30
SESSION_END 5-32
SESSION_INITIALIZE 5-33
NO_RECEIVE 5-34
RECEIVE_OUTSTANDING 5-36
RECEIVE_CONTINUE 5-37
SESSION_ALIVE 5-38
NETBIOS Protocol Examples Without RND 5-39
Name Management Examples 5-40
Remote Adapter Status Examples 5-43
Session Establishment Examples 5-46
Session Data Transfer Examples 5-49
NETBIOS Protocol Examples with RND 5-53
Session Establishment RND Examples 5-57
NETBIOS Protocol SEND.NO.ACK Examples 5-59
Session Establishment Examples 5-60

Session Data Transfer Examples 5-61

Chapter 6. Support of NDIS Adapters Using IBM OS/2 LAN Adapter and Protocol Support 6-1
LAN Adapter and Protocol Support Overview 6-2
LAN Adapters Supported 6-3
Programming Information—IEEE 802.2 API 6-3
Dynamic Link Interface 6-3
Device Driver Interface 6-3
Memory Restriction 6-3
Programming Information—NETBIOS API 6-4
No-Wait Command and Post Routines 6-4
Dynamic Link Interface 6-4
Device Driver Interface 6-5
Memory Restriction 6-5
Resource Information 6-5
NCB Reserve Field Change 6-5
NCB.STATUS Command Extension 6-5
Piggybacked Acknowledgment Behavior 6-6
Differences and Restrictions 6-6
Message Logging Facility 6-9
Operating System/2 Trace Facility 6-10
New Tracing Parameters for the NETBIOS and IEEE 802.2 APIs 6-11
Minor Codes for IEEE 802.2 Traces 6-13
PROTOCOL.INI TRACE Parameter—IEEE 802.2 6-15
Minor Codes for NETBIOS Traces 6-17
PROTOCOL.INI OS2TRACEMASK Parameter—NETBIOS 6-19
Tracing for OS/2 2.0 Virtual DOS LAN Support 6-20

Appendix A. Valid Commands A-1
DLC and Direct Interface Commands A-1
NETBIOS Commands A-3

Appendix B. Return Codes B-1
About This Appendix B-2
CCB Return Codes Listed by Interface B-2
CCB Return Codes Listed by Command B-5
CCB Return Codes Cause and Action B-14
DLC Status Codes B-28
DLC Status Table B-29
DLC Status Codes B-30
Suggested Actions in Response to DLC Status B-31
NCB Return Codes B-32
NCB Return Codes Listed by Command B-34
NCB Return Codes Cause and Action B-35
Adapter Status Parameter Table B-44
Frame Status B-46
Exception Indications B-46
Adapter Check B-46
Adapter Check for CCB1 . . . . B-46
Adapter Check for CCB2 . . . . B-47
Adapter Check for CCB3 . . . . B-49
Token-Ring Network Adapter Check Reason Codes for All CCBs B-50
PC Network and Ethernet Adapter Check Reason Codes for All CCBs B-52

Network Status B-52
Network Status for CCB1 . . . . B-52
Network Status for CCB2 . . . . B-52
Network Status for CCB3 . . . . B-54
Token-Ring Network Status Codes for All CCBs B-55
PC Network and Ethernet Status Codes for All CCBs B-56
Bring-Up Errors for All CCBs B-57
Token-Ring Network Adapter Open Errors for All CCBs B-58
Open Error Codes for All CCBs B-58
Suggested Actions in Response to Open Errors B-59
PC Network and Ethernet Adapter Open Errors for All CCBs B-63
PC System Detected Errors B-63
PC System Detected Errors for CCB1 . . . . B-63
PC System Detected Errors for CCB2 . . . . B-64
PC System Detected Errors for CCB3 . . . . B-67
System Action Exceptions for OS/2 EE 1.3 . . . . B-69
System Action Exceptions for CCB2 . . . . B-69
System Action Exceptions for CCB3 . . . . B-72

Appendix C. Local Area Network Sample Program Listings Diskette . . C-1

Appendix D. The Local Area Network Support Program Interrupt Arbitrator D-1
About This Appendix D-1
The Local Area Network Support Program Interrupt Arbitrator (DXMA0MOD.SYS) D-1
Registration Process Overview D-2

Appendix E. Operating System/2 Extended Edition Information E-1
About This Appendix E-2
CONFIG.SYS Commands E-2
Adapter Parameters E-3
DLC Parameters E-7
OS/2 EE NETBIOS Parameters E-11
OS/2 EE Trace Facility E-11
Trace Code Definition E-12
Trace Entry Format E-12
Trace Code Formats E-14
OS/2 EE NETBIOS Trace Facility E-18

Appendix F. NDIS Overview F-1
About This Appendix F-1
Description of NDIS F-1

List of Abbreviations X-1

Glossary X-3

Index X-19

Figures

1-1. DOS IEEE 802.2 and NETBIOS Interfaces 1-8
1-2. OS/2 EE Communications Manager Interfaces 1-8
1-3. Token-Ring Network Frame Format 1-11
1-4. Routing Information Field 1-12
1-5. Routing Control Field 1-12
1-6. Route Designator Field 1-15
1-7. PC Network Frame Format 1-16
1-8. IEEE 802.3 Frame Format 1-17
1-9. DIX Version 2.0 Frame Format 1-19
2-1. Universally Administered Adapter Address 2-26
2-2. Ethernet Universally Administered Adapter Address 2-27
2-3. Locally Administered Adapter Address 2-27
2-4. SAPs and Link Stations 2-30
2-5. Receive Buffer Formats 2-42
2-6. Transmit Buffers 2-48
2-7. MAC Frame 2-49
2-8. Non-MAC I Frame 2-49
2-9. Other non-MAC Frame 2-50
3-1. Byte 0: Adapter Number (0/1) and Activation Reason Code 3-82
3-2. Timer Entry 3-83
3-3. Initialize Entry 3-83
3-4. CRB Entry 3-83
3-5. ARB Entry 3-84
3-6. FCB Entry 3-84
3-7. TXCB Entry 3-84
4-1. Trace Entry Format Example 4-42
5-1. NETBIOS Frame Example 5-5
5-2. General Format 5-6
5-3. NETBIOS Non-Session Frame Header (DLC UI-Frame) 5-6
5-4. NETBIOS Session Frame Header (DLC I-Frame) 5-6
5-5. Add a Name to the Network Command Sequence 5-40
5-6. Name Already On Network Command Sequence 5-41
5-7. Receive Multiple Responses Command Sequence 5-42
5-8. Remote Adapter Status for a Name That Is Not on the Network Command Sequence 5-43
5-9. Remote Adapter Status for a Name that Is on the Network Command Sequence 5-44
5-10. Remote Adapter Status Data: Segmentation Command Sequence . 5-45
5-11. Call a Name: Name Not on Network Command Sequence 5-46
5-12. Call a Name: Name on Network but No Listen Command Sequence 5-47
5-13. Call a Name: Name Found—Start Session Command Sequence . . 5-48
5-14. Send Session Data: One Send and One Receive Command Sequence 5-49
5-15. Send Session Data: One Send and Multiple Receives Command Sequence 5-50
5-16. Send Session Data: Segmentation and One Receive Command Sequence 5-51
5-17. Send Session Data: Segmentation and Multiple Receives Command Sequence 5-52

5-18. Remote Adapter Status for a Name Not on Network or RND Command Sequence . . . 5-54
5-19. Remote Adapter Status for a Name Not in RND but on the Network Command Sequence . . . 5-55
5-20. Remote Adapter Status for a Remote Name That Is in RND Command Sequence . . . 5-56
5-21. Call a Name: Name Not on Network or RND Command Sequence 5-57
5-22. Call a Name: Name Not in RND but Is on Network—Start Session Command Sequence . . . 5-58
5-23. Call a Name: Name Found—Start Session Command Sequence . . 5-60
5-24. Send Session Data: One SEND.NO.ACK and No RECEIVE Command Sequence . . . 5-61
5-25. Send Session Data: One SEND.NO.ACK and One RECEIVE Command Sequence . . . 5-62
5-26. Send Session Data: One SEND.NO.ACK and Multiple RECEIVEs Command Sequence . . . 5-63
6-1. LAN Adapter and Protocol Support Diagram . . . 6-2
6-2. Sample LANTRAN.LOG . . . 6-9
E-1. CCB Trace Entry . . . E-14
E-2. CCB Completion Trace Entry . . . E-15
E-3. Receive Completion Trace Entry . . . E-15
E-4. Status Exception Completion Trace Entry . . . E-16
E-5. Interrupt Received Trace Entry . . . E-17
E-6. Interrupt Error Trace Entry . . . E-18
F-1. Major Components in the NDIS Environment . . . F-1

Tables

2-1. CCB1 Command Control Block . . . 2-6
2-2. CCB2 Command Control Block . . . 2-12
2-3. CCB3 Command Control Block . . . 2-18
2-4. Command Completion Appendage Information Table . . . 2-22
2-5. Start Command Sequence for CCB1 and CCB3 . . . 2-31
2-6. Start Command Sequence for CCB2 . . . 2-32
2-7. End Command Sequence . . . 2-32
2-8. DLC Parameters . . . 2-37
2-9. Maximum I-field Length for Network Adapters . . . 2-38
2-10. Buffer 1: Option = Not Contiguous MAC/DATA . . . 2-42
2-11. Buffer 1: Option = Contiguous MAC/DATA . . . 2-43
2-12. Buffer 2 and Subsequent Buffers . . . 2-46
2-13. Transmit Buffers (XMIT_QUEUE_ONE and XMIT_QUEUE_TWO) . 2-48
2-14. Transmit Buffer Size in Bytes . . . 2-50
3-1. CCB Parameter Table for BUFFER.FREE . . . 3-4
3-2. CCB Parameter Table for BUFFER.GET . . . 3-5
3-3. Appendage Instruction Sequence . . . 3-9
3-4. CCB Parameter Table for DIR.DEFINE.MIF.ENVIRONMENT . . . 3-10
3-5. CCB Parameter Table for DIR.INITIALIZE . . . 3-12
3-6. CCB Parameter Table for DIR.MODIFY.OPEN.PARMS . . . 3-16
3-7. CCB Parameter Table for DIR.OPEN.ADAPTER . . . 3-20
3-8. Adapter Parms Open Parameters . . . 3-21
3-9. IBM Token-Ring Network Adapter OPEN_OPTIONS . . . 3-23
3-10. Product ID Field . . . 3-27
3-11. Direct Parms Open Parameters for CCB1 . . . 3-28
3-12. DLC Parms Open Parameters for All CCBs . . . 3-31
3-13. CCB Parameter Table for DIR.OPEN.DIRECT . . . 3-33
3-14. CCB Parameter Table for DIR.READ.LOG . . . 3-34
3-15. Log Formats for the Adapter Log . . . 3-36
3-16. Log Formats for the Direct Interface Log. . . . 3-36
3-17. DIR.RESET.MULT.GROUP.ADDRESS Parameter Table for CCB1 3-37
3-18. CCB Parameter Table for DIR.SET.EXCEPTION.FLAGS . . . 3-38
3-19. DIR.SET.MULT.GROUP.ADDRESS Parameter Table for CCB1 . . 3-42
3-20. CCB Parameter Table for DIR.SET.USER.APPENDAGE . . . 3-43
3-21. CCB Parameter Table for DIR.STATUS . . . 3-44
3-22. MICROCODE_LEVEL Fields . . . 3-47
3-23. Extended Status Table . . . 3-49
3-24. GROUP_ADDRESS_LIST . . . 3-50
3-25. CCB Parameter Table for DLC.CONNECT.STATION . . . 3-55
3-26. CCB Parameter Table for DLC.MODIFY . . . 3-57
3-27. CCB Parameter Table for DLC.OPEN.SAP . . . 3-61
3-28. CCB Parameter Table for DLC.OPEN.STATION . . . 3-67
3-29. CCB Parameter Table for DLC.REALLOCATE . . . 3-70
3-30. CCB Parameter Table for DLC.SET.THRESHOLD Command and Paramaters are for CCB2 only . . . 3-73
3-31. CCB Parameter Table for DLC.STATISTICS . . . 3-74
3-32. Log Formats for the SAP Log . . . 3-75
3-33. Log Formats for the Link Station Log . . . 3-76
3-34. CCB Parameter Table for PDT.TRACE.ON . . . 3-78
3-35. CCB Trace Entry . . . 3-79

3-36. Byte 1: Flags 3-79
3-37. Adapter Interrupt Trace Entry (Except Timer) 3-80
3-38. Byte 0: Interrupt Status Register Processor (ISRP) Even 3-80
3-39. Byte 1: Interrupt Status Register Processor (ISRP) Odd 3-80
3-40. Adapter Timer Interrupt Trace Entry 3-80
3-41. NCB Trace Entry 3-81
3-42. CCB Trace Entry 3-81
3-43. Byte 1: Flags 3-82
3-44. Interrupt Trace Entry 3-82
3-45. NCB Trace Entry 3-85
3-46. CCB Parameter Table for READ 3-87
3-47. Posted CCB Fields for Each Event 3-90
3-48. CCB Parameter Table for RECEIVE 3-98
3-49. CCB Parameter Table for RECEIVE.MODIFY 3-104
3-50. CCB Parameter Table for TRANSMIT Commands 3-111
4-1. Network Control Block (NCB) 4-3
4-2. NCB_RESERVE for the Local Area Network Support Program 4-8
4-3. NCB_RESERVE for OS/2 EE 4-8
4-4. Local Area Network Support Program Old Parameters 4-9
4-5. Data Areas Returned for the NCB.FIND.NAME Command 4-22
4-6. Data Areas Returned for the NCB.SESSION.STATUS Command . 4-36
4-7. Data Areas Returned for the NCB.STATUS Command 4-37
4-8. Adapter Types 4-41
4-9. The Trace Table Header Format 4-42
4-10. Trace Entry Format (Bytes 0 through 5) 4-43
4-11. Trace Entry Format (Bytes 6 through 31) 4-43
5-1. NETBIOS Frames Listed Alphabetically 5-7
5-2. NETBIOS Frames Listed Numerically 5-8
5-3. NETBIOS Name Management Frames 5-9
5-4. NETBIOS Session Establishment and Termination Frames 5-9
5-5. NETBIOS Data Transfer Frames 5-9
5-6. Additional NETBIOS Frames 5-9
5-7. NETBIOS UI Frames to Functional Address, Single-Route Broadcast 5-10
5-8. NETBIOS UI Frames to Specific Address, No Broadcast 5-10
5-9. NETBIOS UI Frames to Specific Address, General Broadcast . . . 5-10
5-10. NETBIOS I-Frames to Specific Address, No Broadcast 5-10
5-11. ADD_GROUP_NAME_QUERY Frame Format 5-11
5-12. ADD_NAME_QUERY Frame Format 5-12
5-13. NAME_IN_CONFLICT Frame Format 5-13
5-14. STATUS_QUERY Frame Format 5-14
5-15. TERMINATE_TRACE Frame Format 5-16
5-16. DATAGRAM Frame Format 5-16
5-17. DATAGRAM_BROADCAST Frame Format 5-18
5-18. NAME_QUERY Frame Format 5-19
5-19. ADD_NAME_RESPONSE Frame Format 5-20
5-20. NAME_RECOGNIZED Frame Format 5-21
5-21. STATUS_RESPONSE Frame Format 5-23
5-22. TERMINATE_TRACE Frame Format 5-24
5-23. DATA_ACK Frame Format 5-25
5-24. DATA_FIRST_MIDDLE Frame Format 5-26
5-25. DATA_ONLY_LAST Frame Format 5-29
5-26. SESSION_CONFIRM Frame Format 5-30
5-27. SESSION_END Frame Format 5-32
5-28. SESSION_INITIALIZE Frame Format 5-33

5-29. NO_RECEIVE Frame Format 5-35
5-30. RECEIVE_OUTSTANDING Frame Format 5-36
5-31. RECEIVE_CONTINUE Frame Format 5-38
5-32. SESSION_ALIVE Frame Format 5-39
6-1. DLC Status Event Trace Entry 6-14
6-2. Exception Event Trace Entry 6-14
6-3. IEEE 802.2 TRACE Bit Definition 6-17
6-4. DLC Status Event Trace Entry 6-18
6-5. NETBIOS OS2TRACEMASK Bit Definition 6-20
A-1. DLC and Direct Interface Commands A-1
A-2. NETBIOS Commands A-3
B-1. CCB Return Codes Listed by Interface B-2
B-2. DLC Status Table B-29
B-3. DLC Status Codes B-30
B-4. NCB Return Codes B-32
B-5. Token-Ring Network Adapter Status Parameter Table B-44
B-6. PC Network and Ethernet Adapter Status Parameter Table B-45
B-7. Adapter Check for CCB1 . . . . B-46
B-8. Adapter Check for CCB2 . . . . B-47
B-9. Adapter Check for CCB3 . . . . B-49
B-10. IBM Token-Ring Network Adapter Check Reason Codes for All CCBs B-51
B-11. IBM PC Network Adapter Check Reason Codes for All CCBs B-52
B-12. Network Status for CCB2 . . . . B-53
B-13. Network Status for CCB3 . . . . B-55
B-14. Token-Ring Network Status Codes for All CCBs B-55
B-15. PC Network and Ethernet Status Codes for All CCBs B-56
B-16. Bring-up Error Codes for All CCBs (Token-Ring Network Adapters) B-57
B-17. Phases B-58
B-18. Errors B-59
B-19. Recommended Actions Table B-63
B-20. PC Network and Ethernet Adapter Open Errors for All CCBs B-63
B-21. PC System Detected Errors for CCB2 . . . . B-65
B-22. PC System Detected Errors for CCB3 . . . . B-68
B-23. System Action Exceptions for CCB2 . . . . B-70
B-24. System Action Exceptions for CCB3 . . . . B-73
D-1. Interrupt Arbitrator Return Codes D-3
D-2. Interface Registration CCB Parameter Table D-4
D-3. NCB Registration Parameter Table D-7
D-4. CCB Parameter Table Structure D-8
E-1. Token Release Table E-4
E-2. NETBIOS 3.0 System Default Values E-11
E-3. NETBIOS Trace Table E-18

Notices

Trademarks

The following terms, denoted by an asterisk (*) in this publication, are trademarks of the IBM Corporation in the United States and/or other countries:

IBM	OS/2	Personal System/2
LANStreamer	PC AT	Presentation Manager
Micro Channel	PC XT	PS/2
Operating System/2		

The following terms, denoted by a double asterisk (**) in this publication, are trademarks of other companies:

CompuServe	CompuServe
Intel	Intel Corporation
Microsoft	Microsoft Corporation
NetWare	Novell, Inc.
NetWire	Novell, Inc.
Novell	Novell, Inc.
Windows	Microsoft Corporation
3Com	3Com Corporation

Preface

This manual describes the Institute of Electrical and Electronics Engineers (IEEE) 802.2 and the NETBIOS application program interfaces (APIs) provided by IBM program products for LAN adapters on PC Networks, Token-Ring Networks, and Ethernet networks. These APIs support LAN adapters in IBM Personal Computers (PCs) and IBM* Personal System/2* (PS/2*) computers.

The chapters of the fourth edition of the *IBM Local Area Network Technical Reference* that describe the application program interface are republished and updated in this reference manual. You may need to use this manual if you prepare programs that communicate on a LAN network from an IBM PC or PS/2. The LAN adapters that attach the workstation to the LAN must be supported by IBM software.

Note: If you need information about the adapter interfaces for IBM adapters, refer to the *IBM LAN Technical Reference: Adapter Interfaces*, SBOF-6221, which is described in "Related IBM Publications" on page xviii.

How This Manual Is Organized

This reference manual is divided into the following chapters and appendixes:

- Chapter 1 provides an overview of the IEEE 802.2 and NETBIOS interfaces provided by IBM, and describes the frame formats of local area networks that are supported.
- Chapter 2 describes methods of writing programs to the IEEE 802.2 and NETBIOS interfaces.
- Chapter 3 describes the Command Control Blocks that can be issued to IBM adapter support software.
- Chapter 4 describes the Network Basic Input/Output System (NETBIOS) interface.
- Chapter 5 describes the NETBIOS protocol.
- Chapter 6 describes the support of NDIS adapters using the IBM OS/2 LAN Adapter and Protocol Support.
- Appendix A contains a directory of all valid commands and the related interfaces for each, as well as page references for all commands.
- Appendix B provides return codes and exception condition tables used in programming.
- Appendix C describes the program listings on the sample diskette.
- Appendix D provides information specific to the Local Area Network Support Program.
- Appendix E provides information specific to the Communications Manager of IBM Operating System/2* (OS/2*) Extended Edition.
- Appendix F provides an overview of the NDIS interface.
- Following the Appendixes are a list of abbreviations, a glossary list that defines the terms used in this manual, and an index.

Related IBM Publications

The following IBM publications offer additional general information:

- *IBM LAN Technical Reference: Adapter Interfaces*, SBOF-6221. This Bill of Forms is a group of IBM publications that can be ordered under one publication number or as separate manuals. Each is available as a single document when ordered under its own publication number. At present, this Bill of Forms is composed of the following individual technical references:
  - *IBM LAN Technical Reference: Token-Ring Network Adapter Interface*, SC30-3588
  - *IBM LAN Technical Reference: Ethernet Adapter Interface*, SC30-3661
  - *IBM LAN Technical Reference: Token-Ring Network 16/4 Busmaster Server Adapter/A Interface*, SC30-3663
- The appropriate LAN adapter documentation (provided with the adapter)
- *IBM Token-Ring Network Architecture Reference*, SC30-3374*
- *A Building Planning Guide for Communication Wiring*, G320-8059*
- *IBM Cabling System Planning and Installation Guide*, GA27-3361*
- *Using the IBM Cabling System with Communication Products*, GA27-3620*
- *IBM Token-Ring Network Introduction and Planning Guide*, GA27-3677*
- *IBM PC Network Technical Reference*
- *Extended Services Communications Manager Configuration Guide*
- *Extended Services LAN Adapter and Protocol Support Configuration Guide*
- *OS/2 LAN Server Network Administrator Reference Volume 2: Performance Tuning*
- *IBM NTS/2 LAN Adapter and Protocol Support Configuration Guide*
- *IBM Personal Computer, Computer Language Series, Macro Assembler*
- *IBM Token-Ring Network Problem Determination Guide*, SX27-3710*
- *IBM Token-Ring Network Administrator's Guide*, GA27-3748*
- *Advanced Program-to-Program Communications for the IBM Personal Computer Programming Guide*
- *Advanced Program-to-Program Communications for the IBM Personal Computer Installation and Configuration*
- *IBM Local Area Network Support Program User's Guide*. Note that multiple versions of the LAN Support Program are currently available, depending upon your adapter environment.
- *IBM Micro Channel Technical Reference*
- *IBM Personal System/2 Technical Reference*

For assistance in obtaining IBM publications, see your place of purchase. For items marked with an asterisk (*), see your IBM representative or IBM branch office.

Guide to Information About the TCP/IP Interface

This is an abbreviated list of sources that may be helpful in understanding the TCP/IP interface. This list is not intended as an endorsement or a promotion of these particular sources. A fuller version of this TCP/IP bibliography is available in the *IBM Transmission Control Protocol/Internet Protocol Version 2.1 for DOS: Programmer's Reference.*

General Publications

- *Internetworking With TCP/IP Volume I: Principles, Protocols, and Architecture*, Douglas E. Comer, Prentice Hall, Englewood Cliffs, New Jersey, 1991
- *Internetworking With TCP/IP Volume II: Implementation and Internals*, Douglas E. Comer, Prentice Hall, Englewood Cliffs, New Jersey, 1991

Programming Publications

- *IBM Transmission Control Protocol/Internet Protocol Version 2.1 for DOS: Programmer's Reference*, SC31-7046
- *IBM Transmission Control Protocol/Internet Protocol Version 1.2.1 for OS/2: Programmer's Reference*, SC31-6077
- *IBM AIX Version 3 for RISC/6000 Communication Concepts and Procedures Volume 1*, GC23-2203
- *IBM AIX Version 3 for RISC/6000 Communication Concepts and Procedures Volume 2*, GC23-2203
- *IBM AIX Version 3 for RISC/6000 Communications Programming Concepts*, SC23-2196
- *UNIX Network Programming*, W. Richard Stevens, Prentice Hall, Englewood Cliffs, New Jersey, 1990, ISBN 0-13-949876-1

Guide to Information About the Open Data-Link Interface (ODI)

Specifications and other Novell** development documentation can be acquired through Internet, NetWire**, or NetWare** Express. However, for the complete procedures and documentation required, you should call 1-800-NETWARE.

IBM Local Area Network OEMI

The following publications make up the IBM Token-Ring Network Other Equipment Manufacture Interface (OEMI):

- *IBM Cabling System Technical Interface Information*
- *IBM LAN Technical Reference: IEEE 802.2 and NETBIOS Application Program Interfaces*, SC30-3587 (this book)
- *IBM Token-Ring Network Architecture Reference*, SC30-3374
- *Carrier Sense Multiple Access with Collision Detection*, IEEE Std 802.3-1985
- *Token-Ring Access Method and Physical Layer Specification*, IEEE Std 802.5-1985

The following publications make up the IBM PC Network OEMI:

- *IBM NETBIOS Application Development Guide*, S68X-2270
- *IBM PC Network Adapter Technical Reference*
- *IBM PC Network Adapter II Technical Reference*
- *IBM PC Network Adapter II/A Technical Reference*
- *IBM PC Network Adapter II - Frequency 2 Technical Reference*
- *IBM PC Network Adapter II/A - Frequency 2 Technical Reference*
- *IBM PC Network Adapter II - Frequency 3 Technical Reference*
- *IBM PC Network Adapter II/A - Frequency 3 Technical Reference*
- *IBM PC Network Baseband Adapter Technical Reference*
- *IBM PC Network Baseband Adapter/A Technical Reference*
- *IBM PC Network Baseband Extender Technical Reference*
- *IBM PC Network Translator Unit and Technical Reference*

For assistance in obtaining IBM publications, see your place of purchase.

How This Manual Differs from the Fourth Edition of the *IBM Local Area Network Technical Reference*

The chapters and appendixes of the *IBM Local Area Network Technical Reference*, SC30-3383-03 that describe the application program interfaces (APIs) and the parts of the *Supplement to the LAN Technical Reference*, SD21-0049-00 that deal with the APIs are being replaced by this reference manual. Information about the adapter interfaces is now published in the *IBM LAN Technical Reference: Adapter Interfaces*, SBOF-6221.

The following topics that were not found in the previous technical reference are included in this manual:

- New OS/2-based LAN Adapter and Protocol Support network communication software
- New CCB commands to support multiple group addresses in adapters that conform to the Network Driver Interface Specification (NDIS) interface and have multiple group address capability

There are also minor corrections and additions to the material last published in the fourth edition of the *IBM Local Area Network Technical Reference*, SC30-3383-03.

Chapter 1. LAN Overview and Interfaces

About This Chapter . . . 1-2
Drivers and Programs That Provide the Application Program Interfaces . . . 1-2
  NDIS and non-NDIS Adapters . . . 1-2
  IBM Programs to Support LAN Adapters . . . 1-2
Where to Find Information About IBM Adapter Interfaces . . . 1-3
Introduction to the Networks . . . 1-3
LAN Networks . . . 1-3
  The IBM Token-Ring Network . . . 1-3
  The IBM PC Network (Broadband) . . . 1-4
  The IBM PC Network (Baseband) . . . 1-4
  Ethernet Support . . . 1-4
Related Software . . . 1-4
OS/2 Communications Manager (OS/2 EE 1.3) . . . 1-5
  IEEE 802.2 Application Program Interface in OS/2 . . . 1-5
  Differences Between the Dynamic Link Routine Interface and the Device Driver Interface . . . 1-6
Why Use IEEE 802.2 or NETBIOS? . . . 1-7
IBM LAN Application Program Interfaces . . . 1-7
IEEE 802.2 Interface . . . 1-9
  The Direct Interface . . . 1-9
  The DLC Interface . . . 1-9
The APPC/PC Interface . . . 1-9
NETBIOS . . . 1-10
Token-Ring Network Frame Definition . . . 1-10
  Routing Information Field . . . 1-12
  Routing Control Field . . . 1-12
    Broadcast Indicators . . . 1-12
    Length Bits . . . 1-13
    Direction Bit . . . 1-13
    Largest Frame Bits . . . 1-14
    Reserved Bits . . . 1-15
  Route Designator Fields . . . 1-15
    Ring Number Portion . . . 1-15
    Individual Bridge Portion . . . 1-16
Frame Format on the PC Network . . . 1-16
Ethernet IEEE 802.3 Frame Format . . . 1-17
    Ethernet Frame Field Descriptions . . . 1-17
Ethernet DIX Version 2.0 Frame Format . . . 1-19

About This Chapter

This chapter introduces the software that provides the IEEE 802.2 and NETBIOS application program interfaces (APIs) and then describes interfaces to the following LANs: the IBM Token-Ring Network, the IBM PC Network, IBM Ethernet, and non-IBM Ethernet. Ethernet support includes the Institute of Electrical and Electronic Engineers Inc. (IEEE) 802.3 networks and Digital Intel Xerox (DIX) Version 2.0. The frame formats used by these networks are also discussed.

Drivers and Programs That Provide the Application Program Interfaces

The two application program interfaces provided by IBM LAN support software are IEEE 802.2 and NETBIOS. The IBM LAN support software is composed of protocol drivers that provide communication between the application program interface and the adapter. These protocol drivers provide the IEEE 802.2 and NETBIOS interfaces in one of two ways:

- By interfacing with the network adapter hardware and microcode
- By interfacing with an adapter driver that provides the Network Driver Interface Specificiation (NDIS) interface and is known as the *NDIS MAC driver* or the *network adapter driver.*[1]

NDIS and non-NDIS Adapters

The NDIS MAC driver interfaces with the protocol driver at the NDIS layer and interfaces with the adapter at the MAC layer. The protocol drivers that are supported by NDIS MAC drivers are called *NDIS protocol drivers*. NDIS configuration requires the NDIS MAC driver and an NDIS protocol driver (along with some other required files, such as a protocol manager) to provide the IEEE 802.2 and NETBIOS interfaces.

When an adapter is supported by an NDIS MAC driver, it is referred to in this manual as an *NDIS adapter*. When it is supported directly by protocol drivers, it is referred to as a *non-NDIS adapter.*

The designation as a non-NDIS adapter depends not on the adapter design, but on the type of protocol drivers used to provide the IEEE 802.2 or NETBIOS interfaces. For example, the IBM Token-Ring Network adapters with shared RAM can be either non-NDIS or NDIS. They are non-NDIS when they are supported by protocol drivers that interface with the adapter and NDIS when they are supported by protocol drivers that interface with the NDIS MAC driver.

IBM Programs to Support LAN Adapters

Examples of programs that provide protocol drivers to allow application programs to address IEEE 802.2 or NETBIOS interfaces are the IBM LAN Support Program (LSP) in DOS and several program products in OS/2, including IBM Extended Services for OS/2 Version 1.0 (ES), IBM OS/2 LAN Server Versions 2.0 and 3.0, Network Transport Services/2 (NTS/2), and the Communications Manager delivered with IBM OS/2 Extended Edition (EE) Version 1.3.

[1] Appendix F, "NDIS Overview" provides an overview of NDIS.

These programs, which are loaded in the host computer in which the LAN adapter is installed, are included in the following list:

- The Local Area Network Support Program. Several versions of this program are available, depending upon your adapter environment.
- The Communications Manager provided with OS/2 EE 1.3.

  Note: The LAN protocol support delivered with EE 1.3 cannot operate under OS/2 2.0. Therefore, the new LAN Adapter Protocol Support (LAPS) is required for OS/2 2.0 or higher. The new LAPS will also run in the OS/2 1.3 environment.
- LAN Adapter and Protocol Support (LAPS) provided with ES 1.0, NTS/2, or LAN Server 2.0 or 3.0.

Note: NDIS MAC drivers are shipped with the NDIS adapters, and may not be included in the support program.

Where to Find Information About IBM Adapter Interfaces

The *IBM Local Area Network Technical Reference* has been divided into this manual and a group of manuals that describe the adapter interfaces. See "Related IBM Publications" on page xviii for the titles of the manuals that describe the adapter interfaces.

Introduction to the Networks

The following local area networks use the interfaces described in this book:

- IBM Token-Ring Network
- IBM PC Network
- IBM PC Network (Baseband)
- Ethernet

IBM* Personal Computers (PCs) and Personal System/2* (PS/2*) computers, often referred to as *workstations*, can be connected on these networks. The appropriate LAN adapters must be installed in the workstations; these adapters are supported by the IBM program products listed in "IBM Programs to Support LAN Adapters" on page 1-2.

LAN Networks

Following is a brief description of the applicable local area networks.

The IBM Token-Ring Network

The Token-Ring Network, a token-ring, star-wired local area network, can accommodate on one ring up to 260 attaching devices (printers, processors, controllers). Bridges can connect multiple rings together to form a network of more than 260 devices. These attaching devices connect to one another by a series of cables, access units, and special adapters installed in the attaching devices.

Application programs running in each workstation (such as a PC or PS/2 computer) can direct the adapter to become a part of the ring. This book describes the commands used by programs to control the Token-Ring Network adapter's activity on the network. Refer to *IBM Token-Ring Network Introduction and Planning Guide* for more information about the network.

The IBM PC Network (Broadband)

The PC Network (Broadband) is a bus-attached, broadband local area network that can accommodate up to 72 attaching devices with IBM components.

The IBM PC Network (Baseband)

The PC Network (Baseband) is a bus-attached, baseband, local area network that can accommodate up to 80 attaching devices with IBM components.

Ethernet Support

Ethernet networks are bus-attached local area networks. IBM supports Ethernet with the Operating System/2* Extended Edition Version 1 Release 3 (OS/2* EE 1.3), LAN Adapter and Protocol Support in OS/2, and the LAN Support Program Version 1.2 or higher support in DOS for IEEE 802.3 and DIX Version 2.0. Refer to the *IBM Operating System/2 Extended Edition Version 1.3 System Administrator's Guide for Communications*, the documents for Extended Services (ES) and Network Transport Services/2 (NTS/2) listed in "Related IBM Publications" on page xviii, and the *LAN Support Program User's Guide* for additional information about the Ethernet adapters supported.

Related Software

The software listed below provides the interface to allow communication on the networks using local area network adapters.

- For use with IBM Disk Operating System (DOS)
  - The protocol drivers provided with the Token-Ring Network PC Adapter and Token-Ring Network PC Adapter II
  - The Local Area Network Support Program
  - Advanced Program-to-Program Communications for the IBM PC (APPC/PC)
- For use with OS/2
  - The Communications Manager provided with OS/2 EE 1.3
  - The LAPS provided with ES 1.0, NTS/2, and LAN Server.

  Note: The protocol support in Communications Manager provided with OS/2 EE 1.3 for Ethernet is NDIS-based. Beginning with LAPS for OS/2 2.0, all adapter support is NDIS-based. See Chapter 6, "Support of NDIS Adapters Using IBM OS/2 LAN Adapter and Protocol Support" for a discussion of OS/2 NDIS support.

Application programs use these interfaces to communicate on a local area network.

OS/2 Communications Manager (OS/2 EE 1.3)

Communications Manager is a component of OS/2 EE 1.3. It provides comprehensive communication capabilities for a variety of interconnections. Functions that were previously available only in various communications programs for DOS are now combined with the functions of multitasking and expanded memory support. Communications Manager enables users to connect to a range of computers, including IBM and non-IBM host systems and other personal computers. In addition, multiple connections can be active concurrently, giving users access to information wherever it is located.

Communications Manager supports a wide range of communication capabilities that include:

- 3270 terminal emulation
- 5250 terminal emulation
- ASCII terminal emulation
- IBM Server-Requester Programming Interface (SRPI)
- IBM Systems Network Architecture (SNA) Advanced Program-to-Program Communication (APPC)
- IBM Asynchronous Communications Device Interface (ACDI)
- IEEE 802.2 Application Program Interface (API)
- IBM NETBIOS API
- Emulator High-Level Language Application Program Interface (EHLLAPI).

Since OS/2 provides multitasking capability, the various communications options can usually run concurrently. In many cases, this eliminates the need to load and unload programs to communicate with different systems.

IEEE 802.2 Application Program Interface in OS/2

The IEEE 802.2 API provided by the Communications Manager and LAPS supports both the direct and data link control (DLC) interfaces described in this book. Application programs can use the direct interface only or can use both the direct and DLC interfaces.

Two methods are available to access the IEEE 802.2 API: the dynamic link routine (DLR) interface, and the device driver (DD) interface. Using the device driver interface, the protocol drivers interface directly with the LAN device drivers provided by Communications Manager or LAPS.

Application programs communicate across the IEEE 802.2 API using command control blocks (CCBs). For this communication, the DLR interface uses CCB2, and the DD interface uses CCB3. Chapter 2, "Programming Conventions for the IEEE 802.2 Interface," provides more information about CCBs.

A DLR accesses the NETBIOS API provided by Communications Manager. Application programs communicate across the NETBIOS API with network control blocks (NCBs). For more information on NETBIOS and the NCB commands, see Chapter 4, "NETBIOS."

To use the IEEE 802.2 API or the NETBIOS API with OS/2, Communications Manager must be installed and configured. Refer to *IBM Operating System/2 Extended Edition Getting Started* for information on installing Communications Manager and the LAN device drivers. Refer to *IBM Operating System/2 Extended Edition System Administrator's Guide for Communications* for information on configuring Communications Manager.

Differences Between the Dynamic Link Routine Interface and the Device Driver Interface

Two levels of OS/2 interfaces exist: the Dynamic Link Routine Interface and the Device Driver Interface. An application program can use either interface, but cannot use both interfaces at the same time and still be considered a single application program. Resources provided to and resources obtained from one of the OS/2 interfaces cannot be used at the other OS/2 interface.

An application program can easily use one of the DLR interfaces by making the appropriate external references to an OS/2 DLR interface. In order for an application program to use one of the DD interfaces, the application program itself must be a device driver or have a device driver as one of its components. The application program device driver must be set up to support communication between device drivers so that the application program device driver can be called by the protocol drivers for posting of events.

Several factors can be involved in determining the best OS/2 interface for your programming needs. Consider these factors when choosing your OS/2 interface:

- The programming language used to develop your application programs
  - Device driver components of application programs must be written in Assembler because registers must be accessed and processed. In addition, flags must be tested for error conditions.
- Performance
  - The DLR interfaces use semaphores and create threads for application programs in order to post events; therefore, task switches are involved when using the interface. When events occur that affect the application program (for example, command completions and network status changes), the application program can respond to the event after one of its threads is dispatched by a semaphore being cleared.
  - The DD interface calls the application program's device driver to post events. When an event occurs that affects the application program, the application program is notified without delay and can respond immediately to the event without a task switch.
- Complexity
  - The DLR interface manages asynchronous events and allows the application program to process event information at its convenience.
  - While the DD interface does provide better performance to the application program, it also requires the application program to share in some of the responsibilities associated with processing asynchronous events. When events occur, the protocol drivers call the application program to post the event. No event information is queued for later notification or retrieval by the application program. In addition, the application program device driver is responsible for ensuring that data structures and buffers passed to the

protocol drivers are located in valid memory segments and are locked to prevent moving or swapping by OS/2.

Why Use IEEE 802.2 or NETBIOS?

The IEEE 802.2 and NETBIOS APIs consist of the following interfaces:

- IEEE 802.2 API
  - Direct interface
  - DLC interface
- NETBIOS API
  - NETBIOS interface.

When choosing which interface to use, you should take into account these criteria:

- Usability of the interface

  The NETBIOS interface provides a simple interface for the application program and does not require the application program to understand DLC.
- The performance required by your application program

  The IEEE 802.2 interfaces provide better performance but require the application programs to be significantly more complex. Performance advantages can be up to two times that of NETBIOS based on the amount of data transferred between the application programs.
- The interfaces used by other application programs with which your application program may interact.

IBM LAN Application Program Interfaces

The Local Area Network Support Program with DOS, Communications Manager with OS/2 EE 1.3, and LAPS provide both IEEE 802.2 and NETBIOS interfaces. Within the IEEE 802.2 interface, the direct interface and the DLC interface are supported.

The application program can issue CCBs to the protocol drivers to interface with the adapter. By using CCBs, the application program is freed from the burden of interacting directly with the adapter. For information on CCBs and communicating with the protocol drivers, see Chapter 2, "Programming Conventions for the IEEE 802.2 Interface," and Chapter 3, "The Command Control Blocks." For information about interacting directly with the adapters, see the adapter interface books listed in "Related IBM Publications" on page xviii. Figure 1-1 on page 1-8 shows the relationship of application programs, the protocol drivers, and the network adapter when using DOS. Figure 1-2 on page 1-8 shows the relationship of application programs, the protocol drivers, and different adapters when using OS/2 EE 1.3.

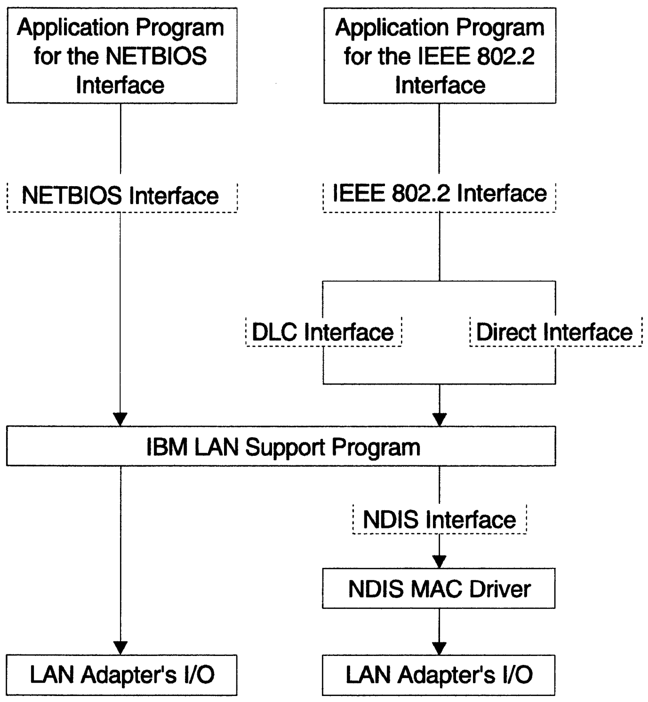


Figure 1-1. DOS IEEE 802.2 and NETBIOS Interfaces

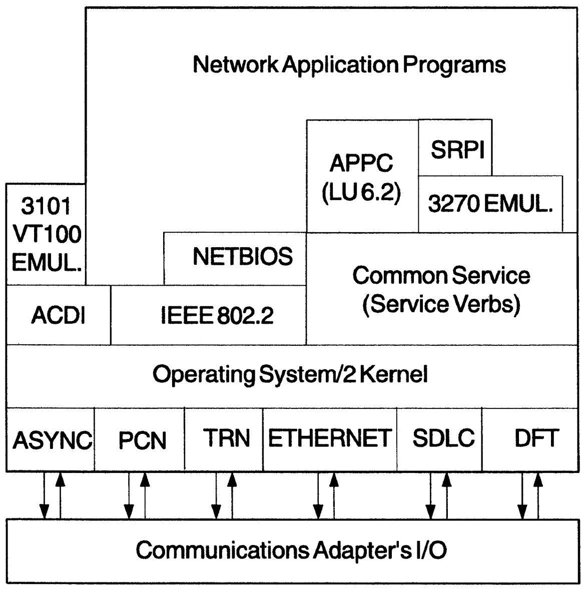

Network Application Programs
APPC (LU6.2)
SRPI
3270 EMUL.
3101 VT100 EMUL.
NETBIOS
Common Service (Service Verbs)
ACDI
IEEE 802.2
Operating System/2 Kernel
ASYNC
PCN
TRN
ETHERNET
SDLC
DFT
Communications Adapter's I/O

Figure 1-2. OS/2 EE Communications Manager Interfaces

A network application program assembles a control block containing a command and related information for the adapter. Control passes to the protocol drivers, and the application program awaits the results.

Appendix A, "Valid Commands," contains a directory of all commands, their related interfaces, and a page reference to a description of each. The functions of the DLC interfaces of the adapter and the protocol drivers are compatible with the service specifications of the IEEE 802.2 Logical Link Control (LLC). Detailed information on these interfaces is contained in the *IBM LAN Technical Reference: Adapter Interfaces*, SBOF-6221.

Each of the following interfaces provides a means of communicating with the adapter. Depending on which you choose, the code you provide shares the responsibility for control of the adapter with the protocol drivers found in the IBM support program you are using.

IEEE 802.2 Interface

This interface is implemented using CCBs and consists of two types of interface: direct interface and DLC interface.

The Direct Interface

The direct interface allows control of the adapter using control blocks.

This interface provides the ability to open and close an adapter, obtain error status, and set addresses. It also permits transmission and reception of Medium Access Control (MAC) (Token-Ring adapter cards only) and non-MAC frames directly without LLC protocol assistance.

Chapter 2, "Programming Conventions for the IEEE 802.2 Interface," describes the use of this interface in detail.

The DLC Interface

The DLC interface, together with the direct interface, provides an interface to application programs using the LLC sublayer of data link control protocol. The DLC protocol consists of the LLC sublayer and the medium access control (MAC) layer protocol. The interface can be used in two ways.

- For IEEE Type 1 communication, which is connectionless communication between devices providing no guarantee of delivery (through the DLC service access point (SAP) interface).
- For IEEE Type 2 communication, which is connection-oriented services (through the DLC station interface), providing point-to-point connectivity with guaranteed delivery and retry.

The adapter and the protocol drivers provide much of the communication overhead function, which permits less complex application programming.

Chapter 2, "Programming Conventions for the IEEE 802.2 Interface," describes the use of this interface in detail. The *IBM Token-Ring Network Architecture Reference* explains communication using DLC in more detail.

The APPC/PC Interface

The APPC/PC program is a product that uses the protocol drivers provided with the Local Area Network Support Program. Refer to *Advanced Program-to-Program Communications for IBM Personal Computer Installation and Configuration Guide.* Also, *Advanced Program-to-Program Communications for the IBM Personal Computer Programming Guide* explains how to design and write APPC/PC transaction programs.

Note: The Communications Manager provides the APPC/PC support for OS/2.

NETBIOS

The IBM NETBIOS API provides a program interface to the LAN so that an application program can have LAN communication without programming to the 802.2 API. NETBIOS provides the necessary DLC communications for the application program. NETBIOS names identify nodes on the LAN, and NETBIOS supports two types of data transfer. Session support provides guaranteed delivery of the data, while datagram support does not guarantee delivery.

NETBIOS application programs require that protocol drivers that provide NETBIOS support be used. These drivers are part of the Local Area Network Support Program or the OS/2 programs.[2] See Chapter 4, "NETBIOS" for information about NETBIOS and its use.

Token-Ring Network Frame Definition

A Token-Ring Network frame contains the following elements:

- Start delimiter (SD)—1 byte
- Access control (AC)—1 byte with the frame bit on
- Frame control (FC)—1 byte
- Destination address—6 bytes
- Source address—6 bytes
- Optional routing field—up to 18 bytes
- Optional DLC header field—3 to 4 bytes
- Optional information (data) bytes
- Frame check sequence (FCS)—4 bytes
- End delimiter (ED)—1 byte
- Frame status (FS)—1 byte.

Figure 1-3 on page 1-11 shows the Token-Ring Network frame format.

[2] When the original PC Network Adapter or the PC Network Protocol Driver is used, the protocol drivers provided by the LAN Support Program or the OS/2 support programs are not required.

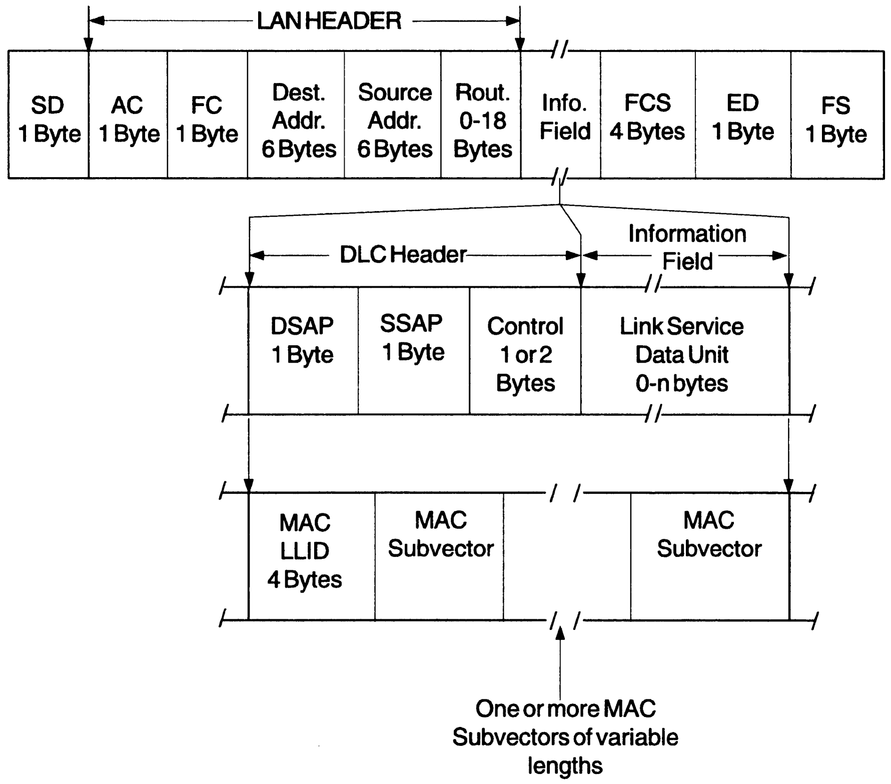


Figure 1-3. Token-Ring Network Frame Format. Bits are transmitted in bytes, most significant bit (bit 7) first.

The physical, or LAN, header consists of the AC byte, the 1-byte FC field, 6 bytes of destination address, 6 bytes of source address, and from zero to 18 bytes of routing information. This is followed by the information field. Finally, the physical trailer (PT) is included, consisting of 4 bytes of the FCS field, the ED byte, and the FS byte.

The frame may be one of two types:

- MAC frame
- Non-MAC frame.

MAC frames contain information about the status of an adapter or the ring.

Certain MAC frames can be received by the adapter and provided to the application program at the direct interface. Some MAC frames can be sent to the adapter for transmission on the ring using the direct interface of IEEE 802.2.

Some non-MAC frames contain data and messages that users transmit to one another.

Some non-MAC frames contain LLC protocol-only information transmitted by the adapter. These frames are used for Type 2 protocol support.

The 2 most significant bits of the FC byte define the frame type. The types are:

B'00'	MAC frame
B'01'	LLC frame (non-MAC)
B'10'	Reserved
B'11'	Reserved.

Routing Information Field

Bridges use the routing field to forward frames to their destination. This field is required if the frame is forwarded by bridges to other rings.

Bit sequences described in the routing field may differ from IBM PC format in that the most significant bit of a byte is designated **7** for the IBM workstation and may be called **0** elsewhere. Only the representation differs; the byte's content is not altered.

This field, when present, consists of a 2-byte routing control field and up to eight 2-byte route designators, as shown below:

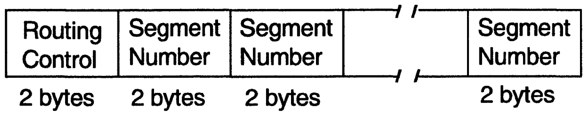
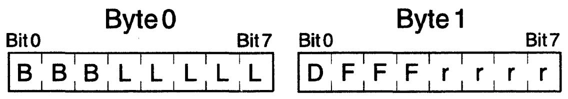


Figure 1-4. Routing Information Field

Routing Control Field

The format for the routing control field is shown below:



B = Broadcast Indicators
L = Length Bits
D = Direction Bit
F = Largest Frame Bits
r = Reserved Bits

Figure 1-5. Routing Control Field

Broadcast Indicators

The broadcast indicators indicate whether the frame is to be sent along a specified path, to all the segments in a network (potentially resulting in multiple copies on a given segment), or to all the segments so that only one copy of the frame appears on each segment in the network.

- B'0XX' = Non-broadcast: This indicates that the route designator field contains a specific route for the frame to travel through the network.
- B'10X' = All-routes broadcast: This indicates that the frame will be transmitted along every route in the network to the destination station. Frames transmitted as all-routes broadcast will result in as many copies at the destination station as there are different routes to the destination station.

Note: An all-routes broadcast is independent of an all-stations broadcast, which is indicated by all ones in the DA field. An all-stations broadcast implies that every station on the segment will copy the frame, while an all-routes broadcast implies that every bridge in a network will copy and forward the frame to its adjoining segment (unless the next route designator already appears in the routing information field).

- B'11X' = Single-route broadcast: This indicates that only certain designated bridges will relay the frame from one segment to another with the result that the frame will appear exactly once on every segment in the network.

Note: X'' means the bit can be either a 0 or a 1. Its value does not affect the meaning of the indicator.

Length Bits

The 5 length bits indicate the length in bytes of the routing information field. Ring stations use the length to parse the rest of the frame correctly. (A ring station *parses* a frame by separating it into its individual fields. When a station parses a frame, it also checks for errors in the formatting of the frame.)

For all-routes or single-route broadcast frames, the originating ring station initializes the length field to X'2', to represent the 2 routing control bytes. Bridges alter the routing information field in broadcast frames by adding route designators.

For non-broadcast frames, which are already carrying routing information, the length field indicates the length of the routing information field, and remains unchanged as the frame traverses the network.

Each bridge checks the length bits. If the length is an odd number of bytes, or if it is less than 2 bytes or greater than 18 bytes, the bridge does not forward the frame.

For all-routes broadcast frames, the length field indicates to a bridge where to append the route designator. The first bridge to forward the frame adds X'4' to the length value (2 bytes for the first route designator and 2 bytes for the next ring's route designator). After that, every bridge that forwards the frame adds X'2' to the length field (2 bytes for the next ring's route designator).

At any given time after crossing the first bridge, the formula {[(Length - 2)/2] - 1} indicates the number of bridges crossed.

Direction Bit

The direction bit enables the bridge to correctly interpret the route designators when it forwards the frame.

If the direction bit is set to B'0', the bridge interprets the routing information field from left to right; if it is set to B'1', it interprets the field from right to left. Using this bit allows the list of ring numbers and bridge numbers in the routing information field to appear in the same order for frames traveling in either direction along the route.

For all-routes broadcast frames, the originating ring station sets the direction bit to B'0'. Bridges do not need the direction bit in broadcast frames, but receivers could uniformly complement the received bit when they obtain routing information from frames with routing information fields.

For off-ring non-broadcast frames, the originating ring station sets the direction bit to B'0' in all frames transmitted to the target, while the target sets the direction bit to B'1' in all non-broadcast frames to the originating ring station.

Largest Frame Bits

These bits specify the largest-size information field for the frame, excluding headers, that can be transmitted between two communicating stations on a specific route.

A station that originates a broadcast frame sets the largest frame bits to B'111', the largest possible frame that can travel any path. Bridges that relay a broadcast frame examine the largest frame bits. If the designated size of the largest frame is greater than the capability of that part of the route, the bridge reduces the largest frame encoding to indicate the maximum information field.

The largest field value returned in the responses to the broadcast indicates the largest possible frame each specified route can handle.

The largest frame code points have the following values:

- 000—As many as 516 bytes in the information field. 516 represents the smallest maximum frame size that a medium access control must support under ISO 8802/2 LLC and ISO connectionless-mode network service (ISO 8473).
- 001—As many as 1470 bytes in the information field. 1470 represents the largest frame size that ISO 8802/3-standard local area networks can support.
- 010—As many as 2052 bytes in the information field. 2052 represents a frame size that is useful for transferring a (typical) screen-full of data; that is, this frame size will support the transfer of data for an 80 X 24 screen plus control characters.
- 011—As many as 4399 bytes in the information field. 4399 represents the largest frame size that can be transmitted using the Fiber Distributed Data Interface (FDDI) Draft Proposed American National Standard. It is also the largest frame size possible for ISO 8802/5-standard stations.
- 100—As many as 8130 bytes in the information field. 8130 represents the largest frame size that ISO 8802/4-standard local area networks can support.
- 101—As many as 11 407 bytes in the information field.
- 110—As many as 17 749 bytes in the information field. 17 749 represents the maximum frame size that a medium access control supports for ISO 8802/5-standard stations.
- 111—Used in all-routes broadcast frames and for values greater than 17 749.

Note: Source-routing end stations on media with a maximum frame size should not send frames in which the headers, routing information fields, and information fields exceed that maximum frame size.

Bit definitions for the largest frame bits can be found in the international standard: ISO/IEC 10038:1993 ANSI/IEEE Std 802.1d 1993 edition.

Reserved Bits

These bits are reserved by IBM for future use. They are transmitted as B'0's; their value is ignored by receiving ring stations.

Route Designator Fields

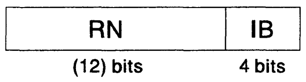
Each ring in a given multiple-ring network is assigned a unique ring number; each bridge is assigned a bridge number, which may or may not be unique. Together, the ring and bridge number form a route designator. When an all-routes broadcast frame is transmitted, each bridge that forwards the frame to another ring adds its bridge number and that ring's number to the frame's routing information field.

When a bridge receives a frame to forward to a ring, the bridge compares the route designators already present in the routing information field with its attached ring numbers and bridge number.

- If there is a target ring number match in an all-route or single-route *broadcast* frame, the bridge discards the frame because it has already circled the target ring.
- If there is *not* a target ring number match in an all-route or single-route *broadcast* frame, the bridge adds its route designator to the frame's routing information field and forwards it.
- If there is a ring number, bridge number, and ring number combination match in a *non-broadcast* frame, the bridge forwards the frame to the indicated ring.
- If there is *not* a ring number, bridge number, and ring number combination match in a *non-broadcast* frame, the bridge discards the frame.

When the frame reaches its destination, the sequence of route designators describes the path from the source ring to the destination ring.

The 2 bytes of the route designator are divided into the ring number portion (12 bits) and the individual bridge number portion (4 bits), as shown below. The individual bridge portion allows parallel bridges to exist, and to share traffic between the same two rings.



RN = Ring Number Portion
IB = Individual Bridge Portion

Figure 1-6. Route Designator Field

Ring Number Portion

Bridges that are attached to different rings have different values for the ring number portion of the route designator; bridges that are attached to the same ring have the same value.

Individual Bridge Portion

Bridges that are attached to the same ring can have the same value for the individual bridge portion of the route designator. However, parallel bridges (those that are attached to the same *two* rings) must have different values.

Because the end of a route is a ring and not a bridge, the individual bridge portion of the last route designator in the routing information field is not defined (that is, it is all B'0's).

Frame Format on the PC Network

A frame on the PC Network consists of the following fields:

- Start delimiter (SD)—1 byte
- Destination address—6 bytes
- Source address—6 bytes
- Zeroes—2 bytes
- Optional routing field—up to 18 bytes
- Optional DLC header field—3 to 4 bytes
- Optional information (data) bytes
- The frame check sequence (FCS)—4 bytes
- Frame status (FS)—1 byte
- Pad characters (flags) if needed to reach the minimum frame length.

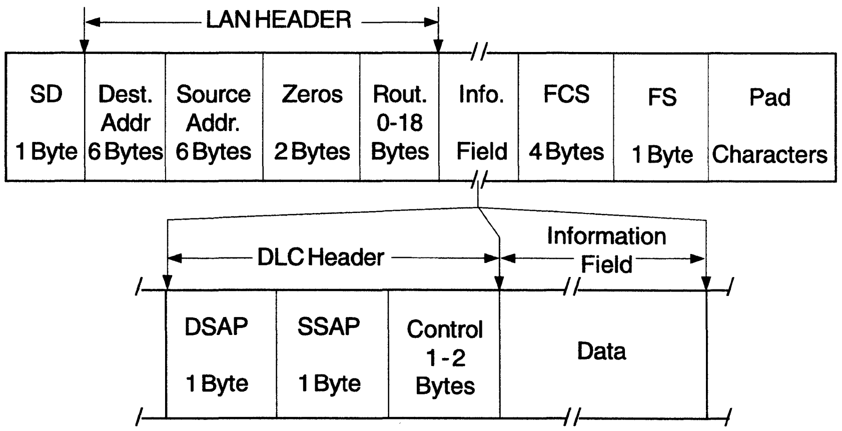


Figure 1-7. PC Network Frame Format. Bits are transmitted in bytes, most significant bit (bit 7) first.

The physical, or LAN, header consists of the destination and source addresses, 2 zero bytes, and from zero to 18 bytes of routing information. This is followed by the user-provided data. The LAN header on the PC Network is different from that on the Token-Ring Network, but to provide compatibility for application programs, the difference is not reflected at the interface to the application program. In cases where the application program provides the access control and frame control bytes (used for the Token-Ring Network), the protocol drivers simply omit the bytes from the transmitted frame and insert 2 bytes of zeroes following the source address. Note, however, that the protocol drivers check to make sure that the frame control byte specifies an LLC, not a MAC, frame. They transmit the address fields with the bit sequence expected on the PC Network, not the bit sequence expected on the Token-Ring Network.

The formats of the routing information, the DLC header, and the information fields are identical on both networks; these formats are described in detail in the *IBM Token-Ring Network Architecture Reference*. Bit sequences in that book, and possibly other documentation, may differ from IBM PC format in that the most significant bit of a byte is designated **7** for the IBM workstation and may be called **0** elsewhere. Only the representation differs; the byte's content is not altered.

Ethernet IEEE 802.3 Frame Format

IBM support for IEEE 802.3 Ethernet uses a frame that consists of the following fields:

- Preamble—7 bytes
- Start delimiter (SD)—1 byte
- Destination address—6 bytes
- Source address—6 bytes
- LPDU length—2 bytes
- LPDU
  - Destination SAP address (DSAP)—1 byte
  - Source SAP address (SSAP)—1 byte
  - Control field—1 or 2 bytes
  - Information field
- Pad—0 to 43 bytes
- Frame check sequence (FCS) or cyclic redundancy check (CRC)—4 bytes.

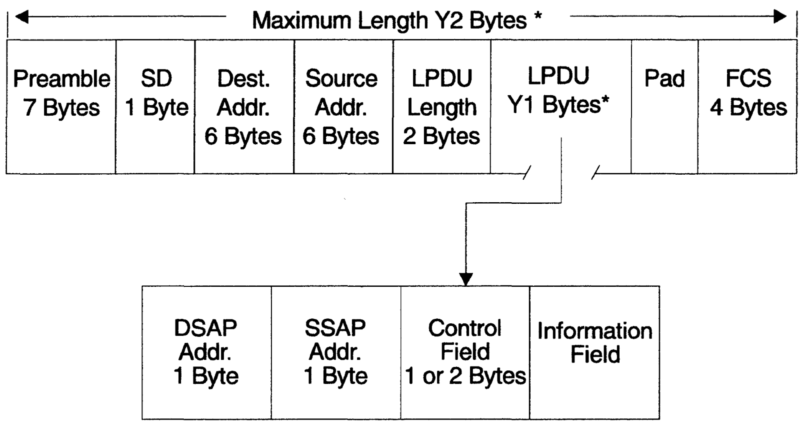


*Figure 1-8. IEEE 802.3 Frame Format. * See Table 2-14 on page 2-50 for the values of Y1 and Y2. See "Ethernet Frame Field Descriptions" for more information about these fields.*

Ethernet Frame Field Descriptions

The following list gives field descriptions for the Ethernet frames:

Preamble

This field is a synchronization pattern that contains alternating binary ones and zeros. For DIX Version 2.0, it is a 64-bit field that ends with a frame start delimiter of two consecutive ones. For IEEE 802.3, it is a 56-bit field and does not end with the frame start delimiter of two consecutive ones.

Start Frame Delimiter (IEEE 802.3)

This 1-byte field contains binary ones and zeros, and ends with two consecutive ones. The contents of this field match the eighth byte of the preamble field in DIX Version 2.0 frames.

Destination Address

This 6-byte field specifies the station to which the packet is being transmitted.

- The first bit transmitted indicates whether the destination address is an individual address (B'0') or a group address (B'1').
- The second bit transmitted indicates whether the address was universally administered (B'0') or locally administered (B'1').

Source Address

This 6-byte field contains the unique address of the station transmitting the packet.

- The first bit transmitted is always zero.
- The second bit transmitted indicates whether the address was universally administered (B'0') or locally administered (B'1').

Type Field (DIX Version 2.0)

This 2-byte field contains the registered value that identifies the high-level protocol that will interpret the LPDU. The value registered for this frame description is X'80D5' for SNA communications. This same Type Field value (X'80D5') is also used for any other application using the IEEE 802.2 API (direct or DLC), including NETBIOS. Note that the SAP values are different for SNA path control and NETBIOS. Refer to the *IBM Token-Ring Network Architecture Reference* for more information.

Data Field (DIX Version 2.0)

This data field contains an integral number of bytes ranging from 46 to 1500 bytes, inclusive. The minimum packet size ensures that valid packets are distinguishable from collision fragments.

LPDU Length

This 2-byte field specifies the byte length of the LPDU, from the destination SAP address to the last byte of the information field, inclusive.

LPDU

This LLC protocol data unit consists of SAP addresses, a 1- or 2-byte DLC field, and an optional information field. The field is 1497 bytes in a DIX Version 2.0 frame, and 1500 bytes in an IEEE 802.3 frame.

Pad (IEEE 802.3)

This field is used when needed to reach the minimum frame size requirement of 64 bytes. The pad can be up to 40 bytes. The protocol drivers will add the appropriate pad bytes, if required. The minimum frame size ensures that valid frames are distinguishable from collision fragments.

Trailing Pad (DIX Version 2.0)

This field is used when needed to reach the minimum frame size requirement of 64 bytes. The trailing pad can contain up to 43 bytes. The protocol drivers will add the appropriate pad bytes, if required. The minimum frame size ensures that valid frames are distinguishable from collision fragments.

FCS

This 4-byte frame check sequence is based on all fields, starting with the destination address.

Ethernet DIX Version 2.0 Frame Format

IBM support for DIX Version 2.0 Ethernet uses a frame that consists of the following fields:

- Preamble—8 bytes
- Destination address—6 bytes
- Source address—6 bytes
- Type field—2 bytes
- Data
  - LPDU length—2 bytes
  - Leading pad—1 byte
  - LPDU
    - Destination SAP address—1 byte
    - Source SAP address—1 byte
    - Control field—1 or 2 bytes
    - Information field—n to 1494 bytes
  - Trailing pad—0 to 40 bytes
- FCS or cyclic redundancy check (CRC)—4 bytes.

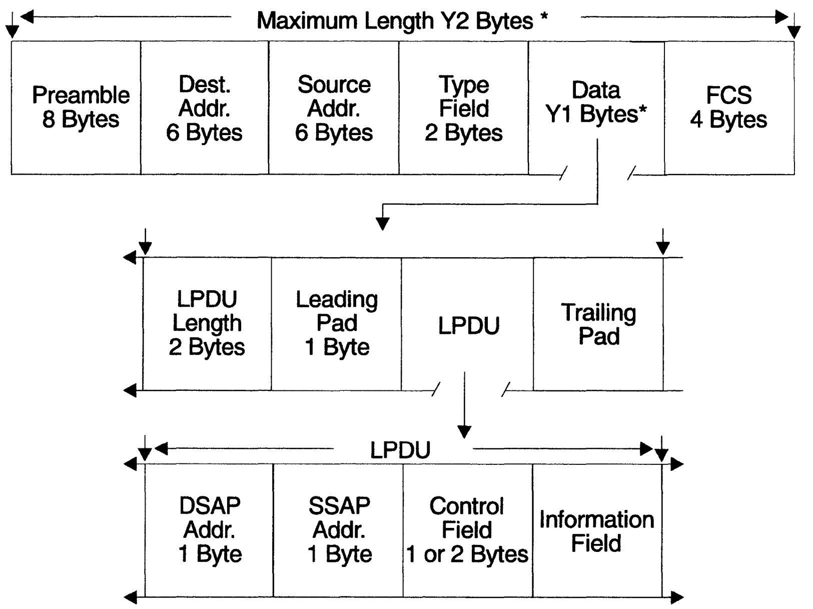


*Figure 1-9. DIX Version 2.0 Frame Format. * See "Ethernet Frame Field Descriptions" on page 1-17 for more information about these fields.*

Chapter 2. Programming Conventions for the IEEE 802.2 Interface

About This Chapter 2-2
IBM Program Products for LAN Adapter Support 2-2
IEEE 802.2 Programming Conventions with DOS 2-3
  Local Area Network Support Program (CCB1) Calling Conventions 2-3
  Local Area Network Support Program (CCB1) Command Completion
    Appendages 2-4
  Local Area Network Support Program (CCB1) Control Blocks 2-6
Programming Conventions with OS/2 2-7
  The OS/2 DLR Interface 2-7
    Execution of Multiple Commands with the DLR Interface 2-7
    User's Data Segment Restrictions with the DLR Interface 2-7
    User's Data Segment Guidelines with the DLR Interface 2-8
    DLR Interface (CCB2) Calling Conventions 2-8
    DLR Interface (CCB2) Command Completion 2-9
    DLR Interface (CCB2) Control Blocks 2-12
  The OS/2 DD Interface 2-12
    User's Data Segment Restrictions with the DD Interface 2-12
    User's Data Segment Guidelines with the DD Interface 2-13
    DD Interface (CCB3) Calling Conventions 2-13
    DD Interface (CCB3) Command Completion 2-14
    DD Interface (CCB3) Control Blocks 2-18
Control Blocks for All CCBs 2-19
  CCB Field Explanations 2-19
Addressing 2-26
Adapter Addresses 2-26
  Stations, SAPs, and IDs 2-27
    SAPs 2-28
  SAP Assignments 2-29
DLC 2-30
  Types of Service 2-31
  Command Sequences 2-31
  Link Station States 2-33
  Timers 2-35
  Guidelines for Selecting Parameter Values 2-36
Transmitting, Receiving, and Buffers 2-39
  Buffer Pools 2-40
  Receive Buffers 2-41
    Receive Buffer Formats 2-42
    Buffer Fields Explanations 2-45
  Transmit Buffers 2-46
    Transmit Buffer Formats 2-47

About This Chapter

This chapter includes a discussion of the programming conventions for application programs that run on LANs and use the IEEE 802.2 application program interface. Also included are a description of the command control blocks (CCBs) and the fields of those CCBs, a description of the command sequence for DLC communication, and a discussion of transmitting and receiving on the LANs.

IBM Program Products for LAN Adapter Support

At this time, OS/2 EE 1.3 with Communications Manager, LAN Adapter and Protocol Support in OS/2, NTS/2, OS/2 LAN Server, and the Local Area Network Support Program in DOS are the IBM program products provided to support adapters on LANs. This chapter discusses the IEEE 802.2 LAN interface provided by these products. The 802.2 interface is implemented by commands which are communicated to the protocol drivers of the support program products using CCBs. Most of the commands used by the DOS and OS/2 program products are the same and the structure of the commands is very similar. Where it is possible to discuss one command that can be used for all products, the term Command Control Block (CCB) is used. Where it is not possible to talk about them as one, each is discussed separately as CCB1, CCB2, and CCB3. This chapter explains the major differences in the CCBs. The different command control block structures are referred to as follows:

CCB1	The command control block for the IEEE 802.2 interface provided by the Local Area Network Support Program in DOS.
CCB2	The command control block for the DLR interface provided with the Communications Manager of OS/2 EE 1.3 and with LAPS.
CCB3	The command control block for the DD interface provided with the Communications Manager of OS/2 EE 1.3 and with LAPS.
CCB	When *CCB* is used without a qualifier, the information refers to all three interfaces.

The Local Area Network Support Program, OS/2 EE 1.3, and other OS/2 support software provide software support for the adapter when using any of the supported Token-Ring Network, PC Network, or Ethernet adapters. The Local Area Network Support Program and OS/2 EE 1.3 support up to two adapters in one workstation. The first adapter, numbered 0, is the *primary adapter*; the second adapter, if used, is numbered 1 and is the *alternate adapter*.

Starting with LAPS in OS/2, the OS/2 support software provides support for up to four adapters in one workstation. These adapters can be identified using any numbers in the range 0–15.

Programs to support the adapter must be loaded into workstation memory. The IEEE 802.2 interface provided by the protocol drivers offers three levels of entry to the LAN:

- The direct interface
- The DLC SAP interface
- The DLC station interface.

If NETBIOS is also loaded into memory, an interface to NETBIOS commands is provided. See Chapter 4, "NETBIOS," for more information about NETBIOS.

The support programs that provide the IEEE 802.2 interface allow an application program to use the adapter by providing control blocks. When the application program uses a CCB to issue a command, the support program calls various protocol drivers that convey the information found in the CCB to the adapter. This process frees the application program from the burden of interacting with the adapter.

IEEE 802.2 Programming Conventions with DOS

The following sections describe the calling conventions, command completion, and control block structures for the Local Area Network Support Program.

Local Area Network Support Program (CCB1) Calling Conventions

To issue a request to the Local Area Network Support Program, a network application program assembles a control block containing a command and related information for the adapter. The application program then puts the workstation's main memory address of this control block into the extra segment (ES) and base (BX) registers.[1] At this point, the application program issues an X'5C' software interrupt. The appropriate protocol driver responds to the X'5C' interrupt by processing the control block. While processing the CCB, the protocol driver enables interrupts.

Local Area Network Support Program (CCB1) Command Completion

The CCB_RETCODE is initially set to a value of X'FF'. Once any immediate command processing is accomplished, control is returned to the application program. At that point the application program can continue with other processing, but cannot disturb the CCB or associated data. (The CCB_RETCODE is still X'FF'.) When the command is completed, the protocol driver sets the return code in both the AL register of the computer and the CCB_RETCODE field, and checks the CCB_CMD_CMPL field. The CCB_CMD_CMPL field provides the protocol driver with the address of a command completion appendage of an application program.

- If the CCB_CMD_CMPL field is not zero, the protocol driver transfers control to the application program at the address provided. The application program continues with the command completion appendage and returns control to the protocol driver when completed.

  Upon entry, the command completion appendage can obtain the final return code from either the AL register or the CCB_RETCODE field.

- If the CCB_CMD_CMPL field contains X'00000000', the application program has not supplied a command completion appendage. The protocol driver performs no further action for this CCB and does not interrupt the application program. In this case, the application program must monitor the CCB_RETCODE for a change from X'FF', indicating that the adapter has completed the command and updated the return code.

If the protocol driver immediately determines that the adapter cannot execute the command, it sets the CCB_RETCODE field with the error code.

[1] These are registers in the Intel** microprocessors of an IBM PC or PS/2.

There are some commands that execute entirely in the workstation and do not use the adapter hardware. When this is the case, the following happens:

- The completion code is set when the protocol driver returns from the interrupt that initiated the command.
- If the command completion appendage is defined, it is given control before the protocol driver returns from the interrupt.

This is an exception and is explained with the command descriptions to which it applies.

Appendages

User-supplied appendages provide exit points from the protocol driver. These appendages are short subroutines that may improve the application program's ability to handle information or events. See Chapter 4, "NETBIOS," for routines used with NETBIOS.

To ensure the integrity of the system, the appendages should have the following characteristics:

- The amount of code executed should be limited, because this routine is an I/O appendage. The appendage is used because a point has been reached where information should be saved for subsequent use.
- When the appendage is entered, the keyboard and DOS timer are disabled, and no more interrupts can be serviced from this adapter until the appendage is completed.
- When control passes to the appendage, interrupts are disabled, and it appears to the appendage that the appendage was entered through an 8086 INT instruction. The stack is established so that an 8086 IRET instruction properly returns control and restores flags.

  When appendage processing is complete, the appendage code must execute the 8086 IRET as the last instruction.

  The protocol driver saves all registers on the stack before giving control to the appendage. Twenty-four bytes of the stack are used by the protocol driver when processing the adapter interrupt. When the appendage is entered, there are 242 bytes of stack space available.
- Execution of the IRET instruction by the appendage returns control to the protocol driver at the point at which it had transferred control to the appendage. The protocol driver restores all registers and returns control to the program that was originally interrupted.

Upon entry to the appendages the following things happen:

- The CX register contains the adapter number.
- The CS register points to the appendage code (current segment).
- The SS and SP registers define the current stack.
- Other specific appendage descriptions define other registers.

The types of user appendages are:

Command completion appendage

A per-command exit that allows asynchronous command completion. The application program can provide several command completion appendages and selectively point to a specific one in each CCB.

The address in the CCB_CMD_CMPL field of the related CCB (which should not be X'00000000', indicating no appendage) indicates the entry point.

The address of the CCB that the adapter completed is in registers ES and BX. The return code is in CCB_RETCODE and the AL register (AH=X'00').

Data received appendage

The RECEIVED_DATA field of the parameter table of the RECEIVE command defines this appendage. The address of the CCB is placed in registers DS and SI. The address of the first receive data buffer is placed in registers ES and BX.

Exception or Status conditions

These appendages are a set of exit points that allow the protocol driver to report hardware and software error conditions and certain status information to the user. When any exception state occurs, all pending adapter commands have the CCB_RETCODE field of their CCBs set for the appropriate reason, and are queued and passed to the exception appendage. The command completion appendage is not taken. See the CCB_POINTER field description on page 2-20 for more about queues. Following are the exception or status condition appendages.

PC-detected error appendage

This appendage is defined in the PC_ERROR_EXIT field of the CCB for a DIR.INITIALIZE command and a DIR.MODIFY.OPEN.PARMS command, or in the PC_ERROR_EXIT field of the DIRECT_PARMS table of a DIR.OPEN.ADAPTER command.

The protocol driver passes parameters to the appendage on entry. Register CX contains the adapter number. Register AX contains the error code. See "PC System Detected Errors" on page B-63 for the code meanings.

Network status appendage

This appendage is defined in the NETW_STATUS_EXIT field of the CCB for a DIR.INITIALIZE command and a DIR.MODIFY.OPEN.PARMS command, or in the NETW_STATUS_EXIT field of the DIRECT_PARMS table of a DIR.OPEN.ADAPTER command.

The protocol driver passes parameters to the appendage on entry. Register CX contains the adapter number. Register AX contains the network status. See "Network Status" on page B-52 for the code meanings.

Adapter check appendage

This appendage is defined in the ADAPTER_CHECK_EXIT field of a DIR.INITIALIZE command and a DIR.MODIFY.OPEN.PARMS command, or in the ADAPTER_CHECK_EXIT field of the DIRECT_PARMS table of a DIR.OPEN.ADAPTER command. See pages 3-11, 3-16, and 3-17 for the descriptions of these commands. See "Token-Ring Network Adapter Check Reason Codes for All CCBs" on page B-50 for the reason code meanings.

Adapter open errors take the normal command completion appendage.

DLC status appendage

This appendage is defined in the DLC_STATUS_EXIT field of the CCB parameter table for a DLC.OPEN.SAP command.

The protocol driver passes parameters to the appendage on entry. Register CX contains the adapter number. Register AX contains the DLC status code. Register SI contains a user-defined value from the USER_STAT_VALUE field in the parameter table of the DLC.OPEN.SAP command. Registers ES and BX point to the DLC status table. See to "DLC Status Codes" on page B-28 for the code meanings and the DLC status table.

Local Area Network Support Program (CCB1) Control Blocks

This section describes the format of the CCB1 for DOS. The content of the first field indicates to the protocol driver which type of interface the application program will use. If the first field contains either X'00' or X'01', the block is considered to be a CCB and either the direct interface or the DLC interface is used. In the case where that field is less than X'03', it is considered to be the CCB adapter field. The values X'02' and X'03' cannot be used; they are reserved, and the protocol driver returns an error code if they are used.

If the first field contains a byte greater than X'0F', the NETBIOS interface is used and the control block is considered to be an NCB. The NCB is described under "NCB Field Explanations" on page 4-3. NETBIOS must be loaded before a NETBIOS command is issued, or the protocol driver returns an X'FB' return code.

Table 2-1. CCB1 Command Control Block

Offset	Field Name	Byte Length	8086 Type	Description
0	CCB_ADAPTER	1	DB	Adapter 0 or 1
1	CCB_COMMAND	1	DB	Command field
2	CCB_RETCODE	1	DB	Completion code
3	CCB_WORK	1	DB	Adapter support software work area
4	CCB_POINTER	4	DD	Queue pointer and protocol driver work area
8	CCB_CMD_CMPL	4	DD	Command completion user appendage
12	CCB_PARM_TAB	4	--	Parameters or pointer to CCB parameter table

Note: Use this control block definition with both the direct interface and the DLC interface.

For a complete description of the fields, see "CCB Field Explanations" on page 2-19.

Programming Conventions with OS/2

The following sections describe the OS/2 DLR interface and the OS/2 DD interface provided with OS/2 EE, OS/2 LAN Server, or NTS/2.

The OS/2 DLR Interface

DLR calling conventions, command completion, and control block structures are described in the topics that follow.

Execution of Multiple Commands with the DLR Interface

To enhance the performance of the protocol driver, the application program can request execution of multiple commands with a single invocation of the protocol driver. This is accomplished by allowing application programs to queue CCB requests using the CCB_POINTER field of the CCBs. All queued CCBs are then linked with the CCB_POINTER fields pointing to the next CCB in the queue of CCBs. All commands queued for a single invocation must be for the same adapter. If the chained commands are not for the same adapter, the processing of the queue is terminated with all unprocessed commands returned to the user with the CCB_RETCODE of the first CCB returned set to X'5F'.

If an error is found while processing the queue of CCB requests, the remaining CCBs in the queue are not processed. The CCB containing the error plus a queue of the commands that have not been processed is returned to the application program using a Bad Command Pointer. The address of this pointer is passed to the DLR interface. See "DLR Interface (CCB2) Calling Conventions" on page 2-8 for more information.

Not all commands should be chained, especially commands that are dependent on the completion of other commands. For example, do not chain together DLC.OPEN.SAP, DLC.OPEN.STATION, and DLC.CONNECT.STATION commands because the later two commands are dependent upon the completion of the first command. However, you can chain together synchronous commands that execute in the workstation and commands that are independent of each other without resulting in time-related errors. Commands such as TRANSMIT, READ, BUFFER.FREE, BUFFER.GET, and commands for different station IDs can be issued successfully in a chain of commands.

User's Data Segment Restrictions with the DLR Interface

All application program data segments referred to in a call to the protocol driver must be accessible by the OS/2 process that is calling the protocol driver. That is, the CCB and all data areas pointed to by the CCB (for example, SAP buffers and transmit buffers) must be in the current local descriptor table (owned by the current OS/2 process) so the protocol driver has access to these areas in order to lock and update them.

All control blocks (for example, parameter tables) referenced with an offset must be located in the same segment as the associated CCB. As a performance suggestion, control blocks should be limited to 1 segment (64 KB).

User's Data Segment Guidelines with the DLR Interface

Place data to be processed by the protocol driver in separate segments. You can separate segments that are supplied by application programs in requests from data segments that contain static local variables of the application programs. Allocating separate segments for data accessed by the protocol driver is not a requirement. However, using separate segments limits the size of the data area that the protocol driver locks. The protocol driver locks data to ensure that OS/2 does not move data to disk or other segments when the data is not frequently accessed. At interrupt time, the protocol driver must have immediate access to user data associated with CCB requests. By locking the segments containing data, the protocol driver ensures that OS/2 does not disturb the data. However, locking segments consumes RAM area; therefore locking fewer segments allows more physical RAM to be available. The following data structure types should each be assigned to a separate allocated segment:

- Receive buffers
- Transmit buffers
- CCBs and associated parameter tables.

To maximize overall performance, an application program should use a single segment to contain the above data structures. By providing a single segment, all the associated data structures are locked as long as one of the structures remains in the domain of the protocol driver. With the single segment locked, no additional locks are required when requesting the protocol driver's services, unless the CCB of the request references a segment outside of the locked segment. Also, performance is enhanced if all data structures start on an even byte boundary.

DLR Interface (CCB2) Calling Conventions

To request the DLR interface, you must place two parameters onto the stack. These parameters are:

- CCB2 Pointer: Pointer to CCB to be processed
- Address of Bad Command Pointer: Address of a returned pointer value.

  This parameter is the address of a DD (Double Word) pointer or Bad Command Pointer. If an error is found with a command that is included in a chain of CCBs passed to OS/2 EE or other OS/2 support software on a single invocation, the address of the CCB containing the error with all unprocessed CCBs still chained to it is returned. The Bad Command Pointer points to these commands upon return from the OS/2 EE invocation. The Bad Command Pointer is valid only when the immediate return code in AX is set to X'0003'. If an immediate return code is set, the semaphores specified in the CCB are not cleared.

The application program must have access to the segments referenced by all pointers (for example, the CCB and all associated data structures).

When a list of commands are passed to OS/2 EE on a single invocation, all chained commands must be for the same adapter.

For a given application program to make a request to OS/2 EE, it must:

- Push the selector of a CCB2 onto the stack.
- Push the offset of a CCB2 onto the stack.
- Push the selector of address for Bad Command Pointer onto the stack.
- Push the offset of address for Bad Command Pointer onto the stack.

- Invoke the OS/2 EE DLR interface (ACSLAN module name within the ACSLAN dynamic link library) using the Call Far/Return Far interface. ACSLAN removes the push parameters from the stack before returning to the caller.

Upon return, the AX register contains one of the following immediate return codes:

X'0000'	Command accepted or command was completed successfully
X'0001'	Invalid CCB pointer
X'0002'	CCB in error
X'0003'	CCB in error; check Bad Command Pointer
X'0004'	Unexpected operating system return code; adapter closed
X'0005'	Unexpected operating system return code
X'0006'	Invalid command pointer.

If, on return, the AX register is set to X'03', check the Bad Command Pointer. If the Bad Command Pointer is non-zero, it points to a queue of commands that were not processed, excluding the first CCB in this queue. The first of these commands had an error and has the CCB_RETCODE field set. All other commands that are queued to the first CCB have not been processed; therefore, no return code is provided.

If, on return, the AX register is set to X'04', the adapter will be closed. If a READ command is outstanding, application program resource information is returned to the READ command's parameter table. See the READ command description on page 3-86 for more information.

If, on return, the AX register is set to X'05', check the CCB_WORK field for an OS/2 EE function and the word at offset 24 of the CCB for the OS/2 EE return code of the failing request.

DLR Interface (CCB2) Command Completion

User notification flags, semaphores, and return codes are used to post events to the application programs. The choice of how each event is posted is left up to the application program. Some events can be posted differently.

User notification flags are used to post events as follows:

1. To request that information relating to an event be placed onto a completion list managed by the protocol driver. By placing an event onto the completion list, you can retrieve information relating to the event at a later time.
2. To enable notification of critical exceptions that result in the adapter closing. By enabling critical exception notification, you can be alerted of the event if a READ command is pending before the event occurs.

All flags are 4 bytes long and are preserved across invocations. The flags are *set* whenever non-zero values are used. If needed, these flags can contain user-specific information. However, if the flags are equal to X'00000000', the flags are considered *not set*. Nothing is placed onto the completion list, and no notification is given for critical exceptions.

If an event is placed onto the completion list, the application program must issue a READ command to remove the event from the completion list. See the READ command description on page 3-86.

Semaphores can be provided with all commands. Upon completion of a command, the protocol driver clears the semaphore to alert the application program of the command completion.

The application program can also poll the completion return code for each command to determine when the command has completed its function.

User Notification Flags: Set the following user flags to place event information onto the completion list.

Command completion flag (CCB_CMPL_FLAG)
For each command issued to the adapter support software, a CCB_CMPL_FLAG is included in the CCB. If this flag is set to a non-zero number, the address of the CCB is queued onto a completion list upon completion of the command.

Receive data flag (RECEIVE_FLAG)
For each RECEIVE command issued to the protocol driver, a RECEIVE_FLAG is included in the CCB's parameter table. If this flag is set to a non-zero number, the first receive buffer address is queued onto a completion list upon reception of data.

DLC status change flag (DLC_STATUS_FLAG)
For each DLC.OPEN.SAP command issued to the protocol driver, a DLC_STATUS_FLAG is included in the CCB's parameter table. If this flag is set to a non-zero number, detection of a DLC status change results in a copy of the current DLC status table being queued onto a completion list. See "DLC Status Codes" on page B-28 for a list of DLC status codes.

User exception flags (for non-critical exceptions)
Use the DIR.SET.EXCEPTION.FLAGS command to set the user exception flags. If these flags are set to a non-zero number, the appropriate information is queued onto a completion list upon detection of an exception condition. See the DIR.SET.EXCEPTION.FLAGS command description on page 3-38. The following is a list of the user exception flags that enable information to be placed onto the completion list for non-critical exceptions:

- NETWORK_STATUS_FLAG
- SYSTEM_ACTION_FLAG.

Set the following user flags with the DIR.SET.EXCEPTION.FLAGS command to enable notification of critical exceptions:

- ADAPTER_CHECK_FLAG
- NETWORK_STATUS_FLAG
- PC_ERROR_FLAG
- SYSTEM_ACTION_FLAG.

If these flags are set to a non-zero number and a READ command that requests notification of critical exceptions is pending, then, when a critical exception condition is detected, the appropriate information is copied to the pending READ command's parameter table. The READ command is then posted as defined by the READ command's CCB.

Note: For more information on the exception information, see “Adapter Check for CCB3” on page B-49, “Network Status for CCB3” on page B-54, “PC System Detected Errors for CCB3” on page B-67, and “System Action Exceptions for CCB3” on page B-72.

Posting of Events: All commands issued to the protocol driver can be posted using any combination of the three post mechanisms: setting the user flag along with issuing a READ command, waiting on a semaphore, or polling the return code set in the CCB. However, posting exceptions and DLC status changes must be implemented with setting the user flags and issuing a READ command.

If the associated user flag is not set, the event is not queued to the completion list and the user must use one of the other mechanisms to post the event.

For each command, a semaphore can be passed to the protocol driver within the CCB. Upon completion of the command, the protocol driver clears the semaphore. Thus, if an application program has a thread waiting on the semaphore, the thread is dispatched. If neither the user flag nor the semaphore is used, the application program must poll the return code of the CCB to determine when the command has completed.

If both the user flag and a semaphore are used, the semaphore is cleared when the command routine is completed, and the CCB of the command is also placed onto the completion list. By clearing the semaphore, the application program is notified of the command completion. However, the application program must still issue a READ command (if one is not already pending for the given event) to remove the CCB from the completion list.

There are two special cases where events are chained together to lessen the number of READ commands that must be issued to retrieve information from the completion list. Both RECEIVE and TRANSMIT commands can be issued specifying that event information relative to each command be linked together. If the user chooses to have receive data frames chained together and the completed TRANSMIT commands' CCBs chained together, the following applies:

- If a RECEIVE command is issued with the CMD_CMPL_FLAG and the RECEIVE_FLAG set, all receive data is placed onto the completion list. If the RECEIVE command is issued requesting that received frames be chained and a READ command is issued with one or more frames being received that meet the READ command's requirements, the frames are chained together using the first receive buffer of each frame.
- If multiple TRANSMIT commands containing the CMD_CMPL_FLAG set have been executed and have requested chaining upon completion, then whenever a READ command is issued and more than one TRANSMIT CCB is executed that matches the READ command's requirements, the TRANSMIT CCBs are chained together using the CCB_POINTER of the TRANSMIT CCB and are returned to the application program.

DLR Interface (CCB2) Control Blocks

Table 2-2 describes the fields of the CCB2 for OS/2 EE 1.3 and LAPS using the DLR interface.

Table 2-2. CCB2 Command Control Block

Offset	Field Name	Byte Length	8086 Type	Description
0	CCB_ADAPTER	1	DB	Adapter number
1	CCB_COMMAND	1	DB	Command field
2	CCB_RETCODE	1	DB	Completion code
3	CCB_WORK	1	DB	Adapter support software work area
4	CCB_POINTER	4	DD	Queue pointer and protocol driver work area
8	CCB_CMPL_FLAG	4	DD	Command completion flag
12	CCB_PARM_OFFSET	2	DW	Offset to CCB2 parameter table
14	CCB_PARAMETER_1	2	DW	Parameter or reserved for an application program
16	CCB_SEMAPHORE	4	DD	Command post semaphore
20	CCB_APPL_ID	1	DB	Application program ID
21	CCB_READ_FLAG	1	DB	READ chained to CCB flag
22	CCB_APPL_KEY	2	DW	Application program key code
24	CCB_PARAMETER_2	2	DW	Parameter for System Key or reserved

Note: Use the control block definition above with both the direct interface and the DLC interface.

For a complete description of the fields, see "CCB Field Explanations" on page 2-19.

The OS/2 DD Interface

DD calling conventions, command completion, and control block structures are described in the topics that follow.

User's Data Segment Restrictions with the DD Interface

All application program data segments referred to in a call to the protocol driver must be accessible by the OS/2 process that is calling the adapter.

Make all segment references with either global descriptor table (GDT) selectors or 32-bit physical addresses (CCB addresses, SAP buffer addresses, transmit buffer addresses) and lock all segments using the OS/2 Device Help routine. See "RECEIVE.MODIFY" on page 3-103 and descriptions of the TRANSMIT commands on page 3-107 for use of the 32-bit physical addresses. Map all other commands and data areas using the GDT selectors.

Locate all control blocks referenced with an offset (for example, parameter tables) in the same segment as the associated CCB.

User's Data Segment Guidelines with the DD Interface

Place data that is processed by the protocol driver in separate segments from data segments that contain static local variables of the application programs. Allocating separate segments for data accessed by the protocol driver is not a requirement. However, using separate segments limits the size of the data area that the protocol driver uses. The protocol driver assumes that data segments are locked before they are called from an application program. Therefore, it is the application program's responsibility to lock the data areas. At interrupt time, the protocol driver must have immediate access to user data associated with CCB requests. By having the application program lock the segments containing data, the protocol driver is ensured that OS/2 does not disturb the data. Following is a list of data structure types that you should assign to a separate allocated segment:

- Receive buffers (use GDT selectors)
- Transmit buffers (use 32-bit physical addresses for better performance)
- CCBs and associated parameter tables (use GDT selectors).

DD Interface (CCB3) Calling Conventions

To request the DD interface, the application program device driver must place the address of the CCB3 to be executed by the DD interface into registers ES and BX, and push an invocation code of X'0000' onto the stack. The application program device driver then issues a Call Far instruction to the OS/2 EE DD interface intercommunication entry point.

Note: The application program device driver must do an ATTACH OS/2 Device Help Function call to obtain the interdevice driver communication entry point of the OS/2 DD interface (LANDD$). Refer to the OS/2 command for details of the call.

Upon return from the DD interface, all registers contain their original values with the exception of the AX register. The AX register contains the immediate return code.

For a given application program's device driver to make a request to the DD interface, it must:

- Set register BX to the address offset of the CCB3 to be executed
- Set register ES to the address selector of the CCB3 to be executed
- Push an invocation code of zero onto the stack
- Call the OS/2 DD interface (LANDD$) interdevice driver communication entry point using the Call Far/Return Far interface.

Upon return, the AX register contains one of the following immediate return codes:

X'0000'	Command accepted or command was completed successfully
X'0001'	Invalid CCB pointer
X'0002'	CCB in error
X'0004'	Unexpected operating system return code; adapter closed
X'0005'	Unexpected operating system return code
X'0007'	Invalid invocation code.

If, on return, the AX register is set to X'04', the adapter will be closed. If the application program has an appendage for workstation-detected errors, then the function code of the Device Help request and the return code of the request that failed are included in the information returned in the 20-byte information table of the workstation-detected appendage.

DD Interface (CCB3) Command Completion

Use user appendages and return codes to post events to the application programs. The choice of how each event is posted is left up to the application program.

Events are posted to user appendages by one of the following methods:

- Appendages request that information relating to an event be passed to the application program by an appendage call from the protocol driver when an event has occurred.
- Appendages enable notification of critical exceptions that result in the adapter closing. By enabling critical exception notification, an application program can be alerted of the event by an appendage call from the protocol driver.

The application program must pass the offsets to the appendages for these different events:

- Completion of commands
- Reception of data
- DLC status change
- The following exceptions
  - Adapter Check
  - Network Status
  - PC Detected Error
  - System Action.

The application program can also poll the completion return code for each command to determine when the command has finished processing.

Posting of Events: If event information is to be posted to the application program, the user appendages described in this section must be defined by passing an offset to the protocol driver through the different commands. The protocol driver enters the application program's device driver with a Call Far instruction using the application program's device driver entry point obtained when the DIR.OPEN.ADAPTER command is issued. The application program's device driver must return using a Return Far instruction.

Note: For all appendage calls and the RECEIVE.MODIFY subroutine call, an invocation code of X'0001' is pushed onto the stack. The called device driver must remove the invocation code from the stack.

Command Completion

For each command issued to the protocol driver, a CCB_APPNDG_OFFSET is included in the CCB. The offset is a 2-byte Define Word (DW) field that the protocol driver uses for the address of the appendage that the protocol driver passes to the application program in register DI when the application program device driver is called.

When the protocol driver calls the application program at the address obtained by the ATTACHDD function, it provides the following information:

- An invocation code of X'0001' was pushed onto the stack. Before returning to the protocol driver, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the command completion appendage.
- Register DS contains the application program's device driver protect mode data segment selector.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of the CCB.
- Registers DX and SI contain the address of a 12-byte information table relating to the command. These registers are set to zero if no information is available to be returned to the application program. See Table 2-4 on page 2-22 for the information table.
- Register AX contains the CCB return code.

Receive Data

When an application program issues a RECEIVE command to the protocol driver, a RCV_DATA_APPNDG is included in the CCB parameter table of the RECEIVE command. The offset is a 2-byte DW field that the protocol driver passes to the application program's device driver when receive data is available and the application program's device driver is called.

When the protocol driver calls the application program at the address obtained by the ATTACHDD function, it provides the following information:

- An invocation code of X'0001' was pushed onto the stack. Before returning to the protocol driver, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the RECEIVE appendage.
- Register DS contains the application program's device driver protect mode data segment selector.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of the first SAP buffer.
- Registers AX and SI contain the address of the RECEIVE command's CCB for which receive data has been processed.

DLC Status

When an application program issues a DLC.OPEN.SAP command to the protocol driver, a DLC_STATUS_OFFSET is included in the CCB parameter table of the DLC.STATUS command. The offset is a 2-byte DW field that the protocol driver passes to the application program's device driver when DLC status data is available and the application program's device driver is called.

When the protocol driver calls the application program at the address obtained by the ATTACHDD function, it provides the following information:

- An invocation code of X'0001' was pushed onto the stack. Before returning to the protocol driver, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the DLC status appendage as defined by the DLC.OPEN.SAP command.
- Register DS contains the application program's device driver protect mode data segment selector.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of a 20-byte information table. See "DLC Status Codes" on page B-28 for description of DLC status codes.
- Register AX contains the DLC status code.
- Register SI contains the USER_STAT_VALUE defined with the DLC.OPEN.SAP command.

Exception Conditions

The user appendages associated with exception conditions are set using the DIR.SET.EXCEPTION.FLAGS command. See the DIR.SET.EXCEPTION.FLAGS command description on page 3-1

An appendage offset is included in the DIR.SET.EXCEPTION.FLAGS command for each of the conditions below. The offset is a 2-byte DW field that the protocol driver passes to the application program's device driver when an exception occurs and the application program's device driver is called.

When the protocol driver calls the application program at the address obtained by the ATTACHDD function, it provides the following information for the different exception conditions:

Adapter Check

See "Adapter Check for CCB3" on page B-49 for more information.

- An invocation code of X'0001' was pushed onto the stack. Before returning to the protocol driver, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the adapter check appendage as defined by the DIR.SET.EXCEPTIONS.FLAG command.
- Register DS contains the application program's device driver protect mode data segment selector.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of a 20-byte information table.
- Register AX contains the reason code.

Network Status

See "Network Status for CCB3" on page B-54 for more information.

- An invocation code of X'0001' was pushed onto the stack. Before returning to the protocol driver, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the network status appendage as defined by the DIR.SET.EXCEPTIONS.FLAG command.
- Register DS contains the application program's device driver protect mode data segment selector.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of a 14-byte information table.
- Register AX contains the network status.

PC Detected Error

See "PC System Detected Errors for CCB3" on page B-67 for more information.

- An invocation code of X'0001' was pushed onto the stack. Before returning to the protocol driver, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the workstation-detected error appendage as defined by the DIR.SET.EXCEPTIONS.FLAG command.
- Register DS contains the application program's device driver protect mode data segment selector.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of a 20-byte information table.
- Register AX contains the error code.

System Action

See "System Action Exceptions for CCB3" on page B-72 for more information.

- An invocation code of X'0001' was pushed onto the stack. Before returning to the protocol driver, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the system action appendage as defined by the DIR.SET.EXCEPTIONS.FLAG command.
- Register DS contains the application program's device driver protect mode data segment selector.
- Register CX contains the adapter number.

- Registers ES and BX contain the address of a 14-byte information table.
- Register AL contains the System Action ID.
- Register AH contains the SAP value associated with the System Action ID.

If the associated user appendage is not defined, the event is not posted to the user.

DD Interface (CCB3) Control Blocks

This table contains a description of the format of the CCB3 for OS/2 EE and other OS/2 support programs using the DD interface.

Table 2-3. CCB3 Command Control Block

Offset	Field Name	Byte Length	8086 Type	Description
0	CCB_ADAPTER	1	DB	Adapter number
1	CCB_COMMAND	1	DB	Command field
2	CCB_RETCODE	1	DB	Completion code
3	CCB_WORK	1	DB	Adapter support software work area
4	CCB_POINTER	4	DD	Queue pointer and protocol driver work area
8	CCB_APPNDG_OFFSET	2	DW	Offset to CCB3 completion appendage
	-reserved-	2	DW	Reserved for application program
12	CCB_PARM_OFFSET	2	DW	Offset to CCB3 parameter table
14	CCB_PARAMETER_1	2	DW	Parameter or reserved for an application program
16	CCB_RESOURCE_ID	2	DW	Resource ID of application program process
	-reserved-	2	DW	Reserved for application program
20	CCB_APPL_ID	1	DB	Application program ID
21		1	DB	Reserved for application program
22	CCB_APPL_KEY	2	DW	Application program key code
24	CCB_PARAMETER_2	2	DW	Parameter for System Key or reserved

Note: The above control block definition is to be used with both the direct interface and the DLC interface.

For a complete description of the fields, see “CCB Field Explanations” on page 2-19.

Control Blocks for All CCBs

The application program must prepare a control block to request an activity from the adapter when using a protocol driver from one of the IBM support programs. When the protocol driver analyzes the control block, it can determine which interface is needed by the content of the first bytes.

The content of the control blocks is explained in "Local Area Network Support Program (CCB1) Control Blocks" on page 2-6, "DLR Interface (CCB2) Control Blocks" on page 2-12, and "DD Interface (CCB3) Control Blocks" on page 2-18.

CCB Field Explanations

The following CCB field descriptions apply to all three CCBs (CCB1, CCB2, and CCB3) unless otherwise stated.

CCB_ADAPTER

Explanation: This field defines which adapter is to be used.

For CCB1: The adapter number has the following binary format:

`B'0000ceaa'`, where

- `0000` (bits 4-7) are always zero.
- `c` (bit 3) is the common storage bit. If this bit is on, the CCB was issued from common memory. This bit is used when running under the 3270 Workstation Program. See "3270 Workstation Support" on page D-8 for more information.
- `e` (bit 2) is the extended CCB bit. If this bit is on, the CCB is extended 2 bytes, with the last 2 bytes indicating bank switching information. This bit is used when running under the 3270 Workstation Program. See "3270 Workstation Support" on page D-8 for more information.
- `aa` (bits 0 and 1) are the adapter number, B'00' through B'11' (only B'00' and B'01' are current valid adapter numbers). It must be either X'00' to use the primary adapter or X'01' for the alternate adapter.

If bank switching in the 3270 Workstation Program is not implemented, the extended bank switch information in the CCB is set to a null value: X'FFFF'.

Common and extended bits apply to the 3270 Workstation Program only. The following list describes these bits:

- If the common and extended bits are both zero (B'00'), the CCB was issued by an application program in one of the memory banks.
- If the common and extended bits are zero and one (B'01'), the CCB is a *pseudo-CCB*. This means that a CCB issued by an application program in one of the memory banks was substituted with an internally generated CCB containing the extended memory bank information.
- If the common and extended bits are one and zero (B'10'), the CCB was issued by code in the common storage area. NETBIOS issues this bit value for all CCBs except transmits of user-defined data. Common

storage is accessed by the protocol driver independent of the currently active bank.

- If the common and extended bits are one and one (B'11'), the CCB was issued by code in the common storage area. The CCB has extended memory bank information. NETBIOS issues this bit value for CCBs that transmit user-defined data.

If the value is greater than X'0F', the control block is an NCB. Values of X'02' and X'03' are reserved and, if used, a CCB_RETCODE of X'1D' is returned. Values greater than X'03' cause the NETBIOS interface to be used. See Chapter 4, "NETBIOS," for more information about NETBIOS.

When used with a 3270 PC, the adapter number of a CCB has two additional bits defined and has the format described above.

CCB_COMMAND

Explanation: This field indicates the command to perform. A value of X'FF' is a permanently defined invalid command code. See Appendix A, "Valid Commands," for reserved and valid commands.

CCB_RETCODE

Explanation: This field contains the completion code as provided by the protocol driver. For all commands, this field is set to X'FF' by the protocol driver when the CCB is received. While the field is X'FF', the application program must not alter the CCB or any associated data. When the adapter completes the command, the protocol driver sets this field to the appropriate completion code. For all commands, X'00' means successful completion. See "CCB Return Codes Listed by Interface" on page B-2 for descriptions of all return codes.

CCB_WORK

Explanation: This field is a work area for the protocol driver to use.

CCB_POINTER

Explanation:

For CCB1: While the CCB_RETCODE is X'FF', the protocol driver uses this field for command processing.

The application program uses this field as follows:

- When the adapter is closed, the application program interrogates this field to find the next command (CCB) in a queue of pending commands.
- When a DLC link station is sending I frames, multiple transmissions are acknowledged at one time. All acknowledged I frames are queued and presented at one time to the application program. That is, the protocol driver issues an interrupt providing a return code in one CCB. The CCB_POINTER field of that CCB contains the address of a queue of CCBs containing an appropriate return code. This continues until a CCB_POINTER field is zero, ending the queue.

For CCB2: While CCB_RETCODE is X'FF', the OS/2 EE DLR interface can use this area for command processing.

Application programs use this field under the following circumstances:

- When it is necessary for the application program to request that multiple commands be processed as a result of a single invocation, the CCB_POINTER is used to chain CCB2 requests.
- When it is necessary for the application program to have a READ chained to the CCB2 to be used for its completion, the READ CCB2 address is placed in CCB_POINTER and the CCB_READ_FLAG is set to a non-zero value.
- When the adapter is closed, a chained list of pending commands is presented to the user through the CCB_POINTER of
  - A CCB whose address is placed into the READ command's CCB parameter table
  - A DIR.CLOSE.ADAPTER command that has been completed successfully.
- If transmissions specify that completed transmission request CCB2s be chained, the CCB2s are linked together upon completion using the CCB_POINTER. For this case the READ command must be used to retrieve the completed command's CCB2. The address of the first CCB is placed into the READ command's CCB2 parameter table.

For CCB3: While CCB_RETCODE is X'FF', the protocol driver can use this area for command processing.

When the adapter is closed, the application program interrogates this field to find the next command (CCB) in a queue of pending commands.

CCB_CMD_CMPL

Explanation: This field is the address of a user appendage to which the protocol driver goes upon command completion. The appendage allows the user to obtain control after a command has been completed. See "Appendages" on page 2-4 for more information. When the user's appendage receives control at this point, the address of the completed CCB is in registers ES and BX, and the CCB_RETCODE is in register AL. Register AH is X'00'. See "Local Area Network Support Program (CCB1) Command Completion" on page 2-3 for more information.

CCB_CMPL_FLAG

Explanation: This flag indicates whether or not a completed command should be posted using the READ command. The protocol driver checks this field after command completion. If the flag is not zero, the completion is posted to the application program with a READ command. If a READ command is already pending for command completions, this completed command is posted immediately. If there is no pending READ, the command completion is queued internally to the protocol driver that is waiting for a READ for command completions. If the flag is zero, the completion is not posted to the application program with a READ command. After completion, the return code is set.

If the CCB_CMPL_FLAG is not set, the application program either uses the CCB_SEMAPHORE or polls the CCB_RETCODE field for notification of the command completion. It is the application program's responsibility to poll the CCB_RETCODE field for a value other than X'FF'. The value X'FF' signifies that the command is in progress, and that the CCB and its associated data should not be altered.

Notes:

1. See the READ command description on page 3-86 for details on posting command completions.
2. As soon as the protocol driver performs any immediate command processing, the command is queued and control is returned to the application program that is using the protocol driver (CCB_RETCODE is set to X'FF'). At that point, the application program can continue with other processing (not disturbing the CCB or any associated data). When the command is completed, the return code is set and the application program is posted if a READ command is pending.

CCB_APPNDG_OFFSET

Explanation: This field is the offset of a user appendage within the application program device driver's code segment that handles the command's completion. When the command is completed, the protocol driver calls the application program device driver at its intercommunication entry point, pushing the invocation code of X'0001' onto the stack. The appendage offset is passed in the DI register. The application program device driver must call the appendage located at the address offset specified in the DI register.

When the protocol driver calls the application program device driver at the intercommunication entry point, the following information is provided:

- An invocation code of X'0001' has been pushed onto the stack. Before returning to the protocol driver, the application program must remove the invocation code from the stack.
- Register DI contains the appendage offset as defined by each individual command.
- Register DS contains the application program's device driver protect mode data segment selector.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of the completing CCB.
- Registers DX and SI contain the address of a 12-byte information table pertaining to the command which has been executed (see Table 2-4). If no information is available for the command, these registers contain zeroes.
- Register AX contains the CCB3 return code.

Table 2-4 (Page 1 of 2). Command Completion Appendage Information Table

Offset	Field Name	Byte Length	8086 Type	Description
0	CCB_COUNT	2	DW	Count of CCBs chained to EVENT_CCB_POINTER

Table 2-4 (Page 2 of 2). Command Completion Appendage Information Table

Offset	Field Name	Byte Length	8086 Type	Description
2	EVENT_CCB_POINTER	4	DD	Pointer to CCB terminated as a result of the CCB addressed with the ES and BX registers
6	BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR
8	FIRST_BUFFER_ADDR	4	DD	Address of first receive buffer for buffer pool

CCB_PARM_TAB

Explanation: This field points to additional parameters that are command-specific. These parameters are explained with the related command descriptions.

If the parameters required 4 or fewer bytes, they are provided in the CCB_PARM_TAB field instead of in an area pointed to by the field.

CCB_PARM_OFFSET

Explanation: This field points to additional parameters that are command-specific. These parameters are explained with the related command descriptions.

If the parameters require 2 or fewer bytes, they are provided in the CCB_PARM_OFFSET field. If the parameters require more than 2 bytes, the field contains the offset within the selector of the parameter table for this command.

CCB_PARAMETER_1

Explanation: This field can contain another 2 bytes of parameter data to combine with the CCB_PARM_OFFSET field for a total of 4 bytes, or it can contain user-specific data, for example, the segment or selector.

CCB_SEMAPHORE

Explanation: A system semaphore can be used to notify an application program of a command completion. When the command has been executed, the protocol driver clears the CCB_SEMAPHORE field to alert the application program that the command has been completed. To specify a system semaphore, the CCB_SEMAPHORE field must contain a handle of a system semaphore that is returned from OS/2 EE when the system semaphore is created or opened . System semaphore handles provided on the DIR.OPEN.ADAPTER command should be used to obtain optimum performance.

Only the OS/2 EE process that issues the DIR.OPEN.ADAPTER command and provides system semaphore handles can use these handles. All other processes associated with the application program must provide a handle returned from OS/2 EE for the given process when the system semaphore is created or opened.

If the application program does not use a semaphore, the CCB_SEMAPHORE field should be coded as zero. If the CCB_SEMAPHORE field contains an invalid system semaphore handle (excluding zeros), the process is terminated by OS/2 EE whenever the handle is used in an OS/2 EE call.

Notes:

1. The system semaphore must be created so that exclusive ownership is not required.
2. The protocol driver does not set the CCB_SEMAPHORE; it only clears it.

CCB_RESOURCE_ID

Explanation: A resource ID must be used to allow proper clean-up of resources owned by terminating processes of an application program. All control blocks that have been passed to the protocol driver can have a resource ID associated with them. This allows the protocol driver to know which resources are associated with which application programs. This ID is required when an application program consists of more than one OS/2 EE process. When a process terminates, the application program's device driver must clean up the resources associated with the process by issuing the PURGE.RESOURCES command with a PURGE_RESOURCE_ID. All control blocks that have a resource ID matching the PURGE_RESOURCE_ID are freed by the protocol driver.

Memory passed to LANDD$ in the form of CCBs or buffers can be owned by different processes of an application program or by an application program's device driver. Since the LAN device drivers cannot guarantee that the active process is the owner of memory being passed to it, all control blocks and buffers are associated with the resource ID. The resource ID is passed as a parameter with the CCB and associated with the CCB and all other control blocks (logs, buffers) referenced by the CCB.

If an application program consists of more than one OS/2 EE process, it is the application program's responsibility to manage the clean-up of each process's resources. For example, if an application program has two OS/2 EE processes (that have both been allocated memory being used in the application program's SAP buffer pool), when one of the processes ends the application program should notify LANDD$ with the PURGE.RESOURCE command. The resource ID is passed as a parameter to specify which control blocks should be removed (cleaned up) from the LAN device driver's internal queues.

CCB_APPL_ID

Explanation: This field contains the ID of the application program issuing the command. The CCB_APPL_ID is returned on the DIR.OPEN.ADAPTER request and must be used by the application program for all following commands that the application program issues. Some of the CCBs can be issued with the System Key, such as DIR.SET.GROUP.ADDRESS, DIR.READ.LOG and others, in which case the CCB_APPL_ID field is not needed. Otherwise, all of the CCBs (with the exception of the DIR.STATUS command) are required to use CCB_APPL_ID, which is returned on DIR.OPEN.ADAPTER CCB.

Note: System Key is for system administrator use only.

CCB_READ_FLAG

Explanation: An application program can specify that a READ command is chained to this CCB using the CCB_POINTER field. This READ command is used to process the completion of this CCB only, and not other commands that may have been executed previously.

CCB_APPL_KEY

Explanation: This field contains a key code used to provide resource security for application programs. For the given command to succeed, the CCB_APPL_KEY parameter must match the CCB_APPL_KEY code provided by the user on the DIR.OPEN.ADAPTER request. If the user chooses not to use a key code (key code is set to zero) when issuing the DIR.OPEN.ADAPTER command, the CCB_APPL_KEY parameter is not checked by the protocol driver when a request is made. Some commands can be issued with the System Key, such as DIR.SET.GROUP.ADDRESS, DIR.READ.LOG, and others. When this is done the CCB_APPL_KEY field is not checked. If the command is issued without the System Key, the CCB is required to have the same CCB_APPL_KEY as the application program did in the DIR.OPEN.ADAPTER command.

Note: The System Key is for system administrator use only.

CCB_PARAMETER_2

Explanation: Command parameters (2 bytes of parameter data) are usually used for the System Key parameter.

For System Key: This parameter is used to enable only a system administrator to perform operations that could stop ring communication for application programs.

This key code is used for the following tasks:

- Change functional address
- Change group addresses
- Reset selected SAPs and stations or all SAPs and all stations
- Relinquish ownership of direct stations
- Force a physical close for an adapter
- Force the adapter to initialize
- Read and reset adapter error and direct interface logs.

The System Key is not typically used by application programs, but rather for maintenance and problem determination.

This command can be issued by a system administrator with the System Key as defined by configuration parameters. If the adapter has not been opened by the system administrator, only polling of the CCB_RETCODE field can be used for posting of this command completion. CCB2 can post the command completion also using an OS/2 system semaphore. If the adapter has been opened and an application program ID has been returned, this command can be posted like any other command. See "System Action Exceptions for OS/2 EE 1.3" on page B-69 for more information on the System Key.

If the System Key is not used, this field should be coded as X'0000'.

Addressing

Each adapter using the network has an address called the node address. When frames are sent on the network by adapters, the frame contains two of these addresses: a source address and a destination address. The frame is sent to the destination address adapter by the source address adapter.

All network addresses, including Ethernet addresses, must be in non-canonical format at the CCB interface. This form of address specification is different from the canonical bit ordering used by Ethernet, where the bits of each byte appear reversed from their representation on the LAN medium. For example, the locally administered group address X'C00000000FBC' (non-canonical) would be represented as X'03000000F03B' using canonical bit ordering.

Additional address and link information to be used in other transmission layers may be included in the frame following the LAN addresses. Additional addressing is used in the implementation of both the LLC and NETBIOS. The LLC sublayer uses an address known as a service access point (SAP), described in the next section. NETBIOS addressing is described further in Chapter 4, "NETBIOS."

An adapter is provided with a permanent, universally administered address. Additionally, the application program has the capability to provide a temporary replacement for this address and to provide a group address for the adapter.

Note: The NODE_ADDRESS field can only be changed by the configuration parameters when CCB2 or CCB3 is used. Group and functional addresses can be set and used as destination addresses; they cannot be used as source addresses.

Refer to the *IBM Token-Ring Network Architecture Reference* for uses and restrictions for these types of addresses.

Adapter Addresses

Adapters are able to identify the intended recipient of any frame because each adapter has a unique address. There are two types of addresses: universally administered and locally administered.

All Token-Ring and PC Network adapters manufactured by IBM have universally administered addresses encoded on them. These addresses use the following format:

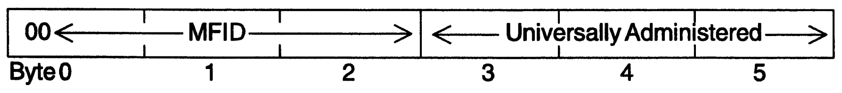


Figure 2-1. Universally Administered Adapter Address

The first two bits (B'00') indicate that the address is a universally administered address. The MFID field contains the manufacturer's identification. The IEEE ensures that every universally administered address is unique.

The format of the universally administered address for Ethernet Networks is different. These addresses use the following format:

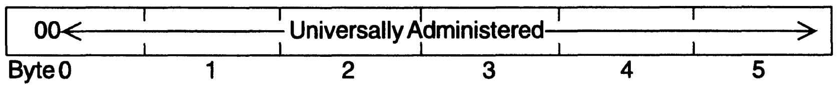


Figure 2-2. Ethernet Universally Administered Adapter Address

The first two bits (B'00') indicate that the address is a universally administered address. The IEEE assures that every universally administered address is unique.

The application program can assign locally administered addresses. A locally administered address overrides the universally administered address encoded on the adapter. These addresses use the following format:

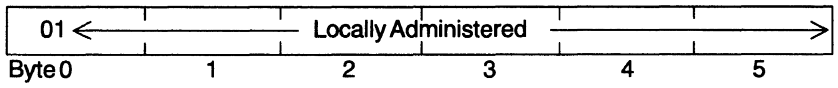


Figure 2-3. Locally Administered Adapter Address

The first two bits (B'01') identify the address as locally administered. Because of restrictions placed on addresses by certain networking protocols, you should assign addresses in the range 0000 0001 to 7999 9999 9999 (decimal). Your network administrator is responsible for preserving the uniqueness of these addresses.

For additional information about maintaining addresses, refer to the *IBM Token-Ring Network Administrator's Guide.*

Stations, SAPs, and IDs

The direct station, service access points (SAPs), and link stations can be defined to the IEEE 802.2 interface. They are referred to by the STATION_ID field in command descriptions. The direct station, which is automatically assigned when the CCB interface is opened, is referred to by three station IDs. This station is automatically prepared to receive frames from the network when the DIR.OPEN.ADAPTER command (for DOS) or the DIR.OPEN.DIRECT command (for OS/2) is issued; however, the application program must issue a RECEIVE command to make the information available at the direct interface. The three station IDs, which differ only in how they receive frames, are described in the following list:

Station ID	**Description**
X'0000'	This station ID of the direct station receives all frames (MAC and non-MAC) not directed to other defined stations. This station can transmit MAC and non-MAC (data) frames.
X'0001'	This station ID of the direct station receives MAC frames and transmits either MAC or non-MAC frames. The PC Network and Ethernet Networks do not use MAC frames.
X'0002'	This station ID of the direct station receives non-MAC frames and transmits either MAC or non-MAC frames.

SAPs

SAPs can be opened for communications with SAPs in other devices connected to the network. It is possible to design an application program to communicate with any SAP. However, the NETBIOS interface is designed to communicate only with the NETBIOS SAP (X'F0').

Both uses of the DLC interface, connectionless and connection operations, use SAPs for communication on the network. An application program can open several SAPs for a workstation and each SAP can have several link stations opened that are associated with it. These link stations can then be directed to connect to link stations in other adapters (or even the same adapter). A SAP can operate in one of two ways:

- To have exchange ID (XID) command frames handled by the LLC sublayer
- To have XID command frames passed to the application for handling.

When the SAP is opened, an option is set that defines the handling of received XID commands. XID responses are always passed to the application program. See "Transmitting, Receiving, and Buffers" on page 2-39.

When an application program opens a SAP, the application program assigns a SAP value (in the SAP_VALUE field) and the protocol driver assigns a station ID. Communication between the application program and the protocol driver refers to a SAP by the 2-byte station ID. For SAPs, the first byte of the STATION_ID field identifies the SAP and the second byte is zero. When a link station is associated with a SAP, a new station ID is assigned. That station ID is 2 bytes: the first byte identifies the associated SAP, and the second byte is the link station number. All link station numbers are unique for a given adapter. Both SAPs and link stations are referenced by using the STATION_ID field. For example, X'0100' represents a SAP and X'0108' represents a link assigned to that SAP.

When a SAP is used to communicate with another SAP, the application program provides the station ID to identify the local SAP and provides a destination address and SAP value (SAP_VALUE) to identify the remote SAP. The same information is needed to open a link station. When both devices have a SAP and link station opened, a connect command actually initiates the link connection.

The LLC header part of a frame contains two 1-byte SAP values: the destination SAP (DSAP), and the source or sending SAP (SSAP). The SAP value actually uses only 7 of the 8 bits. One bit of the SSAP is used to indicate whether the frame is an LLC command or response, and one bit of the DSAP is used to identify the target SAP as a group or individual SAP. The bit used is the low-order bit of the SAP value supplied by the user in the various SAP commands. An individual SAP value is always even, and a group SAP value is always odd.

A group SAP is a set of open individual SAPs. The global SAP is a special case of a group SAP for which the set consists of all currently open individual SAPs. When a frame is sent to a group SAP, a copy of the frame is passed to each individual SAP that is a member of the group. Note that frames cannot be sent from a group SAP because the bit that indicates group or individual has a different meaning in the SSAP.

A SAP can be opened as an individual SAP, a group SAP, or both. This is done using the option bits and the SAP value provided in the parameter list of the DLC.OPEN.SAP command. If the individual option is chosen, frames containing a

DSAP equal to the SAP value with the low order bit off are accepted. If the group option is chosen, frames containing a DSAP equal to the SAP value with the low-order bit set on are accepted. If both options are selected, both odd and even DSAP values are recognized.

An additional option bit is used to specify group membership. A SAP opened with the individual option can be designated to be a member of one or more group SAPs, provided that the group member option is also selected. The group SAPs to which it will belong can be specified in the DLC.OPEN.SAP and DLC.MODIFY commands. Membership is deleted using the DLC.MODIFY command before closing the SAP. The only restriction is that all members of a particular group must have selected the same XID handling option.

See Figures 1-3 and 1-7 starting on page 1-11. For a transmitted frame, the destination address in the LAN header is the remote node address. The source address in the LAN header is the local node address. The DSAP is the destination SAP_VALUE (RSAP_VALUE), and the SSAP is the local SAP value. At the receiving end, the interpreting of local and remote fields is exchanged. For example, the destination address field is the local node address of the receiving adapter.

The maximum number of user-assigned group and individual SAPs possible is 127. The maximum number of link stations per adapter is 255 (all of which may be assigned to the same SAP). However, RAM and memory constraints limit the number of SAPs and link stations that can be open at one time.

More information about these SAPs and links is included with related command descriptions.

SAP Assignments

The following SAPs are opened automatically:

- Null SAP X'00'
- Global SAP X'FF'.

The Null SAP is opened automatically (with a SAP value of X'00'). It represents the LLC as a whole. The Null SAP provides the ability to respond to remote nodes even when no SAP has been activated. This SAP supports only connectionless service and responds only to XID and Test Command frames. The Null SAP is not accessible to the local application program.

The Global SAP is opened automatically (with a SAP value of X'FF'). It is a group SAP with all open individual SAPs as members. XID, TEST, and UI frames directed to the Global SAP are passed to each open SAP in turn, with the DSAP field in the received frame buffer set equal to the receiving individual SAP value, where they are handled according to frame type.

Note: In both DOS and OS/2, an NDIS token-ring network adapter with LLC microcode discards frames sent to the global SAP. These frames are not received by the protocol driver.

SAP X'E4' is opened automatically for the PC Network or Ethernet Networks. This SAP is used for management.

If NETBIOS is used, it uses SAP X'F0'. SAPs X'F1' to X'FE' are reserved.

Figure 2-4 on page 2-30 shows that you can have more than one SAP on a station and more than one link on a SAP. Type 2 connection-oriented communication is shown by the solid lines. Type 1 connectionless communication is indicated by the dotted line that connects two SAPs rather than two stations.

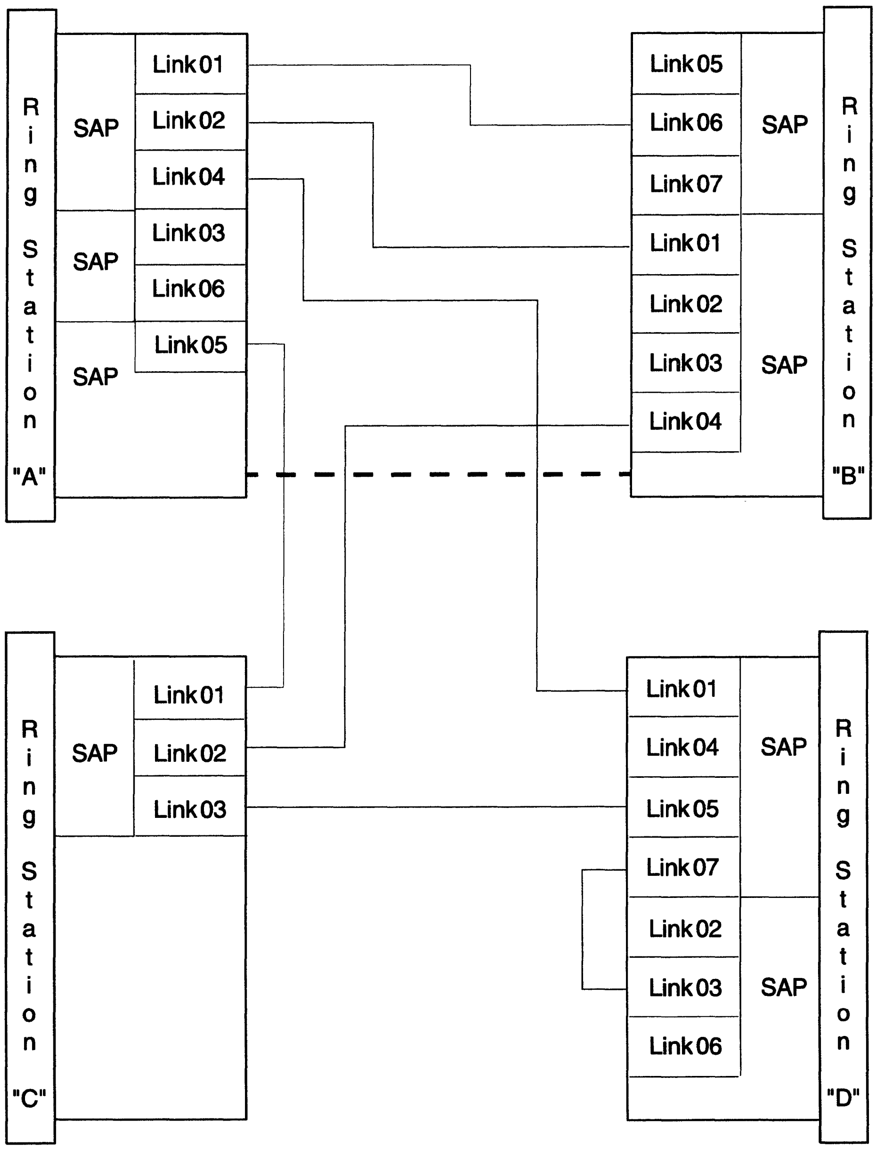


Figure 2-4. SAPs and Link Stations

DLC

The DLC interface provides an interface to application programs using the LLC sublayer of data link control protocol. This interface can be used in two ways:

- For IEEE Type 1 communication, which is connectionless communication between devices providing no guarantee of delivery
- For IEEE Type 2 communication, which is connection-oriented services.

Much of the communication overhead function is provided by the protocol driver, so that programming is simplified. Refer to the *IBM Token-Ring Network Architecture Reference* for more about communication using DLC and LLC.

Types of Service

The IBM Token-Ring Network and IBM PC Network support IEEE 802.2 Type 1 and Type 2 service as described in the *IBM Token-Ring Network Architecture Reference.* Type 1 is connectionless service allowing transmission and receipt of UI frames, XID frames, and TEST frames. Type 1 uses unnumbered LPDUs. Frames sent using this type of service are not followed by a transmission from the receiving device verifying correct receipt and sequence of events unless provided by an application program in that device. Recovery and retry actions must be controlled by the application program. Type 2 is connection-oriented service providing guaranteed delivery and using numbered LPDUs.

Command Sequences

When LLC protocols are used, commands must be issued in certain sequences to obtain the desired result.

In all cases, the adapter must be initialized and opened prior to the use of any TRANSMIT and RECEIVE commands.

Possible command sequences are listed in Tables 2-5, 2-6, and 2-7.

Table 2-5. Start Command Sequence for CCB1 and CCB3

CCB1 and CCB3 Command	**Comments**
DIR.INITIALIZE	Select, clear, and test the adapter
DIR.OPEN.ADAPTER	Make ready, set parameters, and connect the adapter to the ring **For CCB3:** Perform a logical open and return parameters
DIR.SET.EXCEPTION.FLAGS	**For CCB3:** Enable exception notification
DIR.OPEN.DIRECT	**For CCB3:** Open the direct station for one application program
RECEIVE	Prepare for received data for the direct station
DLC.OPEN.SAP	Allocate a SAP
RECEIVE	Prepare for received data for this SAP
DLC.OPEN.STATION	Prepare a link station
RECEIVE	Prepare for received data for this link station
DLC.CONNECT.STATION	Initiate the communication link with the remote station

For CCB1 only: The DIR.INITIALIZE command should be issued only if the adapter is known to be dedicated to the application program. The return code on the DIR.INTERRUPT command can be used to determine if a DIR.INITIALIZE command is needed.

Table 2-6. Start Command Sequence for CCB2

CCB2 Command	Comments
DIR.OPEN.ADAPTER	Perform a logical open and return parameters
DIR.STATUS	Obtain the current status of the network
DIR.SET.EXCEPTION.FLAGS	Enable exception notifications
READ	Allow for posting of exception events
DIR.OPEN.DIRECT	Open the direct station for one application program
RECEIVE	Receive for direct stations
READ	Allow posting for direct stations, receive data
DLC.OPEN.SAP	Allocate a SAP
READ	Allow posting for DLC status change
READ	Allow posting for SAP station and its link station receive data
DLC.OPEN.STATION	Prepare a link station
DLC.CONNECT.STATION	Initiate the communication link with the remote station

After this sequence has been completed, the application program can transmit and receive data on a link station in the following manner:

Receiving Data
Check the RECEIVE command's return code if no appendage was used, or check to see if the appendage routine has received data. After moving data from the receive buffer, issue a BUFFER.FREE command to return the buffer to the pool.

Transmitting Data
Any buffer can be used or the application program can issue a BUFFER.GET command to obtain enough buffers to contain the transmit data, move the data to the buffers, and issue a TRANSMIT.I.FRAME command. Issue a BUFFER.FREE command when the transmit is completed to return buffers that were originally retrieved from the buffer pool, with the exception of buffers referenced by the XMIT_QUEUE_TWO fields of the TRANSMIT command.

When preparing to leave the application program, or when network communication is no longer required, the following commands should be issued:

Table 2-7. End Command Sequence

Command	Comments
DLC.CLOSE.STATION	Close the link station
DLC.CLOSE.SAP	Close the SAP
DIR.CLOSE.DIRECT	**For CCB2 and CCB3:** Close the direct station
DIR.CLOSE.ADAPTER	Remove from the ring

Link Station States

LLC Type 2 protocol maintains primary and secondary states for each link station. Only one of the primary states can be active at a time. If the application program issues a command to a link station that is not valid for the current state, the command is rejected with a return code of X'41'. The DLC.MODIFY and DLC.FLOW.CONTROL commands are accepted in all states. If a link station is not established, there is no control block, and no primary and secondary states exist. Therefore, the link station is "non-existent."

Changes in the DLC status of the link station are reported to the interface. See "DLC Status Codes" on page B-28 and "Suggested Actions in Response to DLC Status" on page B-31 along with the following state information.

The link station primary states are:

Link Closed
All received frames are ignored in this state. The state is entered when:

- A DM response to a SABME or DISC has been queued for transmission. The close command that caused the transmission is executed when the transmission is completed.
- A DM or UA response to a DISC has been received. The close command that caused the transmission is executed when the transmission is completed.
- A reset command has been received, but a transmission has already been queued or is in process and must be completed before the link station can be released.

Disconnected
The following two frames are not ignored in the disconnected state:

- DLC frames with the poll bit set and for which a DM is transmitted
- A SABME which is reported to the workstation.

A DLC.CLOSE.STATION or DLC.CONNECT.STATION command is accepted when this state is active. The state is entered when:

- A DLC.OPEN.STATION command has been accepted.
- A SABME for a previously non-existent station has been accepted.
- A DM response or DISC command from the paired station has been received.
- The retry count has been exhausted because of timeouts.

Disconnecting
This state is normally entered when the initial in-process return code is supplied after receipt of a DLC.CLOSE.STATION command. This state is maintained until one of the following occurs:

- Either a UA or DM response to the transmitted DISC command is received.
- A SABME command is received, and a DM response has been successfully transmitted.
- The retry count expires.

Exit from this state is normally to link-non-existent or link-closed state. Since the DLC.CLOSE.STATION command remains in process while the link is in disconnecting state, no other commands are accepted. All received frames other than SABME, DISC, UA, and DM are ignored while the link station is in this state.

This disconnecting state can also be entered upon expiration of the retry count in FRMR received. In this case, exit is to the disconnected state.

Link Opening

Unexpected received frames are ignored in this state. The link station enters this state when a DLC.CONNECT.STATION command is issued by the workstation. Before entering this state, the adapter transmits either a SABME command, or a UA response if a SABME has been received from the remote station.

If a SABME was transmitted, the adapter expects a UA response. On receipt of the UA response, the adapter transmits an RR command-poll and changes to the link-opened (checkpointing) state.

If a UA was transmitted, the adapter expects either a supervisory command or an information frame. After reception, the adapter changes its state to a link-opened state (possibly with remote busy).

If the expected frame is not received and the retry count is exhausted, the link is returned to the disconnected state unless a SABME has been received.

The DLC.CONNECT.STATION command is completed with a successful return code or with an indication that the remote station failed to respond.

Resetting

All received frames except DISC, DM, FRMR, and SABME are ignored. Only DLC.CLOSE.STATION and DLC.CONNECT.STATION commands are accepted by the adapter when in this state. The state is entered when a SABME command frame is received from the remote station when the link is open and not in disconnected state or link-closed state.

Frame Reject Sent

All received frames except DISC, DM, FRMR, and SABME are ignored. Only DLC.CLOSE.STATION and DLC.CONNECT.STATION commands are accepted by the adapter when in this state. The state is entered when an illegal frame is received and an FRMR frame has been transmitted.

Frame Reject Received

All received frames except DISC, DM, and SABME are ignored. Only DLC.CLOSE.STATION and DLC.CONNECT.STATION commands are accepted by the adapter when in this state. The state is entered when an FRMR frame has been received.

Link Opened

This is the only state that allows information transfer and accepts TRANSMIT commands. In this state, the adapter handles sequential delivery and acknowledgment of information frames, together with retransmission if required. The state is entered when the adapter passes from the link-opening state after the SABME-UA exchange, which completes the connection protocol.

The link station secondary states are:

Checkpointing
A poll is pending; I frame transmission is suspended.

Local Busy (user)
A DLC.FLOW.CONTROL command with a set-local-busy option has been accepted. I frame reception is suspended until a DLC.FLOW.CONTROL command with a reset-local-busy (user) option has been accepted.

Local Busy (buffer)
An out-of-buffer return code has been set by the workstation in response to a request for data service on a receive. I frame reception is suspended until a DLC.FLOW.CONTROL command with a reset-local-busy (buffer) option has been accepted.

Remote Busy
An RNR frame has been received from the remote station. I frame transmission is suspended until a receive ready or reject response, a SABME command, or an in-sequence I response frame with the F bit set to B'1' has been received.

Rejection
An out-of-sequence I frame has been received from the remote station and an REJ transmitted. I frame reception is suspended until an in-sequence I frame or a SABME has been received.

Clearing
A poll is pending, and a confirmation of clearing local busy is required after the response is received.

Dynamic Window
The remote station is on a different ring, and there appears to be congestion through the bridge or bridges.

Timers

The DLC functions use three timers:

T1	Response timer
Ti	Inactivity timer
T2	Receiver Acknowledgment timer.

Refer to the *IBM Token-Ring Network Architecture Reference* for details about the timers.

The rate at which each of these timers is stepped and the value at which they time out are selectable by parameters. The rate of stepping is referred to as the "tick" and is defined as follows:

For CCB1: Define the "tick" with fields in the DLC open parameters provided to the adapter with the DIR.OPEN.ADAPTER command.

For CCB2 and CCB3: Define the "tick" with the configuration parameters at system initialization time.

Each timer requires a short timer tick (TICK_ONE) and a long timer tick (TICK_TWO). The period between timer ticks is some number of 40-ms intervals.

The timer value, or count at which it expires and interrupts the adapter, is selected with parameters provided to the adapter when a DLC.OPEN.SAP, DLC.OPEN.STATION, or DLC.MODIFY command is issued.

A timer value is selected by using a number between 1 and 10. Each timer is divided into two groups of possible values:

- If the number selected is between 1 and 5, the short timer tick (TICK_ONE) is used and is referred to as group 1. The timer value is equal to the number selected multiplied by the short timer tick value (*number_selected* x *short_tick_value*).
- If the number selected is between 6 and 10, the long timer tick (TICK_TWO) is used and is referred to as group 2. The timer value is equal to number selected minus 5 multiplied by the long timer tick value ((*number_selected* - 5) x *long_tick_value*)

Therefore, there are three timers, with two rates selectable for each, which provide a total of six parameters to be selected.

Each DLC.OPEN.SAP command sets the values for the three timers for that specific SAP using the rates selected for the entire adapter. For example, if the value of the T1 timer in one SAP is 4 and the value for the T1 timer in another SAP is 7, the short rate of stepping is selected for the Response timer on the one SAP, and the long rate of stepping is selected for the Response timer in the other SAP. The group-2 timer values should be used when longer delays are expected, such as when in a multi-ring environment.

The time of expiration is not exact, but falls into a range starting with the calculated time.

For example, if a given timer chose the following tick values:

Group-1 tick: 200 ms
Group-2 tick: 1 second

then the following timer values would be available:

Group 1		**(in ms)**	**Group 2**		**(in seconds)**
Number	**Value**	**Actual Range**	**Number**	**Value**	**Actual Range**
1	200	200- 400	6	1	1-2
2	400	400- 600	7	2	2-3
3	600	600- 800	8	3	3-4
4	800	800-1000	9	4	4-5
5	1000	1000-1200	10	5	5-6

The next section includes guidelines for selecting timer values.

Guidelines for Selecting Parameter Values

Following are some basic guidelines to consider when selecting parameter values for the network adapter. There are several basic parameters that can affect the performance obtained when you are using the DLC functions of the adapter. In most cases the default values provide efficient operation. See "Timers" on page 2-35 and the parameter fields of the DLC.OPEN.SAP, DLC.MODIFY, and DLC.OPEN.STATION commands. Table 2-8 on page 2-37 lists the parameters that are outlined here.

Table 2-8. DLC Parameters

Parameter	**Pseudo Parameter**
Response Timer (T1)	The TIMER_T1 parameter of a DLC.OPEN.SAP, DLC.MODIFY, or DLC.OPEN.STATION command
Inactivity Timer (Ti)	The TIMER_Ti parameter of a DLC.OPEN.SAP, DLC.MODIFY, or DLC.OPEN.STATION command
Receiver Acknowledgment Timer (T2)	The TIMER_T2 parameter of a DLC.OPEN.SAP, DLC.MODIFY, or DLC.OPEN.STATION command
Maximum Length I-Field (N1)	The MAX_I_FIELD parameter of a DLC.OPEN.SAP, DLC.MODIFY, or DLC.OPEN.STATION command
Maximum Number of Retransmissions (N2)	The MAX_RETRY_CNT parameter of a DLC.OPEN.SAP, DLC.MODIFY, or DLC.OPEN.STATION command
Number of I-Format LPDUs Received before Sending Acknowledgment (N3)	The MAX_IN parameter of a DLC.OPEN.SAP, DLC.MODIFY, or DLC.OPEN.STATION command
Number of Acknowledgments Needed to Increment Ww (Nw)	The MAXOUT_INCR parameter of a DLC.OPEN.SAP, DLC.MODIFY, or DLC.OPEN.STATION command
Maximum Number of Outstanding I-Format LPDUs (TW)	The MAXOUT parameter of a DLC.OPEN.SAP, DLC.MODIFY, or DLC.OPEN.STATION command

Response Timer (T1)

The Response timer (T1) is maintained by the sending adapter whenever an I-format LPDU or a command LPDU with the poll bit set to B'1' is sent. If this timer expires before a response is received, the sending adapter solicits remote link station status by sending a supervisory command LPDU with the poll bit set to B'1'. The T1 timer value should, therefore, be greater than the total delay time that the frame might encounter within the sending node, the network, and the receiving node. Normal settings for the T1 parameter should be in the range of 1 to 2 seconds. For instance, a setting above 2 seconds can result in noticeable delays to those responses that must be retransmitted (typically less than 3 percent of the total frames).

Inactivity Timer (Ti)

The Inactivity timer (Ti) runs whenever the Response timer (T1) is not running. If this timer expires, the link may have been lost. The Inactivity timer (Ti) value should be five to ten times greater than the T1 value, and it is recommended that the minimum be 30 seconds. The default is 30 seconds.

Receiver Acknowledgment Timer (T2)

A link station starts T2 when an I-format LPDU is received into workstation memory. T2 is stopped when an acknowledgment is sent either with an outgoing frame or when the number of I-format LPDUs received before the sending acknowledgment (N3) value is reached. If T2 expires, the link station must send an acknowledgment as soon as possible. The value of T2 must be less than that of T1 to ensure that the remote link station receives the delayed acknowledgment before T1 expires. Typical values for T2 are 80 to 256 ms.

Maximum Length of I-Field (N1)

The Maximum Length of I-Field (N1) parameter is used primarily to enable a pair of stations to establish the maximum size frame that can be received by either station. For example, one station may be able to transmit and receive frames up to 2 KB each while the other can only send and receive frames of 1 KB or smaller. Under no circumstance should the N1 value exceed the total amount of receive memory available.

A key factor in selecting the N1 value is the receive buffer capacity of the destination adapter. Server devices, for example, can support several sessions concurrently, and therefore have a more limited buffer capacity than a workstation.

Table 2-9. Maximum I-field Length for Network Adapters

Token-Ring Network at 4 Mbps	Token-Ring Network at 16 Mbps	PC Network Adapter	Ethernet
4464 Bytes	17 960 Bytes	2042 Bytes	1490 Bytes

N1 should never exceed 2042 bytes with the Token-Ring Network PC Adapter or with PC Network adapters, or 1490 bytes with Ethernet adapters. N1 values smaller than 512 bytes can result in a perceived decrease in station-to-station response times.

Maximum Number of Retransmissions (N2)

The Maximum Number of Retransmissions (N2), or MAX_RETRY_CNT, defines the maximum number of attempts in which a sending adapter performs the checkpoint procedure following the expiration of the T1 timer. The combination of T1 and N2 values should be great enough to allow for error detection and recovery on the network. This count also prevents continual retransmission of the same I frame.

Typical values for N2 are 10 or less.

Maximum Number of Outstanding I-Format LPDUs (TW) and Number of I-format LPDUs Received before Sending Acknowledgment (N3)

The TW and N3 counts should be considered together since they establish the ratio of acknowledgment frames to I-Format LPDU frames. However, the N3 value should be compared only with the TW value of the remote link station, not the local station. The values of TW and N3 can affect the response perceived by the user in some cases. However, in most instances, the default values provide the best general performance. The following guidelines should be considered:

- The TW count allows the sender to transmit TW frames before it is forced to halt and wait for an acknowledgment. Therefore, the receiver should be able to absorb that number of frames, either in its SAP buffers or within the buffers in workstation memory. A small value of TW reduces the chances that frames are retransmitted due to buffer congestion at the receiver. The TW-to-N3 ratio thus provides a flow control mechanism to prevent overruns at the receiver.
- The TW value should always be at least twice the N3 value. Network response can be severely degraded if N3 exceeds TW.

- Very little network overhead or adapter processing is required to send or receive an acknowledgment frame. Therefore, every frame can be acknowledged without a perceptible degradation in performance.
- Even though the maximum values allowed for TW and N3 are 127 each, practical values should not exceed 8 for TW or 4 for N3.

  Note: For more information, refer to the *IBM Token-Ring Network Architecture Reference*.

Working Window (Ww), and Window Increment (Nw)

There are two counts associated with the dynamic window algorithm for flow control. The purpose of the dynamic window algorithm is to allow the sending station to temporarily reduce the transmit window (Tw) whenever network or receive adapter congestion is resulting in lost frames. By temporarily reducing the window size, the flow of frames over that link is reduced, thus permitting the congested node to recover from the temporary overload.

The DLC interface provides an interface to application programs using the LLC sublayer of DLC protocol. The interface can be used in two ways:

- For IEEE Type 1 communication, which is connectionless communication between devices providing no guarantee of delivery
- For IEEE Type 2 communication, which is connection-oriented services.

Much of the communication overhead function is provided by the adapter and the protocol driver, both of which permit simple programming. The *IBM Token-Ring Network Architecture Reference* contains more information about communication using DLC and LLC.

Transmitting, Receiving, and Buffers

Data exchanged between application programs is sent on the network in frames. A frame consists of headers and data. All frames have a LAN header, although the format is slightly different depending on whether the IBM Token-Ring Network, IBM PC Network, or Ethernet Networks are being used. MAC frames, transmitted only on the IBM Token-Ring Network, consist of the LAN header and a data field. In non-MAC frames the LAN header is followed by the DLC header. The data field that follows the LAN header in MAC frames, and the DLC header in non-MAC frames, is the data provided by the application program. (The data field itself may contain further headers, in a format defined by the application programs exchanging the data.)

The length of the LAN header varies depending on the length of the Routing Information field (if present). The length of the DLC header varies depending on the frame type: the I-frames used to exchange data in Type 2 protocols have a 4-byte DLC header; the frames used for Type 1 protocols have a 3-byte header.

The type of frame being transmitted also affects the amount of information that the application program must provide, and how it is provided. There are three cases:

MAC frames	The application program provides the complete LAN header and the data field. The source address in the LAN header is overwritten with the address being used by the adapter.
I-frames	The application program provides only the data when it wants to transmit a frame. The information required to build the headers is provided in the DLC.OPEN.STATION and DLC.CONNECT.STATION commands.
Direct frames	The application program provides the LAN header and the data in its own buffers. It provides the information needed to build the DLC header with the TRANSMIT command (the remote SAP value and the command type). The source address field in the LAN header is overwritten with the address in use by the adapter.

The application program must have the data, and if required, the LAN header, in buffers prepared in a format understood by the protocol driver. These buffers are in workstation memory belonging to the application program. This memory may be entirely controlled by the application program, or may be given to the protocol driver to manage, in which case it is part of the buffer pool discussed in “Buffer Pools.”

The protocol driver receives frames from the network and moves the frames received from the network to application program buffers, provided that an active RECEIVE command is used to pass the buffers to the application program, and that there is application program buffer space to hold the incoming frame. The application program provides the buffer space in the form of a buffer pool, described in “Buffer Pools.” Once the protocol driver has moved the data to workstation memory, it uses the RECEIVE command (and optionally the READ command for CCB2 users) to tell the application program that it has received a frame. See page 3-86 for a description of the RECEIVE command and page 3-95 for a description of the READ command.

Buffer Pools

A buffer pool is an area of workstation memory provided by the application program to the protocol driver. Each buffer pool is divided into buffers. Received frames are put into buffers from the buffer pool by the protocol driver. These buffers must be returned to the pool (using BUFFER.FREE commands) after the application program has finished with the frame data. When the application program transmits a frame, it can use buffers from a separate area of memory or buffers from the buffer pool (obtained by issuing a BUFFER.GET command). The following commands are associated with generating, defining, and handling buffer pools:

DIR.OPEN.ADAPTER
Allocates direct interface buffer pool (for CCB1 only).

DIR.OPEN.DIRECT
Allocates direct interface buffer pool (for CCB2 and CCB3 only).

DIR.MODIFY.OPEN.PARMS
Changes direct interface buffer pool allocation (for CCB1 only).

DLC.OPEN.SAP
Allocates DLC interface buffer pool for a specific SAP.

BUFFER.GET
Gets one or more buffers from a SAP pool and DIRECT pool.

BUFFER.FREE
Returns one or more buffers to a pool.

RECEIVE
Receives data into buffers.

RECEIVE.MODIFY
Receives data into optional buffers.

TRANSMIT
Sends data from buffers.

Buffer pools can be allocated for every SAP defined to the adapter and for the direct interface direct station at station IDs X'0000', X'0001', and X'0002'. Station ID X'0001' is not used on the PC Network or Ethernet Networks. Every SAP defined can have one pool of buffers defined for its use.

Each buffer pool is independent of the others and has the following characteristics:

- The protocol driver uses these buffers to satisfy the RECEIVE command. Their use is optional for TRANSMIT commands.
- All link stations associated with a specific SAP use the same buffer pool.
- All buffers in a pool have the same length.
- Every buffer has a 12-byte overhead to contain a forward pointer and length information controlled by the protocol driver.
- When you are defining a buffer pool:
  - The buffer length defined must be a multiple of 16 bytes.
  - The user-defined length includes the 12-byte overhead.
  - The minimum user-defined length is 80 bytes (68 data bytes plus 12 bytes of overhead).
- The application program can allow the protocol driver to prepare the buffer pool, or it can take that responsibility itself. Buffer pools are controlled by the protocol driver and individual buffers are obtained and returned by BUFFER.GET and BUFFER.FREE commands. If the application program controls the buffers, it must prepare the control fields in the prescribed format. Since the buffers controlled by the protocol driver must be used for receives, either buffers prepared by the protocol driver are used, or the application program provides a prepared buffer to the protocol driver by issuing a BUFFER.FREE command.

Note: If a SAP has no more available buffers, the reception of data is impacted. See the RECEIVE command description on page 3-95.

Receive Buffers

Data is received from the network into adapter buffers. If there is a RECEIVE command pending for the SAP or link station, the protocol driver moves the data to the appropriate buffer pool in workstation memory. The application program then processes the data and issues a BUFFER.FREE command to return the buffer to the pool. An application program can simultaneously return more than one buffer.

Receive Buffer Formats

The USER_OFFSET field (bytes 8 and 9 of each buffer) allows all buffers to be handled similarly regardless of the amount of information in the buffer prior to the actual received data. When more than one buffer from the buffer pool is used to receive a frame, the format for the first buffer (Buffer 1) is different from the format of the other buffers used to contain this frame. Buffer 2 is an example of the format of the subsequent buffers. By interrogating the contents of the USER_OFFSET field, you can determine the format of any buffer. See the RECEIVE and the RECEIVE.MODIFY command descriptions for buffer assignment.

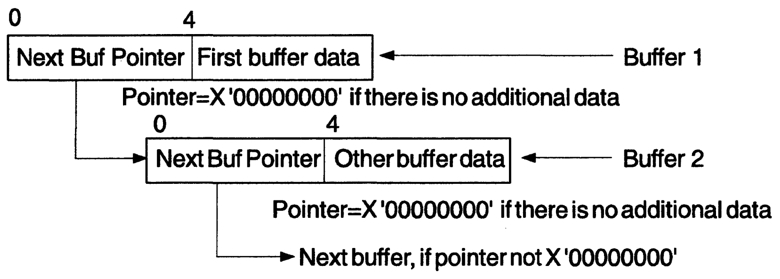

0
4
Next Buf Pointer
First buffer data
Buffer 1
Pointer=X'00000000' if there is no additional data
0
4
Next Buf Pointer
Other buffer data
Buffer 2
Pointer=X'00000000' if there is no additional data
Next buffer, if pointer not X'00000000'

Figure 2-5. Receive Buffer Formats

Table 2-10 and Table 2-11 on page 2-43 show the formats of the receive buffers in detail. The NOT CONTIGUOUS MAC/DATA option means that the received data does not include the DLC header, but begins with the portion of the LAN frame that follows the DLC header; the CONTIGUOUS option means that the received data includes the LAN header and the DLC header.

Table 2-10 (Page 1 of 2). Buffer 1: Option = Not Contiguous MAC/DATA

Off-set	Field Name	Byte Length	8086 Type	Description
0	BUF_POINTER	4	DD	Pointer to the next buffer or X'00000000' if no additional buffers.
4	RCV_LEN	2	DW	Length of entire receive frame.
6	LENGTH_IN_BUFFER	2	DW	Length of data in buffer beginning at byte X (received data).
For CCB1				
8	USER_OFFSET	2	DW	Offset from the beginning of the buffer to the USER_SPACE field. Use this value with the buffer segment-segment+offset.
For CCB2 and CCB3				
8	USER_OFFSET	2	DW	Offset from the beginning of the buffer to the USER_SPACE field. Use this value with the buffer selector-selector+offset.
For all CCBs				
10	USER_LENGTH	2	DW	The length of the USER_SPACE field defined by the USER_OFFSET parameter.
12	STATION_ID	2	DW	Receiving station ID.
14	OPTIONS	1	DB	Option byte from RECEIVE parameter table.
15	MESSAGE_TYPE	1	DB	Type of message received.
16	BUFFERS_LEFT	2	DW	The number of buffers left in the SAP buffer pool.
18	RCV_FS	1	DB	Received Frame Status field.

Table 2-10 (Page 2 of 2). Buffer 1: Option = Not Contiguous MAC/DATA

Off-set	Field Name	Byte Length	8086 Type	Description
19	ADAPTER_NUM	1	DB	Adapter number (0 or 1).
For CCB1				
20	LAN_HEADER_LENGTH	1	DB	The length of the LAN header field (bytes 22-53).
21	DLC_HEADER_LENGTH	1	DB	The length of the DLC_HEADER (bytes 54-57). If the value is X'00', this is for the direct interface.
22	LAN_HEADER	32	DB	The LAN header received with the frame. The actual length is defined by LAN_HEADER_LENGTH.
54	DLC_HEADER	4	DB	The DLC header received with the frame, if applicable. The actual length is defined by DLC_HEADER_LENGTH. (Contents undefined if DLC LENGTH = 0.)
58	USER_SPACE	-	--	An area in the buffer for use by the application program. The length is defined by USER_LENGTH (bytes 10-11).
X	RCVD_DATA	-	DB	The data received following the DLC header in the frame.
For CCB2				
20	NEXT_FRAME	4	DD	A pointer to the next receive frame.
For CCB3				
20		4	DD	Reserved for the application program.
For CCB2 and CCB3				
24	LAN_HEADER_LENGTH	1	DB	The length of the LAN header field (bytes 26-57).
25	DLC_HEADER_LENGTH	1	DB	The length of the DLC_HEADER (bytes 58-61). If the value is X'00', this is for the direct interface.
26	LAN_HEADER	32	DB	The LAN header received with the frame. The actual length is defined by LAN_HEADER_LENGTH.
58	DLC_HEADER	4	DB	The DLC header received with the frame, if applicable. The actual length is defined by DLC_HEADER_LENGTH. (Contents undefined if DLC LENGTH = 0.)
62	USER_SPACE	-	--	An area in the buffer for use by the application program. The length is defined by USER_LENGTH (bytes 10-11).
X	RCVD_DATA	-	DB	The data received following the DLC header in the frame.

Table 2-11 (Page 1 of 2). Buffer 1: Option = Contiguous MAC/DATA

Off-set	Field Name	Byte Length	8086 Type	Description
0	BUF_POINTER	4	DD	Pointer to the next buffer or X'00000000' if no additional buffers.
4	RCV_LEN	2	DW	Length of entire receive frame.
6	LENGTH_IN_BUFFER	2	DW	Length of data in buffer beginning at byte X (received data).
For CCB1				
8	USER_OFFSET	2	DW	Offset from the beginning of the buffer to the USER_SPACE field. Use this value with the buffer segment-segment+offset.

Table 2-11 (Page 2 of 2). Buffer 1: Option = Contiguous MAC/DATA

Off-set	Field Name	Byte Length	8086 Type	Description
For CCB2 and CCB3				
8	USER_OFFSET	2	DW	Offset from the beginning of the buffer to the USER_SPACE field. Use this value with the buffer selector-selector+offset.
For all CCBs				
10	USER_LENGTH	2	DW	The length of the USER_SPACE field defined by the USER_OFFSET parameter.
12	STATION_ID	2	DW	Receiving station ID.
14	OPTIONS	1	DB	Option byte from RECEIVE parameter table.
15	MESSAGE_TYPE	1	DB	Type of message received.
16	BUFFERS_LEFT	2	DW	The number of buffers left in the SAP buffer pool.
18	RCV_FS	1	DB	Received Frame Status field.
19	ADAPTER_NUM	1	DB	Adapter number (0 or 1).
For CCB1				
20	USER_SPACE	-	--	An area in the buffer for use by the application program. The length is defined by USER_LENGTH (bytes 10-11).
X	RCVD_DATA	-	DB	The data received in the frame including the LAN header and the DLC header.
For CCB2				
20	NEXT_FRAME	4	DD	A pointer to the next receive frame.
For CCB3				
20		4	DD	Reserved for the application program.
For CCB2 and CCB3				
24	USER_SPACE	-	--	An area in the buffer for use by the application program. The length is defined by USER_LENGTH (bytes 10-11).
X	RCVD_DATA	-	DB	The data received in the frame including the LAN header and the DLC header.

Buffer Fields Explanations

MESSAGE_TYPE

Explanation: This field indicates the type of message received (byte 15).

X'02'	MAC frame (Direct Station on the Token-Ring Network only)
X'04'	I-frame (Information frame—application program data—link stations only)
X'06'	UI frame
X'08'	XID command (poll bit)
X'0A'	XID command (not poll bit)
X'0C'	XID response (final bit)
X'0E'	XID response (not final bit)
X'10'	TEST response (final bit)
X'12'	TEST response (not final bit)
X'14'	Other; used for non-MAC frame (Direct Station only).

RCVD_FS

Explanation: This field contains the Frame Status (FS) (byte 18).

Note: This field is only valid on the Token-Ring Network.

BIT	MEANING
7	Address recognized indicator (A)
6	Frame copied indicator (C)
5	Reserved
4	Reserved
3	Address recognized indicator (A)
2	Frame copied indicator (C)
1-0	Reserved.

NEXT_FRAME

Explanation: This field contains a pointer to the next frame in the chain.

When the application program specifies that received frames are to be chained, the NEXT_FRAME field of the first buffer of each frame is used to point to the next frame that was received.

USER_SPACE

Explanation: This space can be loaded by the application program. It is not altered by the protocol driver or by the received frame data.

RCVD_DATA

Explanation: The value of this field occupies byte X to end of buffer.

If the option is CONTIGUOUS, this data begins with the LAN header from the received frame.

If the option is NOT CONTIGUOUS, then:

- If MESSAGE_TYPE is X'02' or X'14', this is the data immediately following the LAN header from the received frame.
- If MESSAGE_TYPE is not X'02' or X'14', this is the data immediately following the DLC header from the received frame.

Additional data that is not able to fit into this buffer is placed in buffer 2 and subsequent buffers.

Table 2-12. Buffer 2 and Subsequent Buffers

Off-set	Field Name	Byte Length	8086 Type	Description
0	BUF_POINTER	4	DD	Pointer to the next buffer or X'00000000' if no additional buffers.
4	RCV_LEN	2	DW	Length of entire receive frame.
6	LENGTH_IN_BUFFER	2	DW	Length of data in buffer beginning at byte X (received data).
For CCB1				
8	USER_OFFSET	2	DW	Offset from the beginning of the buffer to the USER_SPACE field. Use this value with the buffer segment-segment+offset.
For CCB2 and CCB3				
8	USER_OFFSET	2	DW	Offset from the beginning of the buffer to the USER_SPACE field. Use this value with the buffer selector-selector+offset.
For all CCBs				
10	USER_LENGTH	2	DW	The length of the USER_SPACE field defined by the USER_OFFSET parameter.
12	USER_SPACE	-	--	An area in the buffer for use by the application program. The length is defined by USER_LENGTH (bytes 10-11).
X	RCVD_DATA	-	DB	A continuation of the data received in the frame.

Transmit Buffers

The application program issues a BUFFER.GET command, or creates a buffer, moves data into the assigned buffer and adds necessary header information, and issues the TRANSMIT command. The protocol driver moves the contents of the buffer into shared RAM or work space and interrupts the adapter to proceed with the transmission. When the TRANSMIT command is executed, the application program issues a BUFFER.FREE command for all buffers originally obtained from the buffer pool except the XMIT_QUEUE_TWO buffer, which is freed by the protocol driver when the transmission is successful (return code is zero).

The total amount of data in all buffers of one issued command must fit into one adapter transmit buffer in shared RAM.

For CCB1: The adapter transmit buffer size is defined by the DIR.OPEN.ADAPTER command DHB_BUFFER_LEN parameter, with the maximum being dependent on the type of adapter being used.

For CCB2 and CCB3: The adapter transmit buffer size is defined by configuration parameters, with the maximum being dependent on the type of adapter being used.

For all LAN network types, the data in the buffer is described as follows:

- Six bytes are used as overhead.
- Fourteen bytes are reserved for the LAN header. On the Token-Ring Network these 14 bytes are used for the access control (AC) byte, the frame control (FC) byte, and the LAN header source and destination address fields. For PC networks and Ethernet, the protocol driver verifies that the FC byte specifies a non-MAC frame, examines the source address to determine the presence of routing information, and uses the destination address to build a LAN header appropriate for the network type.
- The remaining length is reduced if routing information is used (up to 18 bytes) and if a DLC header is included (up to 4 bytes). The routing information field is ignored for Ethernet Networks. For more information on buffer space, see Figures 2-7, 2-8, and 2-9 starting on page 2-49.

  Note: The LAN and DLC headers are not placed in the transmit buffer for an I-frame transmission as a result of a TRANSMIT.I.FRAME command, making an additional 36 bytes available.

Transmit Buffer Formats

Transmit buffers are different for each supported LAN network protocol (Token-Ring, PC Network, and Ethernet). However, to provide compatibility for application programs, the difference is not reflected at the interface to the protocol driver in the IBM support program, so that the transmit buffer is the same for all network protocol types.

For CCB1: For NDIS adapters, the TRANSMIT command is returned with a return code of X'23' if the frame is contained in more buffers than are supported by the NDIS MAC driver. Up to eight buffers are always available. See the maximum number of data blocks defined by the NDIS MAC driver for your adapter to determine how many buffers are supported.

The transmit buffers must be formatted as defined here.

Four groups of buffers are definable by TRANSMIT commands. See "Transmit Command Specifics" on page 3-110. They are:

- XMIT_QUEUE_ONE
- XMIT_QUEUE_TWO
- BUFFER_ONE
- BUFFER_TWO.

XMIT_QUEUE_ONE and XMIT_QUEUE_TWO can each consist of one or more buffers.

Most combinations of XMIT_QUEUE_ONE, XMIT_QUEUE_TWO, BUFFER_ONE, and BUFFER_TWO can be selected for use. BUFFER_TWO can be used only if BUFFER_ONE is also being used. However, they are transmitted sequentially beginning with XMIT_QUEUE_ONE and ending with BUFFER_TWO whenever two or more are selected. XMIT_QUEUE_ONE could contain header information that seldom or never needs modifying. XMIT_QUEUE_TWO could contain data or device-specific header information. BUFFER_ONE could contain the actual data to be transmitted. BUFFER_TWO, if used, might contain additional data. Buffers in XMIT_QUEUE_TWO are freed by the protocol driver if the transmission is successful (return code is zero). These buffers are always returned when you are using OS/2, regardless of the return code.

BUFFER_ONE and BUFFER_TWO are user-defined and can contain any type of information. Any buffer group can be excluded by providing a buffer length of zero in the TRANSMIT command CCB.

The buffers defined by XMIT_QUEUE_ONE and XMIT_QUEUE_TWO are described in Table 2-13.

Table 2-13. Transmit Buffers (XMIT_QUEUE_ONE and XMIT_QUEUE_TWO)

Offset	Field Name	Byte Length	8086 Type	Description
0	BUF_POINTER	4	DD	Pointer to the next buffer or X'00000000' if there are no additional buffers.
4		2	--	Reserved.
6	LENGTH_IN_BUFFER	2	DW	Length of data in buffer beginning at byte 12 plus the USER_LENGTH.
8	USER_DATA	2	DW	Available for user.
10	USER_LENGTH	2	DW	Length of the USER_SPACE starting at byte 12.
12	USER_SPACE	2	DW	USER_SPACE followed by data to be transmitted. If there is no user data, the transmit data starts at byte 12.
14	—	x	DB	Data to be transmitted.

The USER_SPACE can be loaded by the application program. The USER_SPACE information is not transmitted.

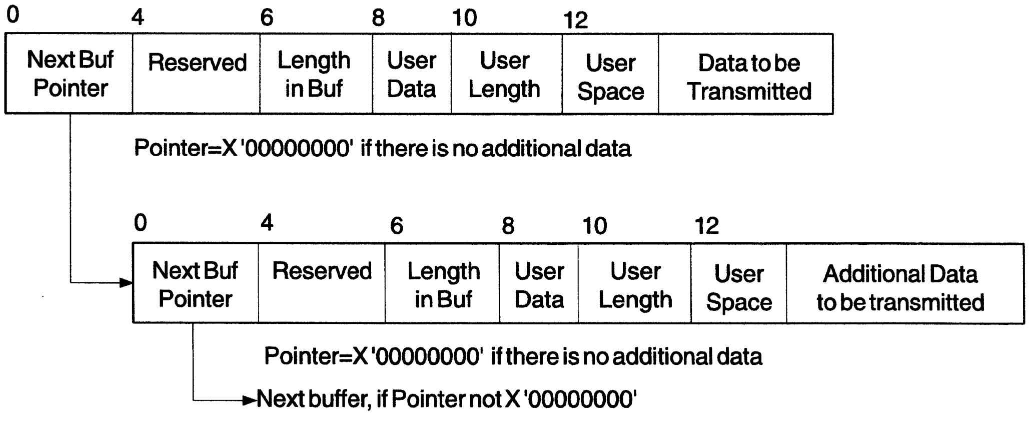
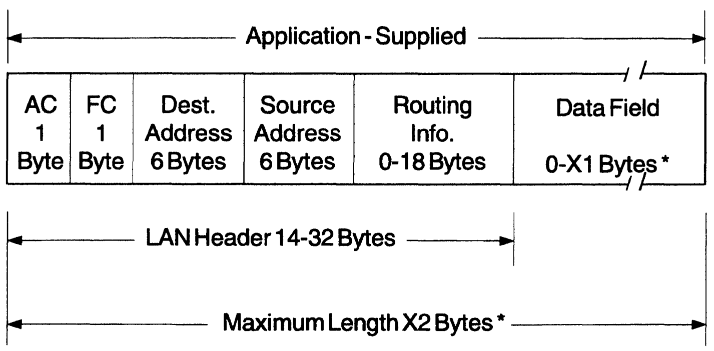

0
4
6
8
10
12
Next Buf Pointer
Reserved
Length in Buf
User Data
User Length
User Space
Data to be Transmitted
Pointer=X'00000000' if there is no additional data
0
4
6
8
10
12
Next Buf Pointer
Reserved
Length in Buf
User Data
User Length
User Space
Additional Data to be transmitted
Pointer=X'00000000' if there is no additional data
Next buffer, if Pointer not X'00000000'

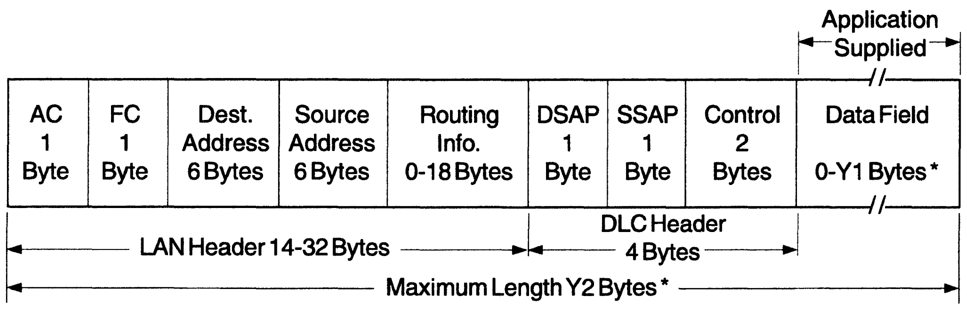
Figure 2-6. Transmit Buffers

The information in the transmit buffer can be transmitted in one of three frame formats: MAC frame, non-MAC I-Frame, or other non-MAC frame. The frame formats are shown in the following figures:



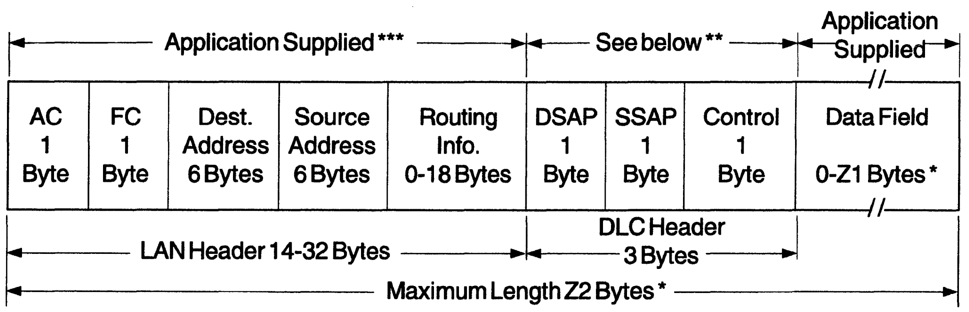
* See Table 2-14 on page 2-50 for the values of X1 and X2.

Figure 2-7. MAC Frame



* See Table 2-14 on page 2-50 for the values of Y1 and Y2.

Figure 2-8. Non-MAC I Frame



* See Table 2-14 for the values of Z1 and Z2.

** The adapter places the DLC header values in 3 bytes of the adapter transmit buffer. None of the application program's buffer space is needed for the DLC header.

*** The LAN header space, including the destination address and routing information fields, is provided by the application program in the first buffer. The adapter fills in the source address, AC, and FC field values. Some adapters may require the application program to provide these fields. Refer to the documentation for your specific adapter type for details.

Figure 2-9. Other non-MAC Frame

Table 2-14. Transmit Buffer Size in Bytes

Value	**Token-Ring Network PC Adapter, PC Adapter II, and Adapter/A**	**Token-Ring Network 16/4 Adapters at 4 Mbps**	**Token-Ring Network 16/4 Adapters at 16 Mbps**	**PC Network Adapters**	**Ethernet DIX V.2 Adapter**	**Ethernet IEEE 802.3 Adapter**
X1	2028	4444	17940	N/A	N/A	N/A
X2	2060	4476	17986	N/A	N/A	N/A
Y1	2042	4458	17954	2042	1493	1496
Y2	2078	4494	17972	2078	1532	1532
Z1	2025	4441	17937	2028	1494	1497
Z2	2060	4476	17972	2063	1532	1532

Chapter 3. The Command Control Blocks

About This Chapter 3-3
Command Descriptions 3-4
- BUFFER.FREE 3-4
- BUFFER.GET 3-5
- DIR.CLOSE.ADAPTER 3-6
- DIR.CLOSE.DIRECT 3-8
- DIR.DEFINE.MIF.ENVIRONMENT 3-9
- DIR.INITIALIZE 3-11
- DIR.INTERRUPT 3-16
- DIR.MODIFY.OPEN.PARMS 3-16
- DIR.OPEN.ADAPTER 3-17
- DIR.OPEN.DIRECT 3-33
- DIR.READ.LOG 3-34
  - Log Formats 3-36
- DIR.RESET.MULT.GROUP.ADDRESS 3-37
- DIR.RESTORE.OPEN.PARMS 3-37
- DIR.SET.EXCEPTION.FLAGS 3-38
- DIR.SET.FUNCTIONAL.ADDRESS 3-39
- DIR.SET.GROUP.ADDRESS 3-41
- DIR.SET.MULT.GROUP.ADDRESS 3-41
- DIR.SET.USER.APPENDAGE 3-43
- DIR.STATUS 3-44
- DIR.TIMER.CANCEL 3-51
- DIR.TIMER.CANCEL.GROUP 3-51
- DIR.TIMER.SET 3-52
- DLC.CLOSE.SAP 3-53
- DLC.CLOSE.STATION 3-54
- DLC.CONNECT.STATION 3-55
- DLC.FLOW.CONTROL 3-56
- DLC.MODIFY 3-57
- DLC.OPEN.SAP 3-60
- DLC.OPEN.STATION 3-66
- DLC.REALLOCATE 3-70
- DLC.RESET 3-71
- DLC.SET.THRESHOLD 3-73
- DLC.STATISTICS 3-74
  - Log Formats 3-75
- PDT.TRACE.OFF 3-76
- PDT.TRACE.ON 3-77
  - Trace Table Formats for the Non-NDIS Token-Ring Network Adapters 3-79
  - Trace Table Formats for the PC Network and NDIS Adapters 3-81
- PURGE.RESOURCES 3-85
- READ 3-86
  - DLC Status Change Events 3-94
- READ.CANCEL 3-94
- RECEIVE 3-95
- RECEIVE.CANCEL 3-102
- RECEIVE.MODIFY 3-103
- TRANSMIT.DIR.FRAME 3-107
- TRANSMIT.I.FRAME 3-108

Transmit Completion 3-108
TRANSMIT.TEST.CMD 3-109
TRANSMIT.UI.FRAME 3-109
TRANSMIT.XID.CMD 3-109
TRANSMIT.XID.RESP.FINAL 3-110
TRANSMIT.XID.RESP.NOT.FINAL 3-110
Transmit Command Specifics 3-110

About This Chapter

This chapter describes all the commands that you can issue to the IEEE 802.2 application interface, both the direct interface and the DLC interface. The commands are listed alphabetically, so you can easily find a particular command.

The commands fall into one of three groups based on which version of the IBM program products you are using.

CCB1	The command control block for the IEEE 802.2 protocol drivers provided with the original Token-Ring Network PC Adapter, Token-Ring Network PC Adapter II, and the IEEE 802.2 protocol drivers supplied with the Local Area Network Support Program.
CCB2	The command control block for the Dynamic Link Routine (DLR) interface provided with OS/2 EE 1.3 and LAPS.
CCB3	The command control block for the Device Driver (DD) interface provided with OS/2 EE 1.3 and LAPS.

Throughout this chapter the term *CCB* is used when information is common to all three groups. You can find detailed information on the types of CCBs used by various program products in "Control Blocks for All CCBs" on page 2-19. This section also describes the various CCB fields.

The commands also fall into four functional groups:

- Commands issued to the direct interface
- Commands issued to the DLC (IEEE 802.2) interface—both SAP and station interfaces
- Commands issued for transmitting and receiving frames
- Commands issued for problem determination (CCB1 only).

The direct interface enables you to perform control functions on the adapter using standard control blocks and parameters. It also enables you to open and close an adapter, obtain error status, and set addresses.

Note: When you are using OS/2 EE, the direct interface commands DIR.OPEN.ADAPTER and DIR.CLOSE.ADAPTER logically open and close an adapter on an application program basis. You must use a System Key to physically close an adapter.

The direct interface also permits transmission of frames directly with no protocol assistance. When you are using the direct interface, an application program can communicate with another application program without links and link stations. The direct interface supports three direct stations, as discussed in "Stations, SAPs, and IDs" on page 2-27. All received frames not directed to an active SAP or link station default to the direct station.

The DLC interface provides application programs with an interface to the logical link control (LLC) sublayer of data link control protocol, which offers both connectionless and connection-oriented services. See "DLC" on page 2-30 for more details.

You can find information on the use of the TRANSMIT and RECEIVE commands, buffer pools, and buffer formats in "Transmitting, Receiving, and Buffers" on page 2-39.

Command Descriptions

All the commands use a control block, as described in Chapter 2, "Programming Conventions for the IEEE 802.2 Interface." All differing uses of variables in the control block and additional control information, such as parameter tables, are included with these command descriptions. A list of the possible return codes for each command is found in "CCB Return Codes Listed by Command" on page B-5.

Each command description begins with a box containing the command name. The hexadecimal number at the top of the box is the command code value. Whenever parameter tables are included, descriptions of the parameters follow the table.

BUFFER.FREE

Hex 27

BUFFER.FREE

Command Description: This command returns one or more buffers to the SAP's buffer pool or to the direct station buffer pool.

Command Specifics: When the buffer is placed back in the buffer pool, bytes 4 and 5 (buffer length) of the buffer are set to zero. This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol drivers.

For CCB1 and CCB3: The command completion appendage is taken if provided.

For CCB2: Either a semaphore or a READ command can be used for command completion.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Table 3-1. CCB Parameter Table for BUFFER.FREE

Offset	Parameter Name	Byte Length	8086 Type	Description
0	STATION_ID	2	DW	SAP/direct station ID; defines the buffer pool.
2	BUFFER_LEFT	2	DW	Number of buffers left in the pool.*
4		4	DB	Reserved.
8	FIRST_BUFFER	4	DD	Address of the first buffer to be added to the pool. The value is set to zero on return.

* Indicates a returned value.

STATION_ID

Explanation: This parameter defines the SAP to which the buffer is currently assigned. The SAP_NUMBER portion of the STATION_ID field must identify a valid opened SAP or X'00' (direct station); the STATION_NUMBER portion is ignored.

BUFFER_LEFT

Explanation: This parameter defines the number of buffers in the pool after the command is completed. The protocol driver returns the value when the command has been executed.

FIRST_BUFFER

Explanation: This parameter is the address of the first buffer to be added to the pool. If this value is zero, no buffer is freed, and the command is completed with a CCB_RETCODE of X'00'.

BUFFER.GET

Hex 26

BUFFER.GET

Command Description: This command gets one or more buffers from the SAP's buffer pool or the direct station buffer pool.

Command Specifics: This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver.

For CCB1 and CCB3: The command completion appendage is taken if provided.

For CCB2: Either a semaphore or a READ command can be used for command completion.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Table 3-2. CCB Parameter Table for BUFFER.GET

Offset	Parameter Name	Byte Length	8086 Type	Description
0	STATION_ID	2	DW	SAP/direct station ID; defines the buffer pool
2	BUFFER_LEFT	2	DW	Number of buffers left in the pool *
4	BUFFER_GET	1	DB	Number of buffers to get
5		3	DB	Reserved
8	FIRST_BUFFER	4	DD	Address of first buffer obtained *

* Indicates a returned value.

STATION_ID

Explanation: This parameter defines the SAP buffer pool from which the buffer is to be taken. The SAP_NUMBER portion of the STATION_ID field must identify a valid opened SAP or X'00' (direct station); the STATION_NUMBER portion is ignored.

BUFFER_LEFT

Explanation: This parameter defines the number of buffers in the pool after the command is completed. The protocol driver returns the value when the command is completed.

BUFFER_GET

Explanation: This parameter defines the number of buffers to get from the pool. If there is an inadequate number of buffers in the pool, the command terminates with a CCB_RETCODE of X'19'. If the value is set to 0, the default of 1 is used.

Note: This command could cause a link station to go into a local-busy state if too many buffers are taken.

FIRST_BUFFER

Explanation: This parameter is the address of the first buffer that was obtained. The adapter returns the value when the command is completed. If no buffers are obtained, this field is set to X'00000000'.

DIR.CLOSE.ADAPTER

Hex 04

DIR.CLOSE.ADAPTER

Command Description: This command closes the adapter and terminates all network communications or terminates the *open wrap test*.

Command Specifics: The command forces an immediate shutdown of network communications, and all pending commands will have the control block field CCB_RETCODE set with X'0B'.

For CCB1: If the adapter was opened with a lock code, this command must have the same hexadecimal value in the first 2 bytes of the CCB_PARM_TAB field in order to close the adapter. If the key code is not provided or is not correct, the DIR.CLOSE.ADAPTER command will be rejected with a CCB_RETCODE of X'05' (required parameters not provided).

For CCB2: For this interface, the close works on a per application program basis. The adapter does not close physically.

The CCB_POINTER field will be set with the address of a queue of CCBs that have been terminated by this command.

If the application program has any SAP stations or link stations open, they will be reset (closed) prior to the command completing.

OS/2 is a multitasking operating system. Therefore, multiple application programs can interface with the protocol drivers. This command can be issued by a system administrator using the System Key as defined by the configuration parameters. A physical close resulting from the DIR.CLOSE.ADAPTER command being issued occurs only if this command is issued by the system administrator with the System Key.

Application programs affected by a DIR.CLOSE.ADAPTER command issued with the System Key can be notified of the event. See “System Action

Exceptions for OS/2 EE 1.3" on page B-69 for more information. The System Key should be placed in the CCB_PARAMETER_2 field, if defined.

Whenever a physical close occurs resulting from the use of the System Key, all pending commands for all application programs will be set with the CCB_RETCODE of X'62'. Commands on a per application program basis will be chained to the first command canceled for each application program.

If the application program has previously set bits of the functional address with the DIR.SET.FUNCTIONAL.ADDRESS command, the protocol driver resets the bits. Bits defined by the configuration parameters, however, are not affected.

If the CCB_CMPL_FLAG field is non-zero, an attempt will be made to update a READ command's parameter table with pointers to outstanding CCBs, freed SAP buffers, and outstanding receive frames. If there is a READ command chained to the DIR.CLOSE.ADAPTER command (if CCB_READ_FLAG is non-zero and CCB_POINTER contains the address of a READ CCB), that READ command will be used to return the outstanding data areas of the application program. If there is no READ command chained to the DIR.CLOSE.ADAPTER command, but there is a pending READ command that specifies notifications of command completions, it will be used to return outstanding data areas of the application program. If there is no READ pending that specifies notifications of command completions, free SAP buffer, and outstanding receive frames will not be posted to the user but the pending CCBs will be chained to the CCB_POINTER field of this command.

For CCB3: For this interface, the close works on a per application program basis. The adapter does not close physically.

The CCB_POINTER field will be set with the address of a queue of CCBs that have been terminated by this command.

If the application program has any SAP stations or link stations open, they will be reset (closed) before the command has finished executing.

OS/2 is a multitasking operating system. Therefore, multiple application programs can interface with the protocol drivers. This command can be issued by a system administrator using the System Key as defined by the configuration parameters. A physical close resulting from the DIR.CLOSE.ADAPTER command being issued occurs only if this command is issued by the system administrator with the System Key.

Application programs affected by a DIR.CLOSE.ADAPTER command issued with the System Key can be notified of the event. See "System Action Exceptions for OS/2 EE 1.3" on page B-69 for more information. The System Key should be placed in the CCB_PARAMETER_2 field, if defined.

Whenever a physical close occurs resulting from the use of the System Key, all pending commands for all application programs will be set with the CCB_RETCODE of X'62'. Commands on a per application program basis will be chained to the first command canceled for each application program.

If the application program has previously set bits of the functional address with the DIR.SET.FUNCTIONAL.ADDRESS command, the protocol driver resets the bits. Bits defined by the configuration parameters, however, are not affected.

The canceled commands and freed SAP buffers are returned to the application program device driver entry point for the system action event if the SYSTEM_ACTION_APPNDG field is defined.

An application program using the DD interface must issue the DIR.CLOSE.ADAPTER command when it no longer requires the services provided by the protocol drivers, since the protocol drivers cannot determine when an application program using the DD interface terminates. If the DIR.CLOSE.ADAPTER command is not issued, the internal control blocks used by the protocol drivers to support an application program will not be available for other application programs.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DIR.CLOSE.DIRECT

Hex 34

DIR.CLOSE.DIRECT

Command Description: This command is for CCB2 and CCB3 only. It releases ownership of all direct stations.

Command Specifics: This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver. This command can be issued by a system administrator with the System Key as defined by the configuration parameters. The System Key code can be used and is located in the CCB_PARAMETER_2 field. This key code is used to force the availability of the direct stations. If the System Key is used, all pending commands and buffers from the direct buffer pool can be returned to the application program. See "System Action Exceptions for OS/2 EE 1.3" on page B-69 for more information. This function is not typically used by an application program.

For CCB2: If a READ command is used to post this command, the buffers from the direct buffer pool are returned in the READ command parameter table. If the System Key is used, all pending commands and buffers from the direct buffer pool are returned to the application program affected when a READ command issued by the application program is executed for a system action event.

For CCB3: This command completion can be posted to the completion appendage specified in the CCB of the command. When the completion appendage is defined, all pending commands and the direct buffer pool are returned in the information table when this command is executed. If the System Key is used, all pending commands and the direct buffer pool are returned to the application program if a SYSTEM_ACTION_APPNDG field is defined.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DIR.DEFINE.MIF.ENVIRONMENT

Hex 2B

DIR.DEFINE.MIF.ENVIRONMENT

Command Description: This command is for **CCB1 only**. It defines the environment required for a NETBIOS emulation program to operate with the protocol drivers.

Command Specifics: This command informs the protocol drivers of the interactive routines to be provided by the NETBIOS emulation program. The adapter number in the CCB must be a value from X'00' to X'03', but the environment will be defined for all adapters supported by the Local Area Network Support Program, if they are installed in the workstation. This command does not have any effect on the original PC Network Adapter if one is installed and if the Local Area Network Support Program is not being used. The command can be issued when an adapter is either open or closed. The command is executed entirely in the protocol drivers in the workstation. Therefore, the command completion appendage is not required, as the command is complete upon return. However, the command completion appendage will be taken if provided.

Note: A NETBIOS emulation program must at some time post a completion code to the Network Control Block (NCB) that was presented to it by an application program. If no command completion appendage (NCB_POST@) has been provided, the emulation program should end with an IRET instruction to return to the protocol driver, which will return to the application program.

If an appendage has been defined, the emulation program should end with the following instruction sequence to cause the protocol driver to call the appendage.

Table 3-3. Appendage Instruction Sequence

`CLI`	
`LES BX,NCBADDR`	ES and BX point to the NCB
`STC`	Set carry flag to indicate POST
`RET FAR 2`	Return (around flags on stack)

This special handling of the flags is the indication to the protocol driver that the appendage (NCB_POST@) is to be called. That appendage should end with an IRET instruction.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-4. CCB Parameter Table for DIR.DEFINE.MIF.ENVIRONMENT

Offset	Parameter Name	Byte Length	8086 Type	Description
For CCB1 only:				
0	NCB_INPUT@	4	DD	The address of NCB module
4	NCB_OPEN@	4	DD	The address of the open module
8	NCB_CLOSE@	4	DD	The address of the close module
12	NCB_ENABLE@	4	DD	The address of the interrupt module *

* Indicates a returned value.

NCB_INPUT

Explanation: This field must have a value other than zero. It must contain the address of a module or routine that the protocol driver can call when it has determined that the control block is an NCB rather than a CCB. Registers ES and BX will point to the NCB. Register AL contains flags as defined in the PDT.TRACE CCB entry byte 1. See page 3-79 for more information.

The specified module or routine must end with an IRET instruction back to the application program that issued the NCB. It does not return to the protocol driver.

The module will be entered with the same stack used by the application program that issued the NCB. It is the responsibility of the module to return the stack and registers as they were when the module was entered. Only the return address and flags are on the stack when the module is entered.

NCB_OPEN

Explanation: This field must have a value other than zero. It must contain the address of a module or routine that the protocol driver can call when it has opened an adapter. It does this to inform the NETBIOS emulator that the adapter is open. Registers ES and BX will point to the CCB used to open the adapter. Register CX contains the adapter number.

The specified module or routine must end with a Far Return instruction back to the protocol driver, with register AL set to indicate the return code. If the AL register is set to zero, the NETBIOS emulator indicates a good return. If AL is not zero, the DIR.OPEN command is completed with a return code of X'10'.

NCB_CLOSE

Explanation: This field must have a value other than zero. It must contain the address of a module or routine that the protocol driver can call when it has closed an adapter for any reason. Register CX contains the adapter number.

The specified module or routine must end with a Far Return instruction back to the protocol driver.

NCB_ENABLE

Explanation: The protocol driver returns the address of a routine that is to be called when interrupts are to be enabled.

DIR.INITIALIZE

Hex 20

DIR.INITIALIZE

Command Description: This command initializes the IBM support program protocol driver areas in the workstation, resets all adapter tables and buffers, and directs the adapter to run the bring-up tests. Bring-up tests are for the Token-Ring network only. The adapter's programmable timer is started and set to interrupt the workstation at 100-ms intervals.

For purposes of system integrity, the DIR.INITIALIZE command loops with interrupts enabled (except in the hardware interrupt appendage for CCB1 as explained below) until one of the following occurs:

- The adapter interrupts, indicating completion.
- Time expires (approximately 12 seconds for a Token-Ring Network adapter).

The 12-second period that may elapse is due to the implementation of a 3-second timeout and retry function. The initialization attempt is not aborted until at least four attempts have been made to initialize the adapter and execute diagnostics.

Command Specifics: This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver.

For CCB1: Because the command loops until it is completed, no command completion appendage needs to be defined. The CCB_RETCODE is available upon return.

The command can be issued at any time. It resets the adapter, and any CCBs in process will not be returned. No attempt is made to recover pending CCBs.

This command must be issued before any other command can be issued. This command will also preempt any other command queued for the protocol driver. It will execute immediately.

For CCB2 and CCB3: This command can be issued at any time by a system administrator with the System Key as defined by the configuration parameters. The System Key should be placed in the CCB_PARAMETER_2 field, if defined.

When issued, DIR.INITIALIZE resets the adapter. It also returns pending CCBs and buffer resources to the application program that issued them as system action exception information. That is, the returned command's CCB_RETCODE will be set to X'62'.

Application programs affected by a DIR.INITIALIZE can be notified of the event. See "System Action Exceptions for OS/2 EE 1.3" on page B-69 for more information.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-5 (Page 1 of 2). CCB Parameter Table for DIR.INITIALIZE

Offset	Parameter Name	Byte Length	8086 Type	Description
0	BRING_UPS	2	DW	Bring-up error code. * This parameter is not set on the PC network or on Ethernet.
For CCB1:				
2	SRAM_ADDRESS	2	DW	Segment value of adapter shared RAM. * This parameter is not recognized on the PC Network or on Ethernet.
4		4	--	Reserved.
8	ADPTR_CHK_EXIT	4	DD	I/O appendage exit, adapter check.
12	NETW_STATUS_EXIT	4	DD	I/O appendage exit, network status change.
16	PC_ERROR_EXIT	4	DD	I/O appendage exit, error in the workstation. This parameter is not set on the PC Network or on Ethernet.
For CCB2 and CCB3:				
2	SRAM_ADDRESS	2	DW	Configured address of shared RAM. * This parameter is not recognized on the PC Network or on Ethernet.
4		1	DB	Reserved. *
5		15	DB	Reserved.
For CCB2:				
20	VIRTUAL_SRAM_ADDRESS	4	DD	Virtual address of shared RAM.* For the PC Network or for Ethernet, this is the address of RAM containing link stations, SAPs, and so on.
24	VIRTUAL_MMIO_ADDRESS	4	DD	Virtual address of MMIO. * This is not set on the PC Network or on Ethernet.

* Indicates a returned value.

Table 3-5 (Page 2 of 2). CCB Parameter Table for DIR.INITIALIZE

Offset	Parameter Name	Byte Length	8086 Type	Description
28	DATA_SEG_ADDRESS	4	DD	Virtual address of device driver's data segment. *
For CCB3:				
20	VIRTUAL_SRAM_ADDRESS	4	DD	Virtual address with GDT selector of shared RAM. * For the PC Network or Ethernet, this is the address of RAM containing link stations, SAPs, and other data structures.
24	VIRTUAL_MMIO_ADDRESS	4	DD	Virtual address with GDT selector of MMIO. * This is not set on the PC Network or on Ethernet.
28	DATA_SEG_ADDRESS	4	DD	Virtual address with GDT selector of device driver's data segment. *

* Indicates a returned value.

BRING_UPS

Explanation: Indicates the results of the adapter bring-up testing for IBM Token-Ring Network Adapters. A value is entered only if the adapter fails bring-up. If the adapter passes bring-up, the value of this parameter is X'0000'. If the adapter fails bring-up, the command terminates with a CCB_RETCODE of X'07'. See "Bring-Up Errors for All CCBs" on page B-57 for a list of bring-up error codes and descriptions.

This parameter is not set on the PC Network or on Ethernet.

SRAM_ADDRESS

Explanation: This defines the workstation memory segment where the adapter shared RAM is to be addressed. This field is ignored on the PC Network or Ethernet.

For CCB1:

- For Token-Ring Network adapters with ISA Bus

  If the application program specifies a non-zero number for shared RAM when loading the protocol drivers, that value is used as the segment address for the shared RAM.

  If zero is coded here, the defaults (X'D800' for adapter 0, X'D400' for adapter 1) are used.

The value must be on the same address boundary as the shared RAM size mapped into the workstation (for example, on an 8 KB boundary for 8 KB of shared RAM, on a 16 KB boundary for 16 KB of shared RAM, and so on).

- For Token-Ring Network adapters with Micro Channel

  The input value is ignored (the shared RAM segment address is set using the Reference Diskette shipped with the workstation). A non-zero value coded here does not update the shared RAM address, and no error code is returned to the application program.

The value returned will be the segment address used for the shared RAM on both Token-Ring Network adapters with ISA Bus and Token-Ring Network adapters with Micro Channel.

For CCB2 and CCB3:

- For Token-Ring Network adapters with ISA Bus

  The shared RAM segment address is defined using the configuration parameters. See Appendix E, "Operating System/2 Extended Edition Information," for additional information.

- For Token-Ring Network adapters with Micro Channel

  The input value is ignored. The shared RAM segment address is set using the Reference Diskette shipped with the workstation.

The value returned will be the segment address used for the shared RAM on both Token-Ring Network adapters with ISA Bus and Token-Ring Network adapters with Micro Channel. This value cannot be used directly by the system administrator while the workstation is running in protect mode.

ADPTR_CHK_EXIT

Explanation: The address of a user-provided appendage routine that is taken when an adapter error condition is detected. If the value is zero, no exit is defined. This exit can also be overridden by the DIR.OPEN.ADAPTER, DIR.SET.USER.APPENDAGE, and DIR.MODIFY.OPEN.PARMS commands. See "Exception Indications" on page B-46 for adapter check error codes.

NETW_STATUS_EXIT

Explanation: The address of a user-provided appendage routine that is taken when the network status changes. This exit may also be overridden by the DIR.OPEN.ADAPTER, DIR.SET.USER.APPENDAGE, and DIR.MODIFY.OPEN.PARMS commands. See "Exception Indications" on page B-46 for network status codes.

PC_ERROR_EXIT

Explanation: The address of a user-provided appendage routine that is taken when the protocol driver detects an error condition in the workstation. This is not used on the PC Network or on Ethernet. This exit can also be overridden by the DIR.OPEN.ADAPTER, DIR.SET.USER.APPENDAGE, and DIR.MODIFY.OPEN.PARMS commands. See "Exception Indications" on page B-46 for workstation errors.

VIRTUAL_SRAM_ADDRESS

Explanation:

For CCB2: Virtual address of adapter shared RAM.

This parameter defines the address (selector:offset) that can be used to access the Token-Ring Network adapter's shared RAM.

It also can define the address (selector:offset) that can be used to access the workstation RAM that has been allocated to the PC Network or Ethernet adapters.

An entry for this memory segment is made into the logical descriptor table (LDT) of the OS/2 EE process that issues this command. This command is only valid for the process that issued it.

For CCB3: Virtual address with global descriptor table (GDT) selector of adapter shared RAM.

This parameter defines the address with GDT selector (selector:offset) that can be used to access the Token-Ring Network adapter's shared RAM.

It can also define the address with GDT selector (selector:offset) that can be used to access the workstation RAM that has been allocated to the PC Network or Ethernet adapters.

MMIO_ADDRESS

Explanation:

For CCB2: Virtual address of adapter memory mapped input output (MMIO).

This parameter defines the address (selector:offset) that can be used to access the adapter MMIO.

This parameter is not returned when you are using PC Network or Ethernet adapters.

An entry for this memory segment is made into the LDT of the OS/2 EE process that issues this command. This command is only valid for the process that issued it.

For CCB3: Virtual address with GDT selector of adapter MMIO.

This parameter defines the address with GDT selector (selector:offset) that can be used to access the adapter MMIO.

This parameter is not returned when you are using PC Network or Ethernet adapters.

DATA_SEG_ADDRESS

Explanation:

For CCB2: Virtual address of the data segment in which the protocol drivers are located.

This parameter defines the address (selector:offset) that can be used to access the data segment in which the protocol drivers are located.

This memory segment has been mapped in the GDT of the OS/2 EE process that issues this command.

An entry for this memory segment is made into the LDT of the OS/2 EE process that issues this command. This command is only valid for the process that issued it.

For CCB3: Virtual address with GDT selector of the data segment in which the protocol drivers are located.

This parameter defines the address with GDT selector (selector:offset) that can be used to access data segment in which the protocol drivers are located.

DIR.INTERRUPT

Hex 00

DIR.INTERRUPT

Command Description: This command only forces an adapter interrupt. It performs no operation.

For CCB1: The adapter must have been initialized, but does not have to be opened for this command to be accepted.

For CCB2 and CCB3: The adapter must be opened before this command is issued.

Command Specifics: No parameter table is required.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DIR.MODIFY.OPEN.PARMS

Hex 01

DIR.MODIFY.OPEN.PARMS

Command Description: This command is for **CCB1 only**. It is used to modify certain values set by the DIR.OPEN.ADAPTER command.

Command Specifics: This command is rejected if either a BUFFER.FREE command has been issued, or a RECEIVE command is active at the direct interface, or if a direct interface buffer pool has been defined.

After this command has been issued successfully, it cannot be issued again until a DIR.RESTORE.OPEN.PARMS command has been issued and successfully completed.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Table 3-6 (Page 1 of 2). CCB Parameter Table for DIR.MODIFY.OPEN.PARMS

Offset	Parameter Name	Byte Length	8086 Type	Description
For CCB1 only:				
0	DIR_BUF_SIZE	2	DW	The size of the new direct interface SAP buffers

Table 3-6 (Page 2 of 2). CCB Parameter Table for DIR.MODIFY.OPEN.PARMS

Offset	Parameter Name	Byte Length	8086 Type	Description
2	DIR_POOL_BLOCKS	2	DW	The length in 16-byte blocks, of buffers in the new direct interface buffer pool
4	DIR_POOL_ADDRESS	4	DD	The starting segment of the new direct interface buffer pool
8	ADPT_CHK_EXIT	4	DD	New I/O appendage exit, adapter check
12	NETW_STATUS_EXIT	4	DD	New I/O appendage exit, network status
16	PC_ERROR_EXIT	4	DD	New I/O appendage exit, error in workstation
20	OPEN_OPTIONS	2	DW	New options (wrap option is ignored)

See the parameter descriptions for the DIR.OPEN.ADAPTER command.

DIR.OPEN.ADAPTER

Hex 03

DIR.OPEN.ADAPTER

Command Description:

For CCB1: This command makes the adapter ready for normal network communication. All buffers and tables will be reinitialized.

For CCB2 and CCB3: This command makes the protocol drivers ready for normal network communications on all network adapters for a given application program. In addition, this command provides the application program with DLC and direct interface information necessary for network communications.

The DIR.OPEN.ADAPTER command must be issued and executed successfully before any network communication can start. The only commands that can be issued before opening the adapter are those commands that use the System Key as defined by the configuration parameters or the DIR.STATUS command.

Since OS/2 EE is a multitasking operating system, where multiple application programs can interface with the protocol drivers, each DIR.OPEN.ADAPTER command results in a logical open. This command is executed entirely in the workstation when a logical open is performed. The return code is available to the application program upon return from the protocol drivers. A physical open occurs only if the adapter has been physically closed due to a system action or an unrecoverable error. When a physical open does occur, all buffers and tables in the adapter are reinitialized.

Command Specifics:

For CCB1: This command cannot be issued unless the adapter is in a closed state, such as that following a DIR.INITIALIZE or a DIR.CLOSE.ADAPTER command. The only commands that should be issued before the DIR.OPEN.ADAPTER command are DIR.INITIALIZE, DIR.INTERRUPT, and DIR.STATUS.

This command also makes the adapter ready for an adapter wrap test on Token-Ring Network adapters.

If the first DIR.OPEN.ADAPTER is in process when the second DIR.OPEN.ADAPTER is issued, the value for NODE_ADDRESS may be invalid.

If the OPEN_OPTIONS bit 9 (Pass Parm Table bit) is on when the DIR.OPEN.ADAPTER command is issued, all the ADAPTER_PARMS and DLC_PARMS values in the parameter table that were set to use default values are updated with the default value actually used. If the adapter is open when this command is issued and the open option bit 9 is on, the command will be executed with the X'03' return code and also have its parameter tables updated with the currently active open parameter values. The actual values for the ADAPTER_PARMS and DLC_PARMS fields are returned. The actual values for the OPEN_LOCK and PRODUCT_ID_ADDRESS fields are not returned; a value of zero is stored in these fields. If the DLC interface is not open, zeros are returned in the DLC_PARMS field.

The DIR.OPEN.ADAPTER parameter table has four pointers to function-oriented tables. These tables contain open parameters for the adapter itself, the direct interface, the DLC interface, and NETBIOS. See Chapter 4, "NETBIOS," for using NETBIOS.

For CCB2: This command cannot be issued unless the adapter is in a logically closed state for the application program issuing the command. A logically closed state exists after one of the following occurs: adapter initialization, detection of a fatal error, or the execution of a DIR.CLOSE.ADAPTER command.

- After a DIR.OPEN.ADAPTER command is issued, no commands (other than commands using the System Key or the DIR.STATUS command) are allowed from the application program until the DIR.OPEN.ADAPTER command is executed.
- Posting of the DIR.OPEN.ADAPTER command can only be done using a system semaphore or polling of the return code.
- The adapter must be logically closed at the time the DIR.OPEN.ADAPTER command is issued.
- The CCB_SEMAPHORE is the posting semaphore.
- The CCB_APPL_ID is the application program ID.

  A unique ID is returned for each application program. The protocol drivers use this ID to reference application program resources. The application program ID returned must be used for all subsequent commands that the application program issues.
- The CCB_APPL_KEY is a key code.

The key code is used for the application program's resource security. The key code logically locks the resources of application programs. No resource of an application program can be used, discarded, or modified without this key code. If you do not want to have resources controlled for an application program, specify X'0000'. If a key code is specified, all following commands must also specify a key that matches this CCB_APPL_KEY code.

The DIR.OPEN.ADAPTER command's parameter table contains two pointers to function-oriented tables. The fields of these two tables are returned by the protocol driver and contain open parameters for the adapter itself and for the DLC interface.

The OS/2 EE process that actually issues the DIR.OPEN.ADAPTER command is considered the "application program." When this process is terminated, all application program control blocks used to maintain the logical relationship between the application program and the protocol driver are cleaned up and made available for other application programs.

After the DIR.OPEN.ADAPTER command has been issued, the application program should ensure that the network is in working order by issuing the DIR.STATUS command to retrieve the current network status.

For CCB3: This command cannot be issued unless the adapter is in a logically closed state for the application program issuing the command. A logically closed state exists after one of the following occurs: adapter initialization, detection of a fatal error, or the execution of a DIR.CLOSE.ADAPTER command.

- After a DIR.OPEN.ADAPTER command is issued, no commands (other than commands using the System Key) are allowed from the application program until the DIR.OPEN.ADAPTER command is executed.
- Posting of the DIR.OPEN.ADAPTER command can only be done using an appendage or polling of the return code.
- The adapter must be logically closed at the time the DIR.OPEN.ADAPTER command is issued.
- The CCB_APPL_ID is the application program ID.

  A unique ID is returned for each application program. The protocol driver uses this ID to reference application program resources. The application program ID returned must be used for all subsequent commands that the application program issues.
- The CCB_APPL_KEY is a key code.

  The key code is used for the application program's resource security. The key code logically locks the resources of application programs. No resource of an application program can be used, discarded, or modified without this key code. If you do not want to have resources controlled for an application program, specify X'0000'. If a key code is specified, all subsequent commands must also specify a key that matches this CCB_APPL_KEY code.

The DIR.OPEN.ADAPTER command's parameter table contains two pointers to function-oriented tables. The fields of these two tables are returned by the protocol driver and contain open parameters for the adapter itself and for the DLC interface.

After the DIR.OPEN.ADAPTER command has been issued, the application program should ensure that the network is in working order by issuing the DIR.STATUS command to retrieve the current network status.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-7. CCB Parameter Table for DIR.OPEN.ADAPTER

Offset	Parameter Name	Byte Length	8086 Type	Description
For CCB1:				
0	ADAPTER_PARMS	4	DD	Pointer to the adapter parameters.
4	DIRECT_PARMS	4	DD	Pointer to the direct interface parameters.
8	DLC_PARMS	4	DD	Pointer to the DLC parameters.
12	NCB_PARMS	4	DD	Pointer to the NETBIOS parameters.
For CCB2 and CCB3:				
0	ADAPTER_PARMS_OFFSET	2	DW	Offset to the adapter parameters.
2		2	DW	Reserved for the application program. Can be used for the segment or selector.
4		4	DD	Reserved.
8	DLC_PARMS_OFFSET	2	DW	Offset to the DLC parameters.
10		2	DW	Reserved for the application program. Can be used for the segment or selector.
12		4	DD	Reserved.

ADAPTER_PARMS and ADAPTER_PARMS_OFFSET

Explanation: These parameters point to a parameter table (see Table 3-8 on page 3-21).

For CCB1: This parameter is required and points to a parameter table. There must be a non-zero value provided for this parameter.

For CCB2 and CCB3: This parameter is an offset that points to a parameter table. This pointer should be provided so that global adapter parameters can be returned to the application program.

Most parameters are returned values that have been defined by configuration parameters.

DIRECT_PARMS

Explanation: This parameter is used by the direct station only and points to a parameter table (see Table 3-11 on page 3-28). There must be a non-zero value provided for this parameter.

DLC_PARMS and DLC_PARMS_OFFSET

Explanation: These parameters point to a parameter table (see Table 3-12 on page 3-31).

For CCB1: This parameter is required and points to a parameter table. It must be defined (not zero) if any interface other than the direct interface is to be used. If this field value is zero, the DLC interface and NETBIOS are not operational.

For CCB2 and CCB3: This parameter is an offset that points to a parameter table. This pointer should be provided so that global DLC parameters can be returned to the application program.

Most parameters are returned values that have been defined by configuration parameters.

NCB_PARMS

Explanation: This field is a pointer to a parameter table. If this field value is zero when NETBIOS is used, all default values are used. See page 4-8 for information about the NETBIOS operational parameters.

Parameters:

Table 3-8 (Page 1 of 2). Adapter Parms Open Parameters

Offset	Parameter Name	Byte Length	8086 Type	Description
0	OPEN_ERROR_CODE	2	DW	Open adapter errors detected. Set by the adapter. **
2	OPEN_OPTIONS	2	DW	Various options. *
4	NODE_ADDRESS	6	DB	This adapter's address. **
10	GROUP_ADDRESS	4	DB	Set group address. *
14	FUNCTIONAL_ADDR	4	DB	Set functional address. *
18	NUMBER_RCV_BUFFERS	2	DW	Number of receive buffers. *
20	RCV_BUFFER_LEN	2	DW	Length of the receive buffers. *
22	DHB_BUFFER_LENGTH	2	DW	Length of the transmit data hold buffers. *

* Indicates a returned value for CCB2 and CCB3.

** Indicates a returned value for all CCBs.

Table 3-8 (Page 2 of 2). Adapter Parms Open Parameters

Offset	Parameter Name	Byte Length	8086 Type	Description
24	DATA_HOLD_BUFFERS	1	DB	Number of transmit data hold buffers. * This parameter is not recognized on the PC Network or on Ethernet. Is used only by the Token-Ring Network.
For CCB1:				
25		1	DB	Reserved.
26	OPEN_LOCK	2	DW	A protection code to control closing the adapter.
28	PRODUCT_ID_ADDRESS	4	DD	Address of the 18-byte product ID.
For CCB2 and CCB3:				
25		3	DB	Reserved.
28	PRODUCT_ID_OFFSET	2	DW	Offset of 18-byte product ID.
30		2	DW	Reserved for the application program. Can be used for the segment or selector.
32	BRING_UPS	2	DW	Bring up errors. *
34	INIT_WARNINGS	2	DW	Initialization warnings. *
For CCB2:				
36	SEMAPHORE_COUNT	2	DW	Count of system semaphore handles.
38	SYS_SEMAPHORE_TABLE	4	DD	Address to system semaphore table.
For CCB3:				
36		2	DW	Reserved for the application program.
38		4	DD	Reserved for the application program.
42	DDNAME	8	DB	Device driver name.

* Indicates a returned value for CCB2 and CCB3.

** Indicates a returned value for all CCBs.

OPEN_ERROR_CODE

Explanation: Indicates the results of the open adapter testing. If the value is not X'0000', the command terminates with a CCB_RETCODE of X'07'. See "Token-Ring Network Adapter Open Errors for All CCBs" in Appendix B, "Return Codes," for open error codes.

OPEN_OPTIONS

Explanation: Various options, each defined by a bit. A bit on (1) indicates that the option is to be taken. Bit 15 is the high-order (leftmost) bit. Only bit 9 is recognized on the PC Network or on Ethernet.

For CCB2 and CCB3: These are returned values.

Table 3-9 (Page 1 of 2). IBM Token-Ring Network Adapter OPEN_OPTIONS

Bit	Name	Description
15	Wrap interface	The adapter will not attach itself to the network. Instead, it causes all user transmit data to be wrapped as received data.
14	Disable Hard Error	Prevents network status changes involving "Hard Error" and "Transmit Beacon" bits from causing interrupts.
13	Disable Soft Errors	Prevents network status changes involving the "Soft Error" bit from causing interrupts.
12	Pass Adapter MAC Frames	Passes, as direct interface data to the workstation, all adapter class MAC frames that are received but not supported by the adapter. If this bit is set to off, these frames are ignored.
11	Pass Attention MAC Frames	Passes, as direct interface data to the workstation, all attention MAC frames that are not the same as the last received attention MAC frame. If this option is set to off, these frames are not passed to the application program.
10	Reserved	Should be zero, but this bit is not checked by the protocol driver.
9	**For CCB1:** Pass Parm Table **For CCB2 and CCB3:** Reserved	If the adapter is already open, the options used when opening the adapter are returned to the user parameter table. Should be zero, but this bit is not checked by the protocol driver.
8	Contender	When the contender bit is set to on, the adapter participates in monitor contention (claim token) if the opportunity occurs. When the contender bit is set to off, and the need is detected by another adapter, this adapter does not participate. If the need for determining a new active monitor is detected by this adapter, monitor contention (claim token) processing will be initiated by this adapter whether the contender bit is set to on or off.
7	Pass Beacon MAC frames	Passes, as direct interface data to the workstation, the first beacon MAC frame and all subsequent beacon MAC frames that have a change in the source address or the beacon type.
6	Reserved	Should be zero, but this bit is not checked by the protocol driver.
5	Remote Program Load	This bit is only implemented in 16/4 adapters. It prevents the adapter from becoming a monitor during the open process. If this bit is set to on, the adapter fails the open process if there is no other adapter on the ring when it attempts to insert on the ring.

Table 3-9 (Page 2 of 2). IBM Token-Ring Network Adapter OPEN_OPTIONS

Bit	Name	Description
4	Token Release	This bit is only available for 16/4 adapters when operating at 16 Mbps. If the bit is not set, 16 Mbps adapters use early token release (the default). Setting this bit on selects no early token release for an adapter operating at 16 Mbps. **For CCB1:** If Token Release is selected by using the command line parameter, this bit will be set on when the DIR.OPEN.ADAPTER command is executed.
0-3	Reserved	Should be zero, but these bits are not checked by the protocol driver.

Refer to the *IBM Token-Ring Network Architecture Reference* for more information about network operation.

NODE_ADDRESS

Explanation: The 6-byte specific node address of this station on the network. The value must not be all ones. The two high-order (leftmost) bits must be B'01'. For other restrictions and details about addresses, refer to the *IBM Token-Ring Network Architecture Reference.*

If the NODE_ADDRESS parameter was provided when the protocol drivers were loaded, that address is used rather than the address provided in this parameter field. The address used is returned in this field by the protocol driver for return to the application program.

For CCB1: If the value is zero, the address encoded on the adapter is the node address by default, and that value is placed in this field by the protocol driver for return to the application program.

If the OPEN_LOCK field is coded as zero, then:

- If a locally administered address was specified when the protocol drivers were loaded, the locally administered address is used regardless of the contents of this field.
- If a zero was specified when the protocol drivers were loaded, the encoded address is used regardless of the contents of this field.

For CCB2 and CCB3: See Appendix E, "Operating System/2 Extended Edition Information."

GROUP_ADDRESS

Explanation: Sets the group address for which the adapter will receive messages. If the value is zero, no group address is set.

FUNCTIONAL_ADDR

Explanation: Sets the functional address the adapter will receive messages for. The most significant bit and the least significant bit of this field are ignored by the adapter. If the value is zero, no functional address is set.

For CCB1: If NETBIOS is made operational, it will reissue a DIR.SET.FUNCTIONAL.ADDRESS command using all bits set in the current functional address and adding X'00000080' to the bits being used.

For Ethernet adapters, if a functional address is specified in the DIR.OPEN.ADAPTER command, the same actions will be taken as described for the DIR.SET.FUNCTIONAL.ADDRESS command. If the number of bits to be set for the functional address exceeds the number of available multicast addresses, the DIR.OPEN.ADAPTER command will not be executed and a return code of X'1E' will be posted in the CCB.

NUMBER_RCV_BUFFERS

Explanation: The number of receive buffers needed for the adapter to open. The adapter configures all remaining RAM as receive buffers after other memory requirements have been met. If the number available is less than the number requested, the DIR.OPEN.ADAPTER command fails. If the number available is greater than the number requested, no action occurs. If this value is less than 2, the default of 8 is used.

If you are using DXME0MOD.SYS, you may find you have inadequate work space. The NDIS MAC driver for an NDIS adapter specifies the largest frame size the adapter can process. When the DIR.OPEN.ADAPTER command is issued, the receive buffer length and the number of receive buffers are checked to make sure one maximum size frame can be received in them. (Receive frames can occupy more than one receive buffer.) If the maximum size frame does not fit, the DIR.OPEN.ADAPTER command returns `X'30' Inadequate receive buffers for adapter to open.`

To correct this problem, you need to specify a larger work space for the adapter. To do this, increase the value of the work space parameter on the command line of the protocol driver DXME0MOD.SYS.

To increase the work space for a PC Network Adapter, alter the work space parameter on the DXMGnMOD.SYS protocol driver.[1] Refer to the *Local Area Network Support Program User's Guide* for more information.

RCV_BUFFER_LENGTH

Explanation: The length of each of the receive buffers. The value must be a multiple of 8 with 96 as the minimum, and 2048 as the maximum.

If the value is zero, the default of 112 is used. When DXME0MOD.SYS is used, each buffer holds 14 fewer bytes of data than the specified length; otherwise, each buffer holds 8 fewer bytes of data than the specified length. Therefore, a buffer defined as 112 bytes can hold only 104 bytes of data. If a frame received from the network is longer than one buffer, receive buffers will be chained.

[1] The letter *n* stands for the version of DXMGnMOD.SYS, 0, 1, or 2.

For CCB1: For Ethernet adapters, the minimum and default receive buffer length is 208. If the user specifies a buffer length between 96 and 200, length of the receive buffer is automatically incremented to 208 without flagging an error or notifying the application.

DHB_BUFFER_LENGTH

Explanation: The length of each of the transmit data hold buffers.[2]

For the following Token-Ring Network adapters, the maximum DHB length is 2048:

- Token-Ring Network PC Adapters and PC Adapter IIs
- Token-Ring Network Adapter/As.

For all new Token-Ring Network adapters operating at 4 Mbps that are supported by IBM support programs, the maximum DHB length is 4464, and for all Token-Ring Network adapters operating at 16 Mbps that are supported by IBM support programs, the maximum DHB length is 17960.

Note: If a length greater than 2048 is used, it is important to make sure that all adapters receiving these frames can also handle the larger frame length.

For CCB1: For Ethernet adapters, the maximum DHB buffer length is 1496.

If the value is zero, the default of 600 is used. Each buffer holds 6 fewer bytes of data than the specified length. Therefore, a buffer defined as 600 bytes can hold only 594 bytes.

DATA_HOLD_BUFFERS

Explanation: Defines the number of transmit data hold buffers in the adapter in which data from the workstation is stored. This parameter is not recognized on the PC Network or on Ethernet.

The adapter accepts any value between zero and 255, but the integrity of adapter operation cannot be guaranteed if the value is greater than 2. Requesting two buffers may improve adapter performance by allowing a frame to be moved into the second buffer while the adapter is transmitting from the first. However, this reduces the storage available for receive buffers. Transmit buffers are not chained. If the value is zero, the default of 1 is used.

OPEN_LOCK

Explanation: A code provided to the application program to lock the adapter open. Only a DIR.CLOSE.ADAPTER command that has a matching key code can close the adapter. When using this feature, you must make sure that the adapter is closed when the application program is finished, or all application programs using the adapter must follow consistent rules about opening and closing the adapter. This field permits one application program to supervise adapter closing when more than one application program or operation has access.

It is recommended that the application program code this field as zero.

This field is not returned by coding the OPEN_OPTION bit 9.

[2] This value must be a multiple of 8.

PRODUCT_ID_ADDRESS

Explanation: The address in workstation memory where an 18-byte product ID is located.

The product ID provides indications about the workstation and programs used. The field can point to a location containing all zeros, or point to a product ID field prepared as shown in Table 3-10.

This field is not returned by coding the OPEN_OPTION bit 9.

PRODUCT_ID_OFFSET

Explanation: The offset in workstation memory where an 18-byte product ID is located.

The product ID provides indications about the workstation and programs used. The field must point to a location containing all zeros or point to a product ID field prepared as in Table 3-10.

Table 3-10. Product ID Field

Offset	Description
Byte 0	X'01' indicates workstation.
Byte 1	X'10'.
Bytes 2-5	The machine type from the serial number tag at rear of workstation (the last 4 digits). Enter in EBCDIC. For example, for serial number=61382, code the field as (F0 F0 F0 F6 F1 F3 F8 F2).
Bytes 6-17	Zero-filled.

BRING_UPS

Explanation: Bring-up error codes. See "Bring-Up Errors for All CCBs" on page B-57 for a list of all valid codes. This parameter indicates the results of Token-Ring Network Adapter bring-up testing.

INIT_WARNINGS

Explanation: Processing during initialization may detect errors that are not fatal to the initialization of the protocol drivers. These errors will result in the use of default values to configure the protocol drivers. This field is used to notify the user that such errors have occurred and were not fixed by the system administrator during system initialization. Below are the values contained in this field.

X'00'	No errors detected during configuration parameters processing.
X'01'	No configuration parameters found for the adapter.
X'02'	Errors found during configuration parameters processing; default values used.

SEMAPHORE_COUNT

Explanation: Number of system semaphore handles referenced by the SYS_SEMAPHORE_TABLE parameter. If this value is greater than 12, the command terminates with CCB_RETCODE set to X'06'.

SYS_SEMAPHORE_TABLE

Explanation: Address of system semaphore handle table. System semaphore handles that will be used for posting asynchronous events can be registered with the protocol driver prior to using the semaphore handles. Registering system semaphore handles prior to using them allows the protocol driver to obtain in advance new handles that it can use. Obtaining the handles means that the protocol driver uses less overhead when it processes a command that is issued with one of the registered system semaphore handles in the CCB_SEMAPHORE field.

A table of system semaphore handles can be passed using this parameter. The table can contain up to twelve 4-byte system semaphore handles that are placed in a control block pointed to by this parameter. The first handle will be located in bytes 0-3 with other handles following (for example, bytes 4-7, bytes 8-11). Values used for system semaphore handles are the values returned from OS/2 when the system semaphore is created or opened.

Only the OS/2 process that issues the DIR.OPEN.ADAPTER command and provides system semaphore handles can use these handles. All other processes associated with the application program must provide a handle returned from OS/2 for the given process when the system semaphore is created or opened.

Note: The semaphores must not be created as exclusive.

DDNAME

Explanation: This is the name of the application program's device driver that contains the user exits to be called when events are completed. The protocol driver issues an OS/2 Device Help ATTACHDD function call to obtain the application program device driver's intercommunication entry point. The application program's device driver must have enabled interdevice driver communications by setting the DEVICE_ATTRIBUTE field of the device driver header (this allows support for the protocol driver to use the ATTACHDD function and obtain the application program's device driver entry point). ATTACHDD is an OS/2 Device Help function. When the application program's device driver issues the DIR.OPEN.ADAPTER command, this parameter specifies the application program's device driver name.

For CCB1: Direct Parms Open Parameters

Table 3-11 (Page 1 of 2). Direct Parms Open Parameters for CCB1

Offset	Parameter Name	Byte Length	8086 Type	Description
0	DIR_BUF_SIZE	2	DW	Size of buffers in "direct" buffer pool.
2	DIR_POOL_BLOCKS	2	DW	The length, in 16-byte blocks, of buffers in direct buffer pool.
4	DIR_POOL_ADDRESS	4	DD	Starting segment of direct buffer pool.
8	ADPT_CHK_EXIT	4	DD	I/O appendage exit, adapter check.

* Indicates a returned value.

Table 3-11 (Page 2 of 2). Direct Parms Open Parameters for CCB1

Offset	Parameter Name	Byte Length	8086 Type	Description
12	NETW_STATUS_EXIT	4	DD	I/O appendage exit, network status change.
16	PC_ERROR_EXIT	4	DD	I/O appendage exit, error in the workstation.
20	WORK_ADDR	4	DD	Adapter work area, segment value. This parameter is not recognized on the PC Network or on Ethernet.
24	WORK_LEN_REQ	2	DW	Adapter work area length, requested. This parameter is not recognized on the PC Network or on Ethernet.
26	WORK_LEN_ACT	2	DW	Adapter work area length, actual. * This parameter is not recognized on the PC Network or on Ethernet.

* Indicates a returned value.

DIR_BUF_SIZE

Explanation: The size of the buffers in the direct station buffer pool including adapter overhead space. The minimum length is 80 bytes, and the length must be a multiple of 16. If the value is zero, the default of 160 is used.

DIR_POOL_BLOCKS

Explanation: The number of 16-byte blocks assigned as the direct station buffer pool. If the value is zero, the default of 256 (4096 bytes) is used. If DIR_POOL_ADDRESS is zero, this parameter is ignored. This field is set to the size of the application program's buffer area divided by 16.

DIR_POOL_ADDRESS

Explanation: The segment value of the address in workstation memory where the protocol driver is to build the direct station buffer pool. See "Buffer Pools" on page 2-40. If the value is zero, the application program has the responsibility of managing its own buffer pool. If this is the case, the DIR_BUF_SIZE parameter must indicate the size of each buffer.

ADPT_CHK_EXIT

Explanation: The address of a user-provided appendage routine that is taken when the adapter detects an error in the adapter. If the value is zero, the default is as defined by the DIR.INITIALIZE command.

NETW_STATUS_EXIT

Explanation: The address of a user-provided appendage routine that is taken when the network status changes. If the value is zero, the default is as defined by the DIR.INITIALIZE command.

PC_ERROR_EXIT

Explanation: The address of a user-provided appendage routine that is taken when the protocol driver detects an error in the workstation. If the value is zero, the default is as defined by the DIR.INITIALIZE command.

WORK_ADDR

Explanation: A segment value in workstation memory where a work area has been allocated for the protocol drivers. This value is ignored if the WORK_LEN_REQ field is zero. This parameter is not recognized on the PC Network or on Ethernet.

WORK_LEN_REQ

Explanation: The length of the work area in workstation memory, defined by the parameter WORK_ADDR. The length must be enough to contain all protocol driver work areas defined by the DIR.OPEN.ADAPTER parameters DLC_MAX_SAP and DLC_MAX_STATIONS.

The following formula enables you to calculate the workstation memory space needed when you are using the LAN Support Program Versions 1.22 through 1.34. To use this formula, add the subtotals for SAPs and stations to the 74 bytes for direct interface stations:

```
        The number of SAPs (DLC_MAX_SAP) x 38 =
The number of stations (DLC_MAX_STATIONS) x 18 =
                                            +   74
                                              ______
           Total workstation RAM needed (bytes) =
```

If this value is 0, the internal work area is used and the WORK_ADDR field will be ignored. This parameter is not recognized on the PC Network or on Ethernet.

WORK_LEN_ACT

Explanation: The returned value for the length of the work area in workstation memory required by the protocol drivers. If the actual length is greater than the work area (internal or specified), the command terminates with the CCB_RETCODE field containing X'12' (available work area exceeded). This parameter is not recognized on the PC Network or on Ethernet. It is retained for Token-Ring Network compatibility.

For all CCBs: DLC Parms Open Parameters

Table 3-12. DLC Parms Open Parameters for All CCBs

Offset	Parameter Name	Byte Length	8086 Type	Description
0	DLC_MAX_SAP	1	DB	Maximum number of SAPs *
1	DLC_MAX_STATIONS	1	DB	Maximum number of link stations *
2	DLC_MAX_GSAP	1	DB	Maximum number of group SAPs *
3	DLC_MAX_GMEM	1	DB	Maximum members per group SAP *
4	DLC_T1_TICK_ONE	1	DB	DLC timer T1 interval, group 1 *
5	DLC_T2_TICK_ONE	1	DB	DLC timer T2 interval, group 1 *
6	DLC_Ti_TICK_ONE	1	DB	DLC timer Ti interval, group 1 *
7	DLC_T1_TICK_TWO	1	DB	DLC timer T1 interval, group 2 *
8	DLC_T2_TICK_TWO	1	DB	DLC timer T2 interval, group 2 *
9	DLC_Ti_TICK_TWO	1	DB	DLC timer Ti interval, group 2 *

* Indicates a returned value for OS/2 (CCB2 and CCB3).

DLC_MAX_SAP

Explanation: The maximum number of SAPs that can be opened at one time. The maximum value that is allowable may be limited by the amount of available workstation RAM work area allocated for the protocol drivers, or by the amount of adapter shared RAM. If NETBIOS is opened, one of the SAPs is required for the use of the NETBIOS protocol driver and therefore is unavailable for application program use. The maximum value of this parameter is 127 (126 for NETBIOS). If the value is zero, the default of 2 is used.

DLC_MAX_STATIONS

Explanation: The maximum number of link stations that can be opened at one time. The maximum value that is allowable may be limited by the amount of available workstation RAM work area allocated for the protocol drivers or by the amount of adapter shared RAM. If NETBIOS is opened, some of the link stations will be used by the SAP assigned for NETBIOS protocol driver use. If the value is zero, the default of 6 is used. The maximum value of this parameter is 255 for the Token-Ring Network, the PC Network, or Ethernet.

DLC_MAX_GSAP

Explanation: The maximum number of group SAPs that can be opened at one time. If the value is zero, no group SAPs are allowed. The maximum value of this parameter is 126 for the Token-Ring Network (125 for the PC Network and Ethernet).

DLC_MAX_GMEM

Explanation: The maximum number of SAPs that can be assigned to any given group. The maximum value of this parameter is 127 for the Token-Ring Network (126 for the PC Network and Ethernet).

DLC_T1_TICK_ONE

Explanation: The number of 40-ms intervals between timer "ticks" for the short DLC timer T1 (T1 timer values 1-5). If the value is zero, the default of 5 (200-400 ms) is used. See "Timers" on page 2-35.

DLC_T2_TICK_ONE

Explanation: The number of 40-ms intervals between timer "ticks" for the short DLC timer T2 (T2 timer values 1-5). If the value is zero, the default of 1 (40-80 ms) is used. See "Timers" on page 2-35.

DLC_Ti_TICK_ONE

Explanation: The number of 40-ms intervals between timer "ticks" for the short DLC timer Ti (Ti timer values 1-5). If the value is zero, the default of 25 (1-2 seconds) is used. See "Timers" on page 2-35.

DLC_T1_TICK_TWO

Explanation: The number of 40-ms intervals between timer "ticks" for the long DLC timer T1 (timer values 6-10). If the value is zero, the default of 25 (1-2 seconds) is used. See "Timers" on page 2-35.

DLC_T2_TICK_TWO

Explanation: The number of 40-ms intervals between timer "ticks" for the long DLC timer T2 (timer values 6-10). If the value is zero, the default of 10 (400-800 ms) is used. See "Timers" on page 2-35.

DLC_Ti_TICK_TWO

Explanation: The number of 40-ms intervals between timer "ticks" for the long DLC timer Ti (timer values 6-10). If the value is zero, the default of 125 (5-10 seconds) is used. See "Timers" on page 2-35.

See page 4-8 for the NETBIOS Open Parameters table.

DIR.OPEN.DIRECT

Hex 35

DIR.OPEN.DIRECT

Command Description: This command is for **CCB2 and CCB3 only**. It provides a single application program with the capability of receiving frames using the direct station.

Command Specifics: An application program is given ownership of the direct station and can receive frames once RECEIVE commands have been issued for the direct station. If the direct station has already been assigned to an application program, this command terminates with a CCB_RETCODE of X'63'. This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-13. CCB Parameter Table for DIR.OPEN.DIRECT

Offset	Parameter Name	Byte Length	8086 Type	Description
For CCB2 and CCB3:				
0	DIR_BUF_SIZE	2	DW	Size of buffers in the buffer pool
2	DIR_POOL_LENGTH	2	DW	Length of buffer pool
4	DIR_POOL_ADDRESS	4	DD	Starting address of buffer pool
8	OPEN_OPTIONS	2	DW	Open options

* Indicates a returned value.

DIR_BUF_SIZE

Explanation: This is the size of the entire buffer including all overhead.

The minimum length is 80 bytes, and the length must be a multiple of 16. If this value is zero, the default of 160 is used. The maximum size is dependent on the DIR_POOL_LENGTH parameter.

DIR_POOL_LENGTH

Explanation: This is the length of the direct buffer pool. This is the number of 16-byte blocks in the direct buffer pool. If the value is left as zero, the value is ignored.

DIR_POOL_ADDRESS

Explanation: This is the starting address of the buffer pool. If this value is zero, it is the responsibility of the application program to build its own buffer pool. If this is done, the DIR_BUF_SIZE parameter must reflect the size of these buffers.

OPEN_OPTIONS

Explanation: See the open options described in Appendix E, "Operating System/2 Extended Edition Information."

Note: The "wrap interface" and "token release" options are ignored if specified with this command. These options can only be specified with the configuration parameters.

DIR.READ.LOG

Hex 08

DIR.READ.LOG

Command Description: This command reads and resets the adapter logs. The log data is transferred to the buffer indicated in the parameter table.

Command Specifics: If the LOG_ID is X'0001', this command executes entirely in the workstation. The CCB_RETCODE is available upon return.

For CCB2 and CCB3: This command can be issued by a system administrator using the System Key as defined by the configuration parameters.

The system administrator or application program that issued the DIR.OPEN.DIRECT command and has been assigned ownership of the direct stations can issue this command without the System Key.

The System Key code is located in the CCB_PARAMETER_2 field. The System Key code is needed to read the adapter log and the direct interface log if the direct stations have not been assigned to the system administrator. For example, if an application program issued the DIR.OPEN.DIRECT command and has ownership of the direct stations, the System Key code has to match the System Key defined by the configuration parameters in order for a system administrator to read the logs as described above.

Application programs affected by a DIR.READ.LOG command issued with the System Key can be notified of the event. See "System Action Exceptions for OS/2 EE 1.3" on page B-69 for more information.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-14 (Page 1 of 2). CCB Parameter Table for DIR.READ.LOG

Offset	Parameter Name	Byte Length	8086 Type	Description
0	LOG_ID	2	DW	Identify the log to read
For CCB1:				
2	LOG_BUF_LENGTH	2	DW	Size of the buffer at LOG_BUF_ADDR
4	LOG_BUF_ADDR	4	DD	Address to place log data

* Indicates a returned value.

Table 3-14 (Page 2 of 2). CCB Parameter Table for DIR.READ.LOG

Offset	Parameter Name	Byte Length	8086 Type	Description
8	LOG_ACT_LENGTH	2	DW	Actual length of log *
For CCB2 and CCB3:				
2	LOG_BUF_LENGTH	2	DW	Size of the buffer at LOG_BUF_OFFSET
4	LOG_BUF_OFFSET	2	DW	Offset to place log data
6		2	DW	Reserved for the application program
8	LOG_ACT_LENGTH	2	DW	Actual length of log *

* Indicates a returned value.

LOG_ID

Explanation: Identifies the log to read as follows:

X'0000'	Adapter error log
X'0001'	Direct interface error log
X'0002'	Both logs.

LOG_BUF_LENGTH

Explanation:

For CCB1: The length of the buffer defined by LOG_BUF_ADDR.

For CCB2 and CCB3: The length of the buffer defined by LOG_BUF_OFFSET.

LOG_BUF_ADDR

Explanation: The address in workstation memory of the buffer (defined by the application program) where the log data is to be placed.

LOG_BUF_OFFSET

Explanation: The offset in workstation memory of the buffer (defined by the application program) where the log data is to be placed.

LOG_ACT_LENGTH

Explanation: The actual length of the log as returned by the adapter or protocol driver. If this value is greater than that defined by the LOG_BUF_LENGTH parameter, the full log was not transferred. The CCB_RETCODE will contain X'15'.

Note: When data is lost (CCB_RETCODE X'15'), the log is still reset.

Log Formats

There are two log formats.

Adapter Log

When one or more log counters reaches 255, the application program network status appendage routine will be taken with network status indicating a counter overflow. These counters are reset after they are read.

The information read from this log is 14 bytes long and returned to the buffer in the following order:

Table 3-15. Log Formats for the Adapter Log

Byte	8086 Type	Token-Ring Network	PC Network and Ethernet
0	DB	Line Error	CRC Errors **
1	DB	Internal error	Reserved
2	DB	Burst error	Alignment errors **
3	DB	A/C error	Reserved
4	DB	Abort delimiter	Transmit errors
5	DB	Reserved	Reserved
6	DB	Lost frame	Collision Errors **
7	DB	Receive congestion	Receive congestion errors
8	DB	Frame copied error	Reserved
9	DB	Frequency error	Reserved
10	DB	Token error	Reserved
11	DB	Reserved	Reserved
12	DB	Reserved	Reserved
13	DB	Reserved	Reserved

** This value is not returned by DIX Version 2.0 and IEEE 802.3 network adapters.

Direct Interface Log

The direct interface log consists of 18 bytes of counters and they are returned to the buffer in this order:

Table 3-16. Log Formats for the Direct Interface Log.

Bytes	8086 Type	Meaning
0-3	DD	Number of frames transmitted
4-7	DD	Number of frames received
8-11	DD	Number of frames discarded (no receive command)
12-15	DD	Number of times data was lost
16-17	DW	Number of buffers available in the SAP buffer pool

Both Adapter and Direct Interface Logs

Both logs will be placed in the buffer. The adapter log is first, followed by the direct interface log.

DIR.RESET.MULT.GROUP.ADDRESS

Hex 42

DIR.RESET.MULT.GROUP.ADDRESS

Command Description: This command is not available with some of the earlier IBM LAN support software. It resets multiple group addresses for NDIS adapters that can have more than one group address (for example, Ethernet adapters).

Command Specifics: Multiple group addresses are reset one at a time unless the RESET_ALL_FLAG parameter is set. If the RESET_ALL_FLAG parameter is set, all the multiple group addresses are reset at once.

Table 3-17 describes the parameter table for the DIR.RESET.MULT.GROUP.ADDRESS command.

Note: If the address to be deleted is not currently set, a return code of `X'00' Operation was completed successfully` is returned.

Table 3-17. DIR.RESET.MULT.GROUP.ADDRESS Parameter Table for CCB1

Offset	Parameter Name	Byte Length	8086 Type	Description
0	GROUP_ADDRESS	6	DB	Group address
6	Reserved	2	DW	Should be set to 0
8	RESET_ALL_FLAG	1	DB	Reset All Flag. If set to 1, all group addresses are reset.

Note: The use of the RESET_ALL_FLAG parameter resets all group addresses, even if a group address was set using the DIR.OPEN.ADAPTER command. If the RESET_ALL_FLAG is set, then the group address (if any) specified in the parameter table is ignored.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

For more information about the use of the DIR.RESET.MULT.GROUP.ADDRESS command, see "DIR.SET.MULT.GROUP.ADDRESS" on page 3-41.

DIR.RESTORE.OPEN.PARMS

Hex 02

DIR.RESTORE.OPEN.PARMS

Command Description: This command is for **CCB1 only**. It is used to restore adapter open parameter values modified by the DIR.MODIFY.OPEN.PARMS command.

Command Specifics: This command is rejected with a X'06' return code if a DIR.MODIFY.OPEN.PARMS has not previously been issued and completed successfully.

No parameter table is required. The parameter table pointer field is used as adapter work space.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DIR.SET.EXCEPTION.FLAGS

Hex 2D

DIR.SET.EXCEPTION.FLAGS

Command Description: This command is for CCB2 and CCB3 only. It defines user notification appendages and flags if an application program wants to be notified of exception conditions.

Command Specifics: This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver. If the appendages or flags are zero, the appendage flag is not defined and the application program will not be notified of the exception condition.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-18 (Page 1 of 2). CCB Parameter Table for DIR.SET.EXCEPTION.FLAGS. See "Exception Indications" on page B-46 for more information.

Offset	Parameter Name	Byte Length	8086 Type	Description
For CCB2:				
0	ADAPTER_CHECK_FLAG	4	DD	User notification flag for adapter check
4	NETWORK_STATUS_FLAG	4	DD	User notification flag for network status
8	PC_ERROR_FLAG	4	DD	User notification flag for workstation errors
12	SYSTEM_ACTION_FLAG	4	DD	User notification flag for system action
For CCB3:				
0	ADAPTER_CHECK_APPNDG_OFFSET	2	DW	Offset to the adapter check appendage
2		2	DW	Reserved for the application program
4	NETWORK_STATUS_APPNDG_OFFSET	2	DW	Offset to the network status appendage
6		2	DW	Reserved for the application program

Table 3-18 (Page 2 of 2). CCB Parameter Table for DIR.SET.EXCEPTION.FLAGS. See "Exception Indications" on page B-46 for more information.

Offset	Parameter Name	Byte Length	8086 Type	Description
8	PC_ERROR_APPNDG_OFFSET	2	DW	Offset to the workstation error appendage
10		2	DW	Reserved for the application program
12	SYSTEM_ACTION_APPNDG_OFFSET	2	DW	Offset to the system action appendage
14		2	DW	Reserved for the application program

For CCB2: The flag parameters are used to request that information relating to exception events be passed to the application program. You should issue this command right after issuing the DIR.OPEN.ADAPTER command if the application program is to be notified of any exception conditions. A READ command should then be issued with EVENT_SET bits 4 and 5 on. Whenever a critical exception or non-critical exception occurs, this READ command is executed, and the posting event and event error code will be returned from the protocol driver.

For CCB3: These offsets to application program appendages are used to request that information relating to exception events be passed to the application program. These parameters will consist of a 2-byte offset for the exception appendage that is passed in register DI to the application program when the protocol driver calls the application program. You should issue this command right after issuing DIR.OPEN.ADAPTER if the application program is to be notified of any exception conditions.

DIR.SET.FUNCTIONAL.ADDRESS

Hex 07

DIR.SET.FUNCTIONAL.ADDRESS

Command Description: This command sets the functional addresses for the adapter to receive messages. This command can be used to change (reset) the functional address values that were set earlier.

Command Specifics: If the most significant bit of the functional address field is **Off** (0), the bits in the adapter functional address corresponding to the bits provided in the functional address field of the CCB are set **On**. If the most significant bit of the field is **On** (1), the bits in the adapter functional address corresponding to the bits provided in the functional address field are reset. Depending on the LAN adapter, it may or may not be possible to set bit 1. Bits 31 and 0 are not affected by this command but are set by the adapter.

For NDIS Adapters, All CCBs: Although not all NDIS adapters support functional addresses, the DIR.SET.FUNCTIONAL.ADDRESS command is accepted when an NDIS adapter is used. The FUNCTIONAL_ADDR parameter of the DIR.OPEN.ADAPTER command is also accepted. Instead of setting a bit in the functional address, the protocol driver adds one group address for each separate bit set in the functional address. For example, if bits 6 and 7 of byte 5 of the functional address are set, then 2 group addresses are added. One group address has bit 6 of its byte 5 set, and the other group address has bit 7 of its byte 5 set. Each group address represents one of the 2 functional address bits.

The functional address is visible as the FUNCTIONAL_ADDRESS parameter of DIR.STATUS. For CCB1, the group addresses that are serving as functional addresses are not visible in the GROUP_ADDRESS_LIST table pointed to by the GROUP_ADDRESS_LIST_ADDR parameter of the DIR.STATUS extended status table. Therefore, to calculate the number of group addresses serving to identify functions, count the bits set in the functional address parameter.

For example, if the functional address parameter of DIR.STATUS shows 2 bits set, 2 of the group addresses available are being used to identify functions.

The broadcast address X'C000 FFFF FFFF' is set by the IBM NDIS MAC driver so that existing token-ring applications which use this broadcast address will not need to be changed.

The code keeps track of the number of remaining multiple group addresses, and if the number of bits to be set by the DIR.SET.FUNCTIONAL.ADDRESS command exceeds the number of available multiple group addresses, the command is not executed and a return code of X'1E' is posted in the CCB.

For CCB1: The functional address bits to be changed are placed in the CCB_PARM_TAB field, but are formatted as DB. The least significant bit is ignored and both the least and most significant bit are always set to zero in the adapter. For example, X'FFFFFFFF' resets all functional address bits, or X'00000060' sets bits 5 and 6.

For CCB2 and CCB3: The functional address bits to be set or reset are placed in the CCB_PARM_OFFSET and CCB_PARAMETER_1 fields but are formatted as DB. For example, X'00000060' sets bits 5 and 6, or X'80000040' resets bit 6.

This command can be issued by a system administrator using the System Key as defined by the configuration parameters.

This function is provided to allow an application program to set bits of the functional address. Bits that have been previously set by application programs are the only bits that can be reset without the System Key. The bits set by the configuration parameters cannot be reset without the System Key. The System Key, if used, should be coded in the CCB_PARAMETER_2 field.

Application programs affected by a DIR.SET.FUNCTIONAL.ADDRESS command issued with the System Key can be notified of the event. See "System Action Exceptions for OS/2 EE 1.3" on page B-69 for more information.

For Ethernet adapters, because functional addresses are treated as multicast addresses by Ethernet adapters, only one functional (multicast) address can be enabled at a time.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DIR.SET.GROUP.ADDRESS

Hex 06

DIR.SET.GROUP.ADDRESS

Command Description: This command sets the group address for which the adapter will receive messages. This command can be used to change the group address value that was set earlier.

Command Specifics: If no group address is desired, set the value to X'00000000'.

For CCB1: The group address to be set is placed in the CCB_PARM_TAB field but is formatted as DB.

For CCB2 and CCB3: The group address to be set is placed in the CCB_PARM_OFFSET field but is formatted as DB. This command can only be issued by a system administrator using the System Key as defined by the configuration parameters. The System Key should be coded in the CCB_PARAMETER_2 field.

Application programs affected by a DIR.SET.GROUP.ADDRESS can be notified of the event. See "System Action Exceptions for OS/2 EE 1.3" on page B-69 for more information.

For Ethernet Adapters: The group address will be set using the NDIS AddMulticastAddress command. If there was a previous group address in use by this adapter, an NDIS Delete Multicast Address command will be issued first.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DIR.SET.MULT.GROUP.ADDRESS

Hex 41

DIR.SET.MULT.GROUP.ADDRESS

Command Description: This command is not available with some of the earlier IBM LAN support software. It supports NDIS adapters that can maintain more than one group address. The Local Area Network Support Program supports up to 64 group addresses per adapter using *multiple group address support*.

Command Specifics: The DIR.SET.MULT.GROUP.ADDRESS command is used to set multiple group addresses (one at a time). The number of addresses up to 64 that can actually be set depends on the number of group addresses that the adapter supports. For example, if an adapter supports a total of 12 group addresses, one group address is used as a broadcast address, so 11 group addresses are available to the application program.

The DIR.SET.GROUP.ADDRESS command and either of the commands DIR.SET.MULT.GROUP.ADDRESS or DIR.RESET.MULTIPLE.GROUP.ADDRESS cannot be used together in one session. If an application program issues a DIR.SET.MULT.GROUP.ADDRESS command or a DIR.RESET.MULT.GROUP.ADDRESS command, and it then attempts to issue a DIR.SET.GROUP.ADDRESS command, it receives the return code `X'01' Invalid` command. The DIR.SET.GROUP.ADDRESS command cannot be used again until a DIR.CLOSE.ADAPTER or DIR.INITIALIZE command is issued.

On the other hand, once an application program has issued a DIR.SET.GROUP.ADDRESS command, the DIR.SET.MULT.GROUP.ADDRESS and the DIR.RESET.MULT.GROUP.ADDRESS commands are returned with an X'01' Invalid command return code until a DIR.CLOSE.ADAPTER or DIR.INITIALIZE command is issued.

The group address set by the DIR.OPEN.ADAPTER command is reported in the 4-byte group address field of the DIR.STATUS CCB parameter table and in the GROUP_ADDRESS_LIST of the DIR.STATUS extended status table. This group address is replaced if the application program issues a DIR.SET.GROUP.ADDRESS command, but it is still valid after a DIR.SET.MULT.GROUP.ADDRESS or DIR.RESET.MULT.GROUP.ADDRESS command unless the DIR.RESET.MULT.GROUP.ADDRESS command resets the address or sets the RESET_ALL_FLAG.

The Local Area Network Support Program filters the setting of the all-stations broadcast address. The value of the all-stations broadcast address is X'FFFF FFFF FFFF' for all adapters and X'C000 FFFF FFFF' to address only adapters that conform to the IEEE 802.5 standard. A return code of X'36' Not a valid group address is returned if the application program attempts to use the DIR.SET.MULT.GROUP.ADDRESS command to set either one of these all-stations broadcast addresses, which are recognized by the adapter hardware and are not considered group addresses.

The group addresses supplied in the DIR.SET.MULT.GROUP.ADDRESS and DIR.RESET.MULT.GROUP.ADDRESS commands are checked to make sure the Individual/Group (I/G) bit is set (for Group). If it is not set, the command is returned with an X'36' return code. If the I/G bit is set and the group address value is not within the range acceptable for the adapter, the command also returns an X'36' return code.

If the address to be added has been added previously, a return code of X'00' Operation was completed successfully is returned.

Table 3-19 describes the parameter table associated with the DIR.SET.MULT.ADDRESS command.

Table 3-19. DIR.SET.MULT.GROUP.ADDRESS Parameter Table for CCB1

Offset	Parameter Name	Byte Length	8086 Type	Description
0	GROUP_ADDRESS	6	DB	Group address
6	Reserved	2	DW	Should be set to 0

Notes:

1. The DIR.SET.GROUP.ADDRESS and DIR.OPEN.ADAPTER commands currently implemented by the Local Area Network Support Program Version 1.2 support only a 4-byte group address. The DIR.SET.MULT.GROUP.ADDRESS and DIR.RESET.MULT.GROUP.ADDRESS commands require a 6-byte group address.
2. If the Functional Address Indicator Bit of the GROUP_ADDRESS parameter is not set when using the DIR.SET.GROUP.ADDRESS and DIR.OPEN.ADAPTER commands, the Local Area Network Support Program automatically sets it.

This bit is **not** automatically set for the DIR.SET.MULT.GROUP.ADDRESS and DIR.RESET.MULT.GROUP.ADDRESS commands.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DIR.SET.USER.APPENDAGE

Hex 2D

DIR.SET.USER.APPENDAGE

Command Description: This command is for **CCB1 only**. It changes the appendage addresses set by the DIR.OPEN.ADAPTER, DIR.INITIALIZE, and DIR.MODIFY.OPEN.PARMS commands.

Command Specifics: This command sets the appendage addresses as defined in the CCB parameter table if the CCB_PARM_TAB field value is other than zero. If the CCB_PARM_TAB field is set to zero, the appendage addresses are restored to their values before the last DIR.SET.USER.APPENDAGE command was completed successfully.

This command can be issued only when the adapter is open. If the adapter was opened with an OPEN_LOCK, this command takes no action and is executed with a return code of X'00'. This command is executed entirely by the protocol driver in the workstation. Therefore, the command completion appendage is not required, as the command is complete upon return. However, the command completion appendage is taken if provided.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-20. CCB Parameter Table for DIR.SET.USER.APPENDAGE

Offset	Parameter Name	Byte Length	8086 Type	Description
0	ADPT_CHK_EXIT	4	DD	I/O appendage, adapter check
4	NETW_STATUS_EXIT	4	DD	I/O appendage, network status change
8	PC_ERROR_EXIT	4	DD	I/O appendage, error in workstation

See the parameter descriptions of the DIR.OPEN.ADAPTER command beginning on page 3-17.

DIR.STATUS

Hex 21

DIR.STATUS

Command Description: This command reads the general status information.

Command Specifics: This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver.

Note: This command can be issued before the DIR.OPEN.ADAPTER command, but after the DIR.INITIALIZE command. The System Key is not required when this command is issued before the DIR.OPEN.ADAPTER command.

For CCB1: The status information is pointed to by the address in the CCB_PARM_TAB field when the command is completed.

For CCB2 and CCB3: The status information is pointed to by the offset in the CCB_PARM_OFFSET field when the command is completed.

If a valid CCB_APPL_ID is not being used, only polling of the CCB_RETCODE can be used for posting of the command completion. In this case, a successful completion results in a return code of X'0C'. If a valid CCB_APPL_ID is used, this command can be posted like any other command. A successful completion results in a return code of X'00'.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-21 (Page 1 of 3). CCB Parameter Table for DIR.STATUS

Offset	Parameter Name	Byte Length	8086 Type	Description
0	ENCODED_ADDR	6	DB	The adapter's permanent encoded address *
6	NODE_ADDRESS	6	DB	This adapter's network address *
12	GROUP_ADDRESS	4	DB	This adapter's group address *
16	FUNCTIONAL_ADDR	4	DB	This adapter's functional address *
20	MAX_SAP	1	DB	Maximum allowable SAPs *
21	OPEN_SAP	1	DB	Number of SAPs open *
22	MAX_STATION	1	DB	Maximum allowable link stations *
23	OPEN_STATION	1	DB	Number of link stations open *

* All entries are returned by the protocol driver to the parameter table in workstation memory.
** This parameter is not set on Ethernet.
Note: The ADAPTER_PARMS_ADDR parameter is returned for CCB1 when the supported adapter is a Token-Ring Network adapter and is not returned when the supported adapter is an Ethernet adapter.

Table 3-21 (Page 2 of 3). CCB Parameter Table for DIR.STATUS

Offset	Parameter Name	Byte Length	8086 Type	Description
24	AVAIL_STATION	1	DB	Number of link stations available *
25	ADAPTER_CONFIG	1	DB	Adapter configuration bits *
26	MICROCODE_LEVEL	10	DB	The adapter's microcode level *
For CCB1:				
36	ADAPTER_PARMS_ADDR	4	DD	Shared RAM address of adapter parms * **
40	ADAPTER_MAC_ADDR	4	DD	Shared RAM address of adapter MAC buffer **
44	TICK_CNTR_ADDR	4	DD	Address of the timer tick counter *
48	LAST_NTWK_STATUS	2	DW	Most recent network status issued *
50	EXTENDED_STATUS_ADDR	4	DD	Address of the extended status table.
For CCB2:				
36	ADAPTER_PARMS_ADDR	4	DD	Address of adapter parms *
40	ADAPTER_MAC_ADDR	4	DD	Address of adapter MAC buffer *
44	TICK_CNTR_ADDR	4	DD	Address of the timer tick counter *
48	LAST_NTWK_STATUS	2	DW	Most recent network status issued *
50	ADAPTER_TYPE	2	DW	Network adapter type *
For CCB3:				
36	ADAPTER_PARMS_ADDR	4	DD	Address with GDT selector of adapter parms *
40	ADAPTER_MAC_ADDR	4	DD	Address with GDT selector of adapter MAC buffer *
44	TICK_CNTR_ADDR	4	DD	Address with GDT selector of the timer tick counter *
48	LAST_NTWK_STATUS	2	DW	Most recent network status issued *

* All entries are returned by the protocol driver to the parameter table in workstation memory.
** This parameter is not set on Ethernet.
Note: The ADAPTER_PARMS_ADDR parameter is returned for CCB1 when the supported adapter is a Token-Ring Network adapter and is not returned when the supported adapter is an Ethernet adapter.

Table 3-21 (Page 3 of 3). CCB Parameter Table for DIR.STATUS

Offset	Parameter Name	Byte Length	8086 Type	Description
50	ADAPTER_TYPE	2	DW	Network adapter type *

* All entries are returned by the protocol driver to the parameter table in workstation memory.
** This parameter is not set on Ethernet.
Note: The ADAPTER_PARMS_ADDR parameter is returned for CCB1 when the supported adapter is a Token-Ring Network adapter and is not returned when the supported adapter is an Ethernet adapter.

ENCODED_ADDR

Explanation: The permanent address encoded on the adapter.

NODE_ADDRESS

Explanation:

For CCB1: The adapter's network address as set by the DIR.OPEN.ADAPTER command.

For CCB2 and CCB3: The adapter's network address as set by the configuration parameters.

GROUP_ADDRESS

Explanation: The adapter's group address.

FUNCTIONAL_ADDR

Explanation: The adapter's functional address.

MAX_SAP

Explanation: The maximum number of SAPs allowed.

OPEN_SAP

Explanation: The number of SAPs that have been opened by DLC.OPEN.SAP commands.

MAX_STATION

Explanation:

For CCB1: The maximum number of link stations allowed as set by the DIR.OPEN.ADAPTER command.

For CCB2 and CCB3: The maximum number of link stations allowed as set by the configuration parameters.

OPEN_STATION

Explanation: The number of link stations that have been opened by DLC.OPEN.STATION commands or the DLC.CONNECT.STATION command as a result of receiving a SABME.

AVAIL_STATION

Explanation: The number of link stations that have not been previously reserved by DLC.OPEN.SAP or DLC.REALLOCATE commands.

ADAPTER_CONFIG

Explanation: Following are the descriptions of the adapter configuration bits:

Bits	Description
7	**For CCB1:** The Original PC Network Adapter is present. **For CCB2 and CCB3:** Reserved.
6	Reserved.
5	**For CCB1:** This bit indicates that extended status information is being requested. If this bit is set to 1 when the DIR.STATUS command is issued, the protocol driver will return the address of the extended status table at offset 50 of the CCB parameter table. **Note:** If the extended status function is not supported, the value in offset 50 of the CCB parameter table will not be changed. To ensure the pointer is valid, the user should set it to zero when building the CCB, and check it for a non-zero value on return.
4	Token release: This bit is only available for 16/4 adapters when operating at 16 Mbps. If not set, 16 Mbps adapters will get early token release; that is the default. Setting this bit on selects no early token release for an adapter operating at 16 Mbps.
3–2	For IBM Token-Ring Network Adapters with shared RAM, bits 3–2 represent the KBs of shared RAM mapped into workstation memory. (B'00'=8 KB, B'01'=16 KB, B'10'=32 KB, and B'11'=64 KB. For example, if bits 3 and 2 are 0 and 1, respectively, then shared RAM size is 16 KB.) **Note:** In the case of a 4/16 adapter with RAM paging active, the size of shared RAM mapped into memory is 16 KB, while the bits would indicate 64 KB — the size of RAM on the card.
1	Reserved.
0	Adapter data rate (0=4 Mbps and 1=16 Mbps).

MICROCODE_LEVEL

Explanation: The number representing the engineering level of the adapter microcode. Table 3-22 shows its structure.

Table 3-22. MICROCODE_LEVEL Fields

Offset	Byte Length	Description
0	4	Microcode engineering level
4	2	Device driver version
6	2	Device driver revision level
8	2	Zeros

ADAPTER_PARMS_ADDR

Explanation: The address of a workstation read-only region of adapter parameters.

See "Adapter Status Parameter Table" on page B-44 for the parameter table layout.

For CCB2: An entry for this memory segment is made into the LDT of the process that issues this command. Only the process that issues this command will have access to this memory area.

For CCB3: This memory segment has been mapped in the GDT.

ADAPTER_MAC_ADDR

Explanation: The address of a workstation read-only region of adapter MAC buffer.

For CCB2: An entry for this memory segment is made into the LDT of the process that issues this command. Only the process that issues this command will have access to this memory area.

For CCB3: This memory segment has been mapped in the GDT.

TICK_CNTR_ADDR

Explanation:

For CCB1: The address of a 4-byte field containing the number of 100-ms timer interrupts received from the adapter's timer since the last DIR.INITIALIZE command. The tick counter can be read, but should not be written.

For CCB2: The address of a 4-byte field containing the number of 100-ms timer interrupts received from the adapter's timer since the last physical open by a DIR.OPEN.ADAPTER command. The tick counter can be read, but should not be written.

An entry for this memory segment is made into the LDT of the process that issues this command. Only the process that issues this command will have access to this memory area.

For CCB3: The address of a 4-byte field containing the number of 100-ms timer interrupts received from the adapter's timer since the last physical open by a DIR.OPEN.ADAPTER command. The tick counter can be read, but should not be written.

This memory segment has been mapped in the GDT.

LAST_NTWK_STATUS

Explanation: The most recent network status change. See "Network Status" on page B-52 for an explanation of the network status bytes.

EXTENDED_STATUS_ADDR

Explanation:

For CCB1: The address of a table containing additional status information. Refer to the following table (Table 3-23) for the table layout. Extended status is not supported on the original PC Network Adapter.

Parameters: Note that all values are returned values.

Table 3-23. Extended Status Table

Offset	Parameter Name	Byte Length	8086 Type	Description
0	EXT_TABLE_LENGTH	1	DB	The length of the table in bytes.
1	RAM_PAGE_FRAME_SIZE	1	DB	The RAM page frame size in KB.
2	ADAPTER_TYPE	2	DW	A hexadecimal value indicating the type of network adapter installed.
4	CURRENT_DHB_SIZE	2	DW	The current data holding buffer (DHB) length in bytes.
6	MAXIMUM_DHB_SIZE	2	DW	The maximum DHB length in bytes.
8	RING_UTILIZATION	2	DW	The ring utilization retrieved from the network.
10	GROUP_ADDR_LIST_ADDR	4	DD	Address of the GROUP_ADDRESS_LIST table.
14	LLC_TYPE	1	DB	In LAN Support Program 1.33 and higher, this byte is X'01' when the protocol is running on an NDIS MAC driver.

ADAPTER_TYPE

Explanation:

For CCB1: This parameter is returned by the DIR.STATUS command only if bit 5 of the ADAPTER_CONFIG byte is set to 1 when the command is issued. It is returned as a parameter in the Extended Status Table.

For CCB2 and CCB3: This parameter is returned in the CCB Parameter Table.

The following list defines the values of the parameter.

X'0001'	IBM Token-Ring Network PC Adapter
X'0002'	IBM Token-Ring Network PC Adapter II
X'0004'	IBM Token-Ring Network Adapter/A
X'0008'	IBM Token-Ring Network PC Adapter II

X'0010'	Unknown or OEM token-ring adapter
X'0011'	IBM Token-Ring Network 16/4 Busmaster Server Adapter/A
X'0012'	IBM LANStreamer* MC 32 Adapter
X'0020'	IBM Token-Ring Network 16/4 Adapter
X'0040'	IBM Token-Ring Network 16/4 Adapter/A
X'4000'	IBM PC Network Adapter
X'8000'	IBM PC Network Adapter/A
X'0100'	Ethernet adapters
X'0200'	3174 Peer Communication
X'0300'	Reserved
X'0400'	FDDI Adapters
X'0401'	IBM FDDI Adapters
X'0500'	Reserved

GROUP_ADDRESS_LIST

Explanation: The GROUP_ADDRESS_LIST is always valid even if the application program is using only the single group address implementation (for example, DIR.SET.GROUP.ADDRESS).

A group address set by the DIR.OPEN.ADAPTER command is reported in the GROUP_ADDRESS parameter of the DIR.STATUS parameter table as currently defined in the *IBM Local Area Network Technical Reference.* This group address is also duplicated in the GROUP_ADDRESS_LIST table described in Table 3-24. However, this 4-byte group address field is maintained only if single group addressing is done. This field is not valid (or is not maintained) when implementing multiple group addressing.

Table 3-24. GROUP_ADDRESS_LIST

Offset	Parameter Name	Byte Length	8086 Type	Description
0	MAX_NUM_GRP_ADDRESSES	2	DW	Maximum number of group addresses
2	NUM_SINGLE_GRP_ADDRESSES	2	DW	Current number of group addresses defined (N)
4	GROUP_ADDRESS 1	6	DB	Group address 1
10	GROUP_ADDRESS 2	6	DB	Group address 2
4 + ((N-1) x 6)	GROUP_ADDRESS N	6	DB	Group address N

DIR.TIMER.CANCEL

Hex 23

DIR.TIMER.CANCEL

Command Description: This command cancels a timer that was previously initiated by a DIR.TIMER.SET command.

Command Specifics: This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver.

When the DIR.TIMER.CANCEL command is completed, the DIR.TIMER.SET command is terminated with a CCB_RETCODE value of X'0A', and the DIR.TIMER.SET command is not posted except for the setting of the return code. The following CCB field points to the address of the DIR.TIMER.SET command that is to be canceled.

For CCB1: The CCB_PARM_TAB field.

For CCB2 and CCB3: The CCB_PARM_OFFSET and CCB_PARAMETER_1 fields are combined to form a 4-byte (8086 declaration type DD) address of the timer to be canceled.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DIR.TIMER.CANCEL.GROUP

Hex 2C

DIR.TIMER.CANCEL.GROUP

Command Description: This command cancels a group of timer commands that were previously initiated by DIR.TIMER.SET commands.

Command Specifics: This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver.

For CCB1: The CCB_PARM_TAB field points to the address of a timer completion appendage. All timers that have that address as a timer completion appendage are terminated with a return code of X'0A'. When the DIR.TIMER.SET command is terminated, the command completion appendage is not taken. All canceled CCBs will be pointed to by the CCB_POINTER field of the DIR.TIMER.CANCEL.GROUP command. This command can be issued while the adapter is closed.

For CCB2: The DIR.TIMER.CANCEL.GROUP command's CCB_PARM_OFFSET and CCB_PARAMETER_1 fields contain a TIMER_CMPL_FLAG. All timer set commands that have been issued with their CCB_CMPL_FLAG equal to the TIMER_CMPL_FLAG will be canceled with a CCB_RETCODE of X'0A'.

No command completion is done for the DIR.TIMER.SET commands other than setting the CCB_RETCODE to X'0A'. The DIR.TIMER.SET command's CCBs are chained using the CCB_POINTER field of the DIR.TIMER.CANCEL.GROUP command CCB.

For CCB3: The DIR.TIMER.CANCEL.GROUP command's CCB_PARM_OFFSET and CCB_PARAMETER_1 fields contain a TIMER_CMPL_OFFSET. This offset is used to specify the address of the command completion appendage (CCB_APPNDG_OFFSET) of all pending DIR.TIMER.SET commands that are to be canceled. These commands will be canceled with a CCB_RETCODE of X'0A'.

No command completion is done for the DIR.TIMER.SET commands other than setting the CCB_RETCODE to X'0A'. The DIR.TIMER.SET command's CCBs are chained using the CCB_POINTER field of the DIR.TIMER.CANCEL.GROUP command CCB.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DIR.TIMER.SET

Hex 22

DIR.TIMER.SET

Command Description: This command sets a special programmable timer to expire in some multiple of half-second periods. On the PC Network and Ethernet, the value is not actually half-seconds, but 491.5-ms periods:

- For the Token-Ring Network: From 0 to 13 107 (109 minutes)
- For the PC Network: From 0 to 13 107 (107.3 minutes)
- For Ethernets: From 0 to 13 107 (109 minutes, on average).

The timer expires after the next tick if the value is set to zero.

Command Specifics:

For CCB1: The timer value is coded in the first 2 bytes of the CCB_PARM_TAB field.

When the specified time expires, the command completes with the CCB_RETCODE set to X'00'.

This command can be issued any time after the adapter has been initialized while the adapter is either open or closed, but if a command follows that changes the adapter open or closed state, all pending timers will be canceled. The number of timers that can be set is limited only by the number of commands that are pending.

For CCB2: The timer value is coded in the CCB_PARM_OFFSET field.

When the allotted time expires, the command is executed, as defined by the CCB_CMPL_FLAG and the CCB_SEMAPHORE parameters with the CCB_RETCODE set to X'00'.

For CCB3: The timer value is coded in the CCB_PARM_OFFSET field.

When the allotted time expires, the command completion appendage is called, if defined. The CCB_RETCODE is set to X'00'.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DLC.CLOSE.SAP

Hex 16

DLC.CLOSE.SAP

Command Description: This command closes (deactivates) a SAP.

Command Specifics: If any station associated with the SAP is open, the command terminates with a CCB_RETCODE of X'47'. The SAP cannot close unless all link stations are closed.

If an error code X'47' occurs when a DLC.CLOSE.SAP command closely follows a DLC.CLOSE.STATION command for the last open station for that SAP, reissue the DLC.CLOSE.SAP command.

If a RECEIVE command is pending for the SAP, it terminates with a return code of X'0A', and the RECEIVE command's CCB address is placed in the CCB_POINTER field of the command CCB.

For CCB1: The station ID of the SAP to be closed is placed in the first 2 bytes of the CCB_PARM_TAB field, and the second 2 bytes are reserved.

If a RECEIVE command is pending, the command completion appendage of the RECEIVE command is not taken, but the CCB of the RECEIVE command is returned to the application program in the CCB_POINTER field of the DLC.CLOSE.SAP command.

For CCB2: The station ID of the SAP to be closed is placed in the CCB_PARM_OFFSET field. If a link station is still open for this SAP, the station ID (of the link station not closed) is returned in the CCB_PARAMETER_1 field.

If a RECEIVE command is pending for the SAP, it is not put on the completion list. A semaphore is not cleared to post its command completion. If the application program wants to receive pointers to all pending CCBs for this SAP, the CCB_CMPL_FLAG of the DLC.CLOSE.SAP command must be set. When a READ command is issued or if one is pending that requests notification of command completions, it posts the completed DLC.CLOSE.SAP command. The outstanding data area pointers are returned in the parameter table of the READ command. If a READ command is not already pending or if one is not chained to the CCB_POINTER field to post the DLC.CLOSE.SAP when it is completed, the DLC.CLOSE.SAP command completion event is placed on a completion queue. Since the SAP was closed when this command was executed successfully, the only way to retrieve the event is by issuing a READ command with the OPTION_INDICATOR field set to match on all events. This event cannot be retrieved by posting a READ with the OPTION_INDICATOR set to match on the SAP that was closed, since it is no longer valid.

For CCB3: The station ID of the SAP to be closed is placed in the CCB_PARM_OFFSET field. If a link station is still open for this SAP, the station ID (of the link station not closed) is returned in the CCB_PARAMETER_1 field.

If a RECEIVE command is pending, the command completion appendage of the RECEIVE command is not taken, but the CCB of the RECEIVE command is returned to the application program in the CCB_POINTER field of the DLC.CLOSE.SAP command. In addition to the RECEIVE command, buffers from the SAP buffer pool also return in the information table.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DLC.CLOSE.STATION

Hex 1A

DLC.CLOSE.STATION

Command Description: This command closes (deactivates) a link station.

Command Specifics: If this command is issued while the ring is beaconing, it cannot complete until the ring is no longer beaconing. Refer to the *IBM Token-Ring Network Architecture Reference* for a description of ring beaconing.

For CCB1: The station ID of the link station to be closed is placed in the first 2 bytes of the CCB_PARM_TAB field and the second 2 bytes are reserved.

If a RECEIVE command is pending for this link station, it terminates with a CCB_RETCODE of X'0A' and its address is placed in the CCB_POINTER of the DLC.CLOSE.STATION command.

The CCB_CMD_CMPL appendage of the RECEIVE command is not taken. Any pending TRANSMIT commands are aborted immediately.

For CCB2: The station ID of the link station to be closed is placed in the CCB_PARM_OFFSET field.

If a RECEIVE command is pending for this link station, it terminates with a CCB_RETCODE of X'0A' and its address is placed in the CCB_POINTER of the DLC.CLOSE.STATION command.

No command completion notification occurs for the RECEIVE command other than setting the return code. If the application program wants to receive pointers to all pending CCBs for this station, the CCB_CMPL_FLAG of the DLC.CLOSE.STATION command must be set. When a READ command is issued or if one is pending that requests notification of command completions, it posts the completed DLC.CLOSE.STATION command. The outstanding data area pointers are returned in the parameter table of the READ command. If a READ command is not already pending or if one is not chained to the CCB_POINTER field to post the DLC.CLOSE.STATION when it is completed, the DLC.CLOSE.STATION command completion event is placed on a completion list. Since the link station was closed when this command was completed successfully, the only way to retrieve the event is by issuing a READ command with the OPTION_INDICATOR field set to match on all events or to match on the SAP that owned the link station. This event cannot be retrieved by posting a READ with the OPTION_INDICATOR set to match on the link station that was closed, since it is no longer valid.

For CCB3: The station ID of the link station to be closed is placed in the CCB_PARM_OFFSET field.

If a RECEIVE command is pending, the command completion appendage of the RECEIVE command is not taken, but the CCB of the RECEIVE command is returned to the application program in the CCB_POINTER field of the DLC.CLOSE.SAP command. When a DLC.CLOSE.STATION command is issued, the RECEIVE command associated with that station terminates with a CCB_RETCODE of X'0A', and the CCB of the RECEIVE command is returned to the application program in the information table passed to the completion appendage of the DLC.CLOSE.STATION command.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DLC.CONNECT.STATION

Hex 1B

DLC.CONNECT.STATION

Command Description: This command starts or completes a SABME-UA exchange to place both the local and remote link stations in a data transfer state.

Command Specifics:

For CCB1: The CCB_PARM_TAB parameter points to the parameter table.

For CCB2 and CCB3: The CCB_PARM_OFFSET parameter is the offset to the parameter table.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-25. CCB Parameter Table for DLC.CONNECT.STATION

Offset	Parameter Name	Byte Length	8086 Type	Description
0	STATION_ID	2	DW	Link station ID to be connected
2		2	DW	Reserved
For CCB1:				
4	ROUTING_ADDR	4	DD	Address in workstation memory of 18 bytes of network routing information
For CCB2 and CCB3:				
4	ROUTING_OFFSET	2	DW	Offset in workstation memory of 18 bytes of network routing information
6		2	DW	Reserved for the application program Can be used for the segment or selector.

STATION_ID

Explanation: This parameter is the station ID of the link station.

ROUTING_ADDR and **ROUTING_OFFSET**

Explanation: This parameter is the address or offset to a memory location where the routing information is found. If the remote partner for this link station is on a different ring, routing information is necessary for frames to be exchanged. If the link station was established because of a DLC.OPEN.STATION command, the routing information must be provided with this command. If the link station was established because of receipt of a SABME from the remote partner, the adapter obtains the routing information from the received frame and ignores any provided with this command. The DLC.CONNECT.STATION command can also be used to provide new routing information if there is a link failure. The information must be provided in the format in which it will be used in transmitted frames. If this field is set to zero, or if the length field of the routing information field is zero and no SABME is pending, then the remote partner is assumed to be on the same ring. See the description of bit 10 in "DLC Status Codes" on page B-28. Refer to the *IBM Token-Ring Network Architecture Reference* for more about routing information and exchange identification (XID). Also, refer to any documentation related to implementation of bridges in your network.

DLC.FLOW.CONTROL

Hex 1D

DLC.FLOW.CONTROL

Command Description: This command controls the flow of data across a specified link station on a SAP, or every link station on a SAP, by setting and resetting a local busy status. See "Link Station States" on page 2-33.

Command Specifics:

For CCB1: The CCB_PARM_TAB field contains the station ID and the FLOW_CONTROL option byte. The station ID is the first 2 bytes, the FLOW_CONTROL (option byte) is the third byte, and the last byte is reserved.

For CCB2 and CCB3: The CCB_PARM_OFFSET field contains the station ID. The CCB_PARAMETER_1 field contains the FLOW_CONTROL (option byte); the FLOW_CONTROL (option byte) is the first byte. The second byte is reserved.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

STATION_ID

Explanation: The first 2 bytes of the CCB_PARM_TAB parameter contain either the station ID of a specific link station of a SAP or the station ID of a SAP. If the ID is a SAP ID, all the link stations on the SAP are affected. If the ID is a link station ID, only that specific station is controlled.

FLOW_CONTROL

Explanation: Contains bits that define options. Bit 7 is the high-order bit (leftmost bit position). Following are the complete descriptions of the FLOW_CONTROL bits:

Bits	Description
7	Used to set or reset a local-busy state • If this bit is off (0), the link station enters the local-busy state (bit 6 is ignored). • If this bit is on (1), then the local-busy state is reset based on the condition of bit 6.
6	Used to indicate the type of local-busy state that is being reset (bit 7=1). • If this bit is off (0), it indicates a "user-set" local-busy state is to be reset. • If this bit is on (1), it indicates a local-busy state caused by either an "out-of-receive-buffers" state, or "no-receive command-outstanding" state. The state will be reset.
0-5	Reserved. These bits should be set to zeros, but are not checked by the adapter.

DLC.MODIFY

Hex 1C

DLC.MODIFY

Command Description: This command modifies certain work values of an open link station or the default values of a SAP.

Command Specifics: This command allows you to alter the values without the need to close and reestablish the SAP and links. The values to be modified are contained in the parameter table pointed to by the CCB_PARM_TAB or CCB_PARM_OFFSET field.

For CCB1: The CCB_PARM_TAB field points to the parameter table.

For CCB2 and CCB3: The CCB_PARM_OFFSET field points to the parameter table.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-26 (Page 1 of 2). CCB Parameter Table for DLC.MODIFY

Offset	Parameter Name	Byte Length	8086 Type	Description
0		2	DW	Reserved.
2	STATION_ID	2	DW	SAP station or link station ID.
4	TIMER_T1	1	DB	T1 value (response timer).

Table 3-26 (Page 2 of 2). CCB Parameter Table for DLC.MODIFY

Offset	Parameter Name	Byte Length	8086 Type	Description
5	TIMER_T2	1	DB	T2 value (acknowledgment timer).
6	TIMER_TI	1	DB	Ti value (inactivity timer).
7	MAXOUT	1	DB	Maximum transmits without a receive acknowledgment.
8	MAXIN	1	DB	Maximum receives without a transmit acknowledgment.
9	MAXOUT_INCR	1	DB	Dynamic window increment value.
10	MAX_RETRY_CNT	1	DB	N2 value.
11		3	DB	Reserved.
14	ACCESS_PRIORITY	1	DB	Ring access priority. Access priority is not recognized on the PC network or on Ethernet.
15		4	DB	Reserved.
19	GROUP_COUNT	1	DB	Length of data in GROUP_LIST or GROUP_LIST_OFFSET.
For CCB1:				
20	GROUP_LIST	4	DD	Address of a list of Group SAP values.
For CCB2 and CCB3:				
20	GROUP_LIST_OFFSET	2	DW	Offset to a list of group SAP values.
22		2	DW	Reserved for the application program. Can be used for the segment or selector.

STATION_ID

Explanation: The link station ID whose working values are to be changed, or the SAP ID whose defaults are to be changed.

TIMER_T1

Explanation: Specifies the time period between 1 and 10 used to determine an inoperative condition on a link. The time intervals are based on the tick values specified by the DIR.OPEN.ADAPTER command or by configuration parameters. If the value is zero, the current value remains in effect. See "Timers" on page 2-35 for a complete description of timers.

TIMER_T2

Explanation: Specifies the timer value for the T2 timer used to delay transmission of an acknowledgment for a received I-LPDU for a link station being modified in this SAP. The time intervals are based on the tick values specified by the DIR.OPEN.ADAPTER command or by configuration parameters. If the value is zero, the current value remains in effect. If the value is greater than 10, the timer is not used. See "Timers" on page 2-35 for a complete description of timers.

TIMER_Ti

Explanation: Specifies the time period between 1 and 10 used to determine an inactive condition on a link. The time intervals are defined by the DIR.OPEN.ADAPTER command or by configuration parameters. If the value is zero, the current value remains in effect.

MAXOUT

Explanation: Specifies the maximum number of sequentially numbered, transmitted I-LPDUs that a link station associated with this SAP can have pending at any one time. The maximum valid value is 127. If the value is zero, the current value remains in effect.

MAXIN

Explanation: Specifies the maximum number of sequentially numbered, received I-LPDUs that a link station associated with this SAP can receive prior to sending an acknowledgment. The maximum valid value is 127. If the value is zero, the current value remains in effect.

MAXOUT_INCR

Explanation: This dynamic window increment value is used to reduce bridge congestion. If the two end points of the link are on different rings, and the adapter detects an error condition requiring retransmission, the MAXOUT parameter is set to 1. It is then incremented by 1 each time MAXOUT_INCR frames are acknowledged by the remote station, until it reaches the value requested by the application program. For more details, refer to the *IBM Token-Ring Network Architecture Reference*. If the value is zero, the current value remains in effect.

MAX_RETRY_CNT

Explanation: Specifies the number of retries for an unacknowledged command LPDU, or in the case of an I-LPDU timeout, the number of times that the non-responding remote link station will be polled with an RR/RNR command LPDU. This count is used in conjunction with the Response timer and should be great enough to ensure time for ring error detection and recovery. This parameter also prevents continual retransmission of the same I frame. The maximum valid value is 255. If the value is zero, the current value remains in effect.

ACCESS_PRIORITY

Explanation: The transmit access priority value to be placed in the AC byte of all transmissions from the SAP or link station. The format is B'nnn00000', where *nnn* is the access priority value. If the access priority is higher than that authorized for the adapter, the command terminates with a CCB_RETCODE of X'08'. Valid values for the access priority are from 0 to 3. Access priority is not recognized on the PC Network or on Ethernet.

GROUP_COUNT

Explanation: The number, from 0 to 13, of group SAPs as defined by the GROUP_LIST or GROUP_LIST_OFFSET field.

GROUP_LIST or **GROUP_LIST_OFFSET**

Explanation: This field can be used either to request membership in additional group SAPs for an individual SAP, or to request that membership be canceled. The GROUP_COUNT parameter indicates the number of valid values in this field. If the low-order bit of a SAP value is zero, membership in the corresponding group is requested. If the low-order bit of a SAP value is 1, membership is canceled.

This field is ignored if the GROUP_COUNT parameter is zero.

DLC.OPEN.SAP

Hex 15

DLC.OPEN.SAP

Command Description: This command activates a SAP and reserves a number of link stations for the SAP.

Command Specifics: This command can be used to define:

- An individual SAP
- A group SAP
- A SAP as a member of a group.

The application program is responsible for checking that the parameters are reasonable.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

If the return code is X'45' or X'49', the SAP has been opened, but there has been some problem with the GROUP_LIST or GROUP_LIST_OFFSET parameter.

An open for an individual SAP requesting group SAP membership for a group SAP can cause a return code of X'06' if the membership is requested in group SAPs with different types. See the bits described in the OPTIONS_PRIORITY field. If this happens, the SAP opens with membership into the group SAPs specified until the SAP type of the group SAP listed changes. All subsequent group SAPs listed after a SAP type change has been detected will not be used for membership.

Parameters:

Table 3-27 (Page 1 of 2). CCB Parameter Table for DLC.OPEN.SAP

Offset	Parameter Name	Byte Length	8086 TYPE	Description
0	STATION_ID	2	DW	SAP station ID (X'nn00').*
2	USER_STAT_VALUE	2	DW	User value passed back on DLC status.
4	TIMER_T1	1	DB	T1 value (response timer).
5	TIMER_T2	1	DB	T2 value (acknowledgment timer).
6	TIMER_TI	1	DB	Ti value (inactivity timer).
7	MAXOUT	1	DB	Maximum transmits without a receive acknowledgment.
8	MAXIN	1	DB	Maximum receives without a transmit acknowledgment.
9	MAXOUT_INCR	1	DB	Dynamic window increment value.
10	MAX_RETRY_CNT	1	DB	N2 value.
11	MAX_MEMBERS	1	DB	Maximum SAPs for a group SAP.
12	MAX_I_FIELD	2	DW	Maximum received information field.
14	SAP_VALUE	1	DB	SAP value to be assigned.
15	OPTIONS_PRIORITY	1	DB	SAP options and ring access priority. Access priority is not recognized on the PC Network or Ethernet.
16	STATION_COUNT	1	DB	Number of link stations to reserve.
17		2	DW	Reserved.
19	GROUP_COUNT	1	DB	Length of data in GROUP_LIST or GROUP_LIST_OFFSET.
For CCB1:				
20	GROUP_LIST	4	DD	Address of a list of group SAP values.
24	DLC_STATUS_EXIT	4	DD	I/O appendage exit, DLC status change.
For CCB2 and CCB3:				

* Indicates a returned value.

Table 3-27 (Page 2 of 2). CCB Parameter Table for DLC.OPEN.SAP

Offset	Parameter Name	Byte Length	8086 TYPE	Description
20	GROUP_LIST_OFFSET	2	DW	Offset to a list of group SAP values.
22		2	DW	Reserved for the application program. Can be used for segment or selector.
For CCB2:				
24	DLC_STATUS_FLAG	4	DD	User notification flag for DLC status changes.
For CCB3:				
24	DLC_STATUS_OFFSET	2	DW	Offset of DLC status appendage.
26		2	DW	Reserved for the application program. Can be used for segment or selector.
For all CCBs:				
28	DLC_BUF_SIZE	2	DW	Size of buffers in pool.
30	DLC_POOL_LEN	2	DW	Length of pool buffer.
32	DLC_POOL_ADDR	4	DD	Starting address of buffer pool.
For CCB2 and CCB3:				
36	ADAPTER_STNS_AVAIL	1	DB	Number of link stations available.*

* Indicates a returned value.

STATION_ID

Explanation: The station ID returned by the adapter or the protocol driver. This value is used to identify this SAP in subsequent commands.

USER_STAT_VALUE

Explanation:

For CCB1: On entry to the DLC status appendage (defined by parameter DLC_STATUS_EXIT), this value is passed back to the user in register SI.

For CCB2: User value, passed back on DLC status change notification using the READ command's CCB parameter table.

For CCB3: User value, passed back in register SI when calling application program device driver entry point for the DLC status change appendage.

TIMER_T1

Explanation: Specifies the time period between 1 and 10 used to determine an inoperative condition on a link. If the value is zero, the default of 5 is used. See "Timers" on page 2-35. The time intervals are defined by the DIR.OPEN.ADAPTER command or by configuration parameters.

TIMER_T2

Explanation: Specifies the time period between 1 and 10 used to delay transmission of an acknowledgment for a received I-LPDU for a link station in this SAP. If the value is zero, the default of 2 is used. If the value is greater than 10, the timer will not be used. See "Timers" on page 2-35. The time intervals are defined by the DIR.OPEN.ADAPTER command or by configuration parameters.

TIMER_Ti

Explanation: Specifies the time period between 1 and 10 used to determine an inactive condition on a link. If the value is zero, the default of 10 is used. See "Timers" on page 2-35. The time intervals are defined by the DIR.OPEN.ADAPTER command or by configuration parameters.

MAXOUT

Explanation: Specifies the maximum number of sequentially numbered, transmitted I-LPDUs that the link stations using this SAP can have pending at any time. The maximum valid value is 127. If the value is zero, the default of 2 is used.

MAXIN

Explanation: Specifies the maximum number of sequentially numbered, received I-LPDUs that the link station using this SAP can have before sending an acknowledgment. The maximum valid value is 127. If the value is zero, the default of 1 is used.

MAXOUT_INCR

Explanation: This dynamic window increment value is used to reduce bridge congestion. If the two end points of the link are on different rings, and the adapter detects an error condition requiring retransmission, the MAXOUT parameter is set to 1. It is then incremented by 1 each time MAXOUT_INCR frames are acknowledged by the remote station, until it reaches the value requested by the application program. For more details, refer to the *IBM Token-Ring Network Architecture Reference*. If the value is zero, the default of 1 is used.

MAX_RETRY_CNT

Explanation: Specifies the number of retries for an unacknowledged command LPDU, or in the case of an I-LPDU timeout, the number of times that the non-responding remote link station will be polled with an RR/RNR command LPDU. This count is used in conjunction with the Response timer and should be great enough to ensure time for ring error detection and recovery. This parameter also prevents continual retransmission of the same I frame. The maximum valid value is 255. If the value is zero, the default of 8 is used.

MAX_MEMBERS

Explanation: The maximum number of individual SAPs that can be assigned membership in the group SAP if this SAP is to be a group SAP as well as an individual SAP. Membership in the group SAP is assigned as the member SAPs are opened. This parameter cannot exceed the similar parameter provided with the DIR.OPEN.ADAPTER command or configuration parameters, and defaults to that value if this parameter is zero.

MAX_I_FIELD

Explanation: This parameter is the maximum size for an I frame that can be received by this SAP's link station. It applies to the information field in received I frames for link stations, and is ignored if STATION_COUNT is zero. If the value is zero, the default of 600 is used.

SAP_VALUE

Explanation: This is the value of the SAP to be assigned. The value must not be zero and the low-order bit is ignored. (In this example, "x" is the low order bit: B'nnnnnnnx'.) X'00' is invalid and X'FE' cannot be used as a group SAP.

This is the SSAP for transmitted messages and the DSAP for received messages. See Figure 1-3 on page 1-11 or Figure 1-7 on page 1-16 for information about Token-Ring Network, PC Network or Ethernet frame formats.

OPTIONS_PRIORITY

Explanation: Various SAP options, each represented by a bit. The bit being on (value of B'1') indicates taking the option. The high-order bit is the leftmost bit, 7. Following are the complete descriptions of the OPTIONS_PRIORITY bits:

Bits **Description**

7-5 Ring access priority. Access priority is not recognized on the PC Network or on Ethernet, and may not be implemented on an NDIS token-ring network adapter. When using an NDIS token-ring network adapter, see your adapter documentation for more information.

The transmit access priority for a token-ring network is placed in the AC byte of all transmissions from the SAP. If the access priority is too high, the command terminates with the CCB_RETCODE set to X'08'. This value is typically B'000'.

4 Reserved. It should be zero.

3 XID handling option.

- If this bit is zero, the XID command frames are handled for this SAP by the DLC function of the adapter.
- If this bit is 1, the XID command frames for this SAP are passed to the application program.

2 Individual SAP bit. If this bit is 1, the SAP is an individual SAP. The STATION_COUNT parameter must be zero if bit 2 is not 1.

1 Group SAP bit. If this bit is 1, the SAP is a group SAP.

0	Group SAP membership bit. If this bit is 1, the SAP may be a member of a group SAP. See the GROUP_COUNT and GROUP_LIST parameters.
	At least one of the bits 0, 1, and 2 must be on. Bit 0 can be on only if bit 2 is on.

STATION_COUNT

Explanation: The number of link stations to reserve. This parameter is to provide link station resources so that subsequent DLC.OPEN.STATION commands can be issued.

If the requested number of stations is not available, the command terminates with a CCB_RETCODE of X'46'.

If the value is zero, no station can be opened for the SAP.

GROUP_COUNT

Explanation: The number of group SAPs defined in the GROUP_LIST or GROUP_LIST_OFFSET field. If additional memberships are needed, use the DLC.MODIFY command. Valid values for the Token-Ring Network are from 0 to 8. Valid values for the PC Network and Ethernet are from 0 to 255.

GROUP_LIST or **GROUP_LIST_OFFSET**

Explanation: This field points to a list of group SAP values. The GROUP_COUNT parameter indicates the number of valid values in this field. This field is ignored if the GROUP_COUNT parameter is zero.

For CCB1: The GROUP_LIST is the address of the list.

For CCB2 and CCB3: The GROUP_LIST_OFFSET is an offset to the list.

DLC_STATUS_EXIT

Explanation: This field points to the beginning of an appendage routine provided by the application program. This routine receives control whenever DLC status changes for this SAP. See "DLC Status Codes" on page B-28.

DLC_STATUS_FLAG

Explanation: This flag must be set to a non-zero value if notification is wanted when any DLC status changes for the specified SAP and associated link stations. See "DLC Status Codes" on page B-28.

DLC_STATUS_OFFSET

Explanation: This offset identifies the DLC status appendage so that the application program can be notified of any changes for the specified SAP and associated link stations. This appendage offset is passed to the application program device driver entry point in register DI when the DD interface calls the application program for notification of any DLC status change. See "DLC Status Codes" on page B-28.

DLC_BUF_SIZE

Explanation: The size of the buffers in the SAP buffer pool. This is the size of the entire buffer including all adapter overhead. The size must be a multiple of 16 with a minimum size of X'80'. If this value is zero, the default of X'160' is used. This field must be on a 16-byte boundary of the form X'xxx0'.

For CCB2 and CCB3: The maximum DLC_BUF_SIZE for OS/2 is 4000.

DLC_POOL_LEN

Explanation: The number of 16-byte blocks in the SAP buffer pool, which is the size of the SAP buffer pool divided by 16.

If this value is zero, the default of 256 (4096 bytes) is used. If the DLC_POOL_ADDR is zero, this parameter is ignored.

For CCB2 and CCB3: The maximum DLC_POOL_LEN for OS/2 is 4000.

DLC_POOL_ADDR

Explanation: The starting address in workstation memory where the adapter is to build the SAP buffer pool. If this value is zero, the application program has the responsibility of building the buffer pool, and the DLC_BUF_SIZE parameter must indicate the size of each buffer.

ADAPTER_STNS_AVAIL

Explanation: If the command terminates with a return code of X'46', this field contains the number of link stations currently available that have not been reserved. This SAP is not opened and no link stations are reserved for this SAP.

DLC.OPEN.STATION

Hex 19

DLC.OPEN.STATION

Command Description: This command allocates resources for a link station.

Command Specifics: The protocol driver performs functions to set up the link station in the adapter, but no ring communication takes place. A DLC.CONNECT.STATION command must be issued to either the local or remote link station by its application program to initiate connection-oriented communications. Thereafter, a DLC.CONNECT.STATION command must be issued at the other station to complete establishing the link.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-28. CCB Parameter Table for DLC.OPEN.STATION

Offset	Parameter Name	Byte Length	8086 Type	Description
0	SAP_STATION_ID	2	DW	SAP station ID.
2	LINK_STATION_ID	2	DW	Link station ID (X'nnss'). *
4	TIMER_T1	1	DB	T1 value (response timer).
5	TIMER_T2	1	DB	T2 value (acknowledgment timer).
6	TIMER_TI	1	DB	Ti value (inactivity timer).
7	MAXOUT	1	DB	Maximum transmits without a receive acknowledgment.
8	MAXIN	1	DB	Maximum receives without a transmit acknowledgment.
9	MAXOUT_INCR	1	DB	Dynamic window increment value.
10	MAX_RETRY_CNT	1	DB	N2 value.
11	RSAP_VALUE	1	DB	The remote SAP value.
12	MAX_I_FIELD	2	DW	Maximum received information field.
14	ACCESS_PRIORITY	1	DB	Ring access priority. Access priority is not recognized on the PC network or on Ethernet.
15		1	DB	Reserved
For CCB1:				
16	DESTINATION	4	DD	Pointer to remote station address.
For CCB2 and CCB3:				
16	DESTINATION_OFFSET	2	DW	Offset to remote station address.
18		2	DW	Reserved for the application program. Can be used for segment or selector.

* Indicates a returned value.

SAP_STATION_ID

Explanation: The SAP ID value passed to the adapter.

LINK_STATION_ID

Explanation: The link station ID value assigned by the adapter.

TIMER_T1

Explanation: Specifies the time period between 1 and 10 used to determine an inoperative condition on a link. If the value is zero, the default is as defined by the DLC.OPEN.SAP command. See "Timers" on page 2-35. The time intervals are defined by the DIR.OPEN.ADAPTER command or by the configuration parameters.

TIMER_T2

Explanation: Specifies the time period between 1 and 10 used to delay transmission of an acknowledgment for a received I-LPDU for this SAP. If the value is zero, the default is as defined by the DLC.OPEN.SAP. If the value is greater than 10, the timer will not be used. See "Timers" on page 2-35. The time intervals are defined by the DIR.OPEN.ADAPTER command or by the configuration parameters.

TIMER_Ti

Explanation: Specifies the time period between 1 and 10 used to determine an inactive condition on a link. If the value is zero, the default as defined by the DLC.OPEN.SAP command is used. See "Timers" on page 2-35. The time intervals are defined by the DIR.OPEN.ADAPTER command or by the configuration parameters.

MAXOUT

Explanation: Specifies the maximum number of sequentially numbered, transmitted I-LPDUs that the link station can have pending at any time. The maximum valid value is 127. If the value is zero, the default as defined by the DLC.OPEN.SAP command is used.

MAXIN

Explanation: Specifies the maximum number of sequentially numbered, received I-LPDUs that the link station can have before sending an acknowledgment. The maximum valid value is 127. If the value is zero, the default as defined by the DLC.OPEN.SAP command is used.

MAXOUT_INCR

Explanation: This dynamic window increment value is used to reduce bridge congestion. If the two end points of the link are on different rings, and an error condition requiring retransmission is detected, the MAXOUT counter is set to 1. It is then incremented by 1 each time MAXOUT_INCR frames are acknowledged by the remote station, until it reaches the value requested by the application program in the MAXOUT parameter. For more details refer to the *IBM Token-Ring Network Architecture Reference*. If the value is zero, the default as defined by the DLC.OPEN.SAP command is used.

MAX_RETRY_CNT

Explanation: Specifies the number of retries for an unacknowledged command LPDU, or in the case of an I-LPDU timeout, the number of times that the non-responding remote link station will be polled with an RR/RNR command LPDU. The maximum valid value is 255. If the value is zero, the default as defined by the DLC.OPEN.SAP command is used.

RSAP_VALUE

Explanation: This is the value of the remote SAP partner. It must follow the same guidelines as the SAP_VALUE parameter of the DLC.OPEN.SAP command, and it must be an individual SAP. A group SAP cannot have a link station.

MAX_I_FIELD

Explanation: This parameter applies to the information field in received I frames for this link station. If the value is zero, the default is as defined by the DLC.OPEN.SAP command.

ACCESS_PRIORITY

Explanation: Specifies the transmit access priority value to be placed in the AC byte of all transmissions from the link station. The format is B'nnn00000', where 'nnn' is the access priority value. If the access priority is higher than that authorized for the adapter, the command terminates with a CCB_RETCODE of X'08'. Valid values for the access priority are from 0 to 3. Access priority is not recognized on the PC network or on Ethernet.

DESTINATION

Explanation: This field points to a 6-byte destination NODE_ADDRESS parameter of the remote adapter. The high-order bit of the NODE_ADDRESS must be zero. Any routing information required for the frame header is supplied when the DLC.CONNECT.STATION command is issued.

The value must not be all ones. The 2 high-order (leftmost) bits must be B'01'. For other restrictions and details about addresses, refer to the *IBM Token-Ring Network Architecture Reference*.

DESTINATION_OFFSET

Explanation: This field is the offset to the 6-byte destination NODE_ADDRESS of the remote adapter. The high-order bit of the NODE_ADDRESS must be zero. Any routing information required for the frame header is supplied when the DLC.CONNECT.STATION command is issued.

The value must not be all ones. The 2 high-order (leftmost) bits must be B'01'. For other restrictions and details about addresses, refer to the *IBM Token-Ring Network Architecture Reference*.

DLC.REALLOCATE

Hex 17

DLC.REALLOCATE

Command Description: This command adjusts (reallocates) the number of link stations allocated to a SAP without closing and reopening the SAP.

Command Specifics: If this command is used to reduce the number of allocated link stations, then these deallocated link stations are returned to the adapter's available pool.

If this command increases the number of allocated link stations, the link stations are assigned from the adapter's available pool.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-29. CCB Parameter Table for DLC.REALLOCATE

Offset	Parameter Name	Byte Length	8086 Type	Description
0	STATION_ID	2	DW	ID of affected SAP
2	OPTION_BYTE	1	DB	Increase or decrease indicator
3	STATION_COUNT	1	DB	Number of link stations to be reallocated
4	ADAPTER_AVAILABLE_STNS	1	DB	Number of link stations available for this adapter *
5	SAP_AVAILABLE_STNS	1	DB	Number of link stations available for this SAP *
For CCB2 and CCB3:				
6	ADAPTER_STATION_COUNT	1	DB	Number of link stations configured for this adapter *
7	SAP_STATION_COUNT	1	DB	Number of link stations reserved for this SAP *

* Indicates a returned value.

STATION_ID

Explanation: The SAP station ID affected (X'nn00').

OPTION_BYTE

Explanation: The high-order bit (bit 7) indicates whether the command increases or decreases the number of link stations. If bit 7 is 0, it increases the number of link stations. If bit 7 is 1, it decreases the number of link stations. Bits 6 through 0 are reserved.

STATION_COUNT

Explanation: This parameter is the number of link stations to be reallocated as indicated by the OPTION_BYTE parameter. If this number is greater than the available link stations (either to increase or decrease), those that are available are reallocated, and the command terminates with a return code of X'57'. If this value is zero, the command is executed and returns the indicated values in the parameter table.

ADAPTER_AVAILABLE_STNS

Explanation: Number of available link stations (not allocated to a SAP) for this adapter (after this command is completed). This value is only valid for the following return codes: X'00', X'40', X'57'.

SAP_AVAILABLE_STNS

Explanation: Number of available link stations allocated to this SAP that are not currently in use (after this command is completed). This value is only valid for the following return codes: X'00', X'57'.

ADAPTER_STATION_COUNT

Explanation: Number of link stations configured for this adapter when it was initialized. This value is only valid for the following return codes: X'00', X'40', X'57'.

SAP_STATION_COUNT

Explanation: Number of stations currently reserved for this SAP (after this command is completed). This value is only valid for the following return codes: X'00', X'57'.

DLC.RESET

Hex 14

DLC.RESET

Command Description: This command resets the following SAPs and link stations:

One SAP and all associated link stations
All SAPs and all associated link stations
For CCB2 and CCB3: All SAPs and all associated link stations for one application program.

Command Specifics: This command should be used in situations where some activity on the workstation is being shut down. The affected stations will be closed. The command will not be executed until all affected resources are freed. TRANSMIT commands already queued for transmission will be executed. If this command is issued while the ring is beaconing, it cannot be completed until the ring is no longer beaconing. Refer to the *IBM Token-Ring Network Architecture Reference* for a description of ring beaconing.

All pending commands will be terminated for the SAPs and stations. All communication will cease, and the SAPs and stations resources will be released. They must be reopened to be used further.

For CCB1: The CCB_PARM_TAB field contains the STATION_ID value in the 2 high-order bytes. A STATION_ID value of X'0000' defines all SAPs and all link stations. A STATION_ID value of X'nn00' defines SAP 'nn' and all its link stations. The 2 remaining bytes are reserved.

When this command is completed, all pending commands that were terminated can be located using the CCB_POINTER.

For CCB2: The CCB_PARM_OFFSET field contains the STATION_ID value. A STATION_ID value of X'0000' defines all SAPs and all link stations. A STATION_ID value of X'nn00' defines SAP 'nn' and all its link stations.

A system administrator can issue this command using the System Key as defined by the configuration parameters. The System Key should be placed in the CCB_PARAMETER_2 field, if defined.

If it is necessary for an application program to have pointers to the following returned, then the CCB_CMPL_FLAG of the DLC.RESET command must be set:

- Pending commands
- List of buffers remaining in the SAP buffer pools
- The pending receive frames.

When a READ command is issued requesting command completions that post the completed DLC.RESET command, the pointers are returned in the READ command's parameter table. When this command is completed, all pending commands that were terminated are queued on the CCB_POINTER of the DLC.RESET command.

When a system administrator uses the System Key with this command, only the SAP buffer pool buffers and pending receive frames for SAPs opened by the system administrator are returned in the READ command's parameter table. In addition, only pending commands issued by the system administrator are returned to the system administrator with the CCB_POINTER of the DLC.RESET command.

Application programs affected by a DLC.RESET command issued with a System Key must have a READ command pending that requests notification of a system action to be notified of the event. See "System Action Exceptions for OS/2 EE 1.3" on page B-69 for more information.

For CCB3: The CCB_PARM_OFFSET field contains the STATION_ID value. A STATION_ID value of X'0000' defines all SAPs and all link stations. A STATION_ID value of X'nn00' defines SAP 'nn' and all its link stations.

A system administrator can issue this command using the System Key as defined by the configuration parameters. The System Key should be placed in the CCB_PARAMETER_2 field, if defined.

If it is necessary for an application program to have buffers from the SAP buffer pools returned, the CCB_CMPL_APPNDG of the DLC.RESET command must be set. When this command is executed, all pending commands that were terminated are queued on the CCB_POINTER of the DLC.RESET command.

When a system administrator uses the System Key with this command, only the buffers from buffer pools of SAPs opened by the system administrator are returned when the application program is called for command completion posting.

Application programs affected by a DLC.RESET issued with the System Key can be notified of the event. See "System Action Exceptions for OS/2 EE 1.3" on page B-69 for more information.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

DLC.SET.THRESHOLD

Hex 33

DLC.SET.THRESHOLD

Command Description: This command is for **CCB2 only**. It sets a threshold for the SAP or direct station's buffer pool. Whenever the number of buffers in a SAP buffer pool falls below the specified threshold, the application program is notified.

Command Specifics: This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-30. CCB Parameter Table for DLC.SET.THRESHOLD Command and Paramaters are for CCB2 only

Offset	Parameter Name	Byte Length	8086 Type	Description
For CCB2 only:				
0	STATION_ID	2	DW	SAP station ID
2	BUFFER_THRESHOLD	2	DW	SAP buffer pool threshold number
4	ALERT_SEMAPHORE	4	DD	Alert semaphore

STATION_ID

Explanation: The station ID of the SAP or direct station's buffer pool.

Use the SAP station ID to identify the buffer pool for which the threshold is being set.

BUFFER_THRESHOLD

Explanation: Number of buffers that define the threshold. If the number of buffers in the SAP buffer pool is less than this threshold, the ALERT_SEMAPHORE parameter is cleared to notify the application program using the buffers. If a buffer is removed from the pool and the pool contains fewer buffers than the threshold, the ALERT_SEMAPHORE parameter is cleared.

ALERT_SEMAPHORE

Explanation: A system semaphore can be used to alert the application program of the threshold exceeded condition. When the condition exists, the protocol driver clears the ALERT_SEMAPHORE parameter to notify the application program.

To specify a system semaphore, the application program places the semaphore handle into the ALERT_SEMAPHORE field. The semaphore handle is returned from OS/2 EE when the system semaphore is created or opened.

DLC.STATISTICS

Hex 1E

DLC.STATISTICS

Command Description: This command reads and optionally resets the DLC logs specified. The log is transferred to the buffer indicated by the parameter table.

Command Specifics: For information about the adapter and direct interface logs, see the DIR.READ.LOG command.

If the STATION.ID indicates a SAP and no link station data, this command is executed totally in the workstation. The return code is available to the application program upon return from the protocol driver.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-31. CCB Parameter Table for DLC.STATISTICS

Offset	Parameter Name	Byte Length	8086 Type	Description
0	SAP_STATION_ID	2	DW	Identify the log to read
2	LOG_BUF_LENGTH	2	DW	Size of buffer at LOG_BUFFER_ADDR and LOG_BUFFER_OFFSET
For CCB1:				
4	LOG_BUFFER_ADDR	4	DD	Address to place the log data
For CCB2 and CCB3:				
4	LOG_BUFFER_OFFSET	2	DW	Offset of buffer for log data
6		2	DW	Reserved. Can be used for segment or selector.
For all CCBs:				
8	LOG_ACT_LENGTH	2	DW	Actual length of log *
10	OPTIONS	1	DB	Command options

* Indicates a returned value.

STATION_ID

Explanation: The station ID of the SAP or link station that the statistics are read from or optionally reset.

LOG_BUF_LENGTH

Explanation: The length of the buffer defined by LOG_BUFFER_ADDR or LOG_BUFFER_OFFSET.

LOG_BUFFER_ADDR

Explanation: The address of the buffer where the log data is to be placed.

LOG_BUFFER_OFFSET

Explanation: The offset of the buffer where the log data is to be placed.

LOG_ACT_LENGTH

Explanation: The returned value of the actual length of the log that was requested. If this value is greater than the value of LOG_BUF_LENGTH, not all of the log has been transferred. CCB_RETCODE will contain X'15'.

Note: On a "lost data" condition, the log is cleared if the options indicate reset.

OPTIONS

Explanation: Command options. Following are the complete descriptions of the OPTIONS bits:

Bits	**Description**
7	If this bit is on, the counters are reset to zero where appropriate.
0-6	These bits are reserved; they should be zero, but are not checked.

Log Formats

There are two log formats.

SAP log (maintained in workstation memory)

If the DLC.STATISTICS command request is for a SAP log (X'nn00'), or a direct station log (X'00ss') for CCB1, the format of the data placed in the buffer is shown in Table 3-32. The counters are reset after they are read.

Table 3-32. Log Formats for the SAP Log

Bytes	**8086 Type**	**Meaning**
0-3	DD	Number of frames transmitted
4-7	DD	Number of frames received
8-11	DD	Number of frames discarded (no RECEIVE command)
12-15	DD	Number of times data was lost. Can be caused by inadequate SAP buffers.
16-17	DW	Number of buffers available in the SAP buffer pool

Link Station log (maintained in the adapter)

If the DLC.STATISTICS command request is for a link station log (X'nnss'), the format of the data placed in the buffer is shown in Table 3-33.

Table 3-33. Log Formats for the Link Station Log

Bytes	8086 Type	Meaning
0-1	DW	Number of I frames transmitted
2-3	DW	Number of I frames received
4	DB	Number of I frame receive errors
5	DB	Number of I frame transmission errors
6-7	DW	Number of times T1 expired (other than in data-transfer mode)
8	DB	Last command/response received
9	DB	Last command/response sent
10	DB	Link primary state
11	DB	Link secondary state
12	DB	Send state variable
13	DB	Receive state variable
14	DB	Last received NR
15	DB	Length of network header used in station transmissions
16-47	DB	Network header used in station transmissions

Notes:

1. See "Link Station States" on page 2-33 for more about bytes 10 and 11.
2. For the SAP counters that are 4 bytes long, an overflow condition is not reported. An overflow indication is given when the counters reach half their maximum value (X'80' or X'8000') by a DLC status change notification.
3. Some parameters are not counters, and these parameters are not reset when the reset option is selected.

PDT.TRACE.OFF

Hex 25

PDT.TRACE.OFF

Command Description: This command is for **CCB1 only**. It terminates the PDT.TRACE.ON command.

Command Specifics: The results of the PDT.TRACE.OFF command (when successful) are:

- The PDT.TRACE.ON command's CCB_RETCODE field is set to X'0A'.
- The PDT.TRACE.OFF command's CCB_RETCODE field is set to X'00'.
- The PDT.TRACE.OFF command's CCB_POINTER field contains the address of the PDT.TRACE.ON CCB that was terminated.

- The CCB_CMD_CMPL completion exit of the PDT.TRACE.OFF command, if defined, is taken.
- The CCB_CMD_CMPL completion exit of the PDT.TRACE.ON command is ignored.

If a PDT.TRACE.ON command is not pending:

- The PDT.TRACE.OFF command's CCB_RETCODE field is set to X'00'.
- The PDT.TRACE.OFF command's CCB_POINTER field contains 0.
- The CCB_CMD_CMPL completion exit of the PDT.TRACE.OFF command, if defined, is taken.

This command is executed entirely by the protocol driver in the workstation. Therefore, the command completion appendage is not required, as the command is complete upon return. However, if the command completion appendage is provided, it will be used as defined in this section.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

PDT.TRACE.ON

Hex 24

PDT.TRACE.ON

Command Description: This command is for **CCB1 only**. It provides an interrupt trace for all adapter traffic.

For OS/2 trace information, see Appendix E, "Operating System/2 Extended Edition Information."

Command Specifics: The command provides entries for the following activities:

- Each CCB when it is issued to the adapter if the initial return code is X'FF'.
- Each CCB completion.
- Each NCB when issued by the application program (return code = X'FF').
- All adapter interrupts to the workstation. If the interrupt is a timer interrupt only, a trace entry is not made, but the timer interrupts are counted for reporting. Then when a non-timer interrupt occurs, a timer trace entry is made containing the accumulated timer interrupts followed by a trace entry for the non-timer interrupt.

Only one trace command can be active. The trace includes all activity for either the primary or alternate adapter, or both. The command is terminated by either a PDT.TRACE.OFF command or an exception, when issued.

This command is executed entirely by the protocol driver in the workstation. Therefore, the command completion appendage is not required, as the command is complete upon return. However, the command completion appendage will be taken, if provided.

The CCB_ADAPTER field of the CCB can be any value between X'00' and X'03'. The values X'00' or X'02' identify CCB adapter 0; the values X'01' or X'03' identify CCB adapter 1.

The location of the trace table is pointed to by the value placed in the CCB_PARM_TAB field by the application program. The trace wraps when the buffer fills.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-34. CCB Parameter Table for PDT.TRACE.ON

Offset	Parameter Name	Byte Length	8086 Type	Description
For CCB1 only:				
0	TABLE_LENGTH	2	DW	Length of the TRACE_TABLE
2	CURRENT_OFF	2	DW	Offset of the current trace entry *
4	START_TICK_0	4	DD	Adapter 0 timer tick count, trace start *
8	STOP_TICK_0	4	DD	Adapter 0 timer tick count, trace stop *
12	START_TICK_1	4	DD	Adapter 1 timer tick count, trace start *
16	STOP_TICK_1	4	DD	Adapter 1 timer tick count, trace stop *
20		12	--	Adapter work area
32		--	--	Trace table

* Indicates a returned value.

TABLE_LENGTH

Explanation: The length of the requested trace table with a minimum value of 256. The entries are 16 bytes long. If the length specified is not a multiple of 16, the last 1 to 15 bytes are unused.

CURRENT_OFF

Explanation: The offset from the TRACE_TABLE value of the most recent table entry.

The table wraps around when full. If the first entry in the table is the PDT.TRACE.ON command, the table has not wrapped.

START_TICK_0

Explanation: The value of the adapter 0 timer tick counter, as set by the adapter, when the trace started.

Adapter ticks occur every 100 ms.

STOP_TICK_0

Explanation: The value of the adapter 0 timer tick counter, as set by the adapter, when the trace stopped.

START_TICK_1

Explanation: The value of the adapter 1 timer tick counter, as set by the adapter, when the trace started.

STOP_TICK_1

Explanation: The value of the adapter 1 timer tick counter, as set by the adapter, when the trace stopped.

TRACE_TABLE

Explanation: The trace table starts here. The length is defined by the TABLE_LENGTH field. The trace table formats for the non-NDIS Token-Ring Network and the PC Network are explained in the following section.

Trace Table Formats for the Non-NDIS Token-Ring Network Adapters

Four trace entry formats are used, and each trace entry is 16 bytes long.

The SS and SP registers point to 26 bytes of stack space used by the protocol driver when the trace entry is made.

Table 3-35. CCB Trace Entry

Byte	Meaning
0	Adapter number (0/1)
1	Flags
2	CCB command
3	Return code
4-7	SS:SP registers
8-11	Address of the interrupted application program code
12-15	ES:BX registers

The command code (byte 2) is zero if not applicable. If this entry is the result of a request for data by the adapter following a TRANSMIT CCB, the specific TRANSMIT command code is inserted.

Table 3-36. Byte 1: Flags

Bit	Meaning
7	Adapter initialized
6	Initialize in process
5	Adapter opened
4	Open in process
3	SRB busy
2	Block bit on
1	Always 0
0	No adapter

The ES and BX registers point to the CCB.

Table 3-37. Adapter Interrupt Trace Entry (Except Timer)

Byte	Token-Ring Network
0	ISRP Even
1	ISRP Odd
2	Command code of the interrupt
3	Return code
4-7	SS:SP registers
8-11	Address of the interrupted application program code
12-15	ES:BX registers

Table 3-38. Byte 0: Interrupt Status Register Processor (ISRP) Even

Bit	Meaning
7	CHCK/IRQ steering control (always 1)
6	Interrupt enabled (always 1)
5	Reserved
4	Timer Interrupt (100-ms programmable timer)
3	Error Interrupt
2	Access Interrupt
1	Always on
0	Adapter number (0 or 1)

Table 3-39. Byte 1: Interrupt Status Register Processor (ISRP) Odd

Bit	Meaning
7	Reserved
6	Adapter check
5	SRB response
4	ASB free
3	ARB command
2	SSB response
1	Reserved
0	Reserved

The ES and BX registers point to the applicable CCB or buffer if the interrupt is a result of a CCB. Otherwise these bytes will be zero.

Table 3-40 (Page 1 of 2). Adapter Timer Interrupt Trace Entry

Byte	Meaning
0	ISRP Even, adapter 0=X'D2' and adapter 1=X'D3'
1	X'00'
2-3	2-byte counter: number of timer interrupts, adapter 0 and 1
4-7	SS:SP registers

Table 3-40 (Page 2 of 2). Adapter Timer Interrupt Trace Entry

Byte	Meaning
8-11	Address of the interrupted application program code
12-15	ES:BX registers

When no other interrupts are occurring, timer interrupts are maintained for reporting. The counter is updated for each timer interrupt (either adapter). The accumulated interrupts are placed in a timer interrupt trace entry when a non-timer interrupt occurs, producing another trace entry. The ES and BX registers point to the DIR.TIMER.SET CCB if this timer interrupt causes a DIR.TIMER.SET command to be completed. Otherwise, this field is zero.

Table 3-41. NCB Trace Entry

Byte	Meaning
0	X'0F' X'1F' X'2F' (see below)
1	Adapter number (0/1)
2-3	NCB command and return code
4-7	SS:SP registers
8-11	Address of the interrupted application program code
12-15	ES:BX registers

The ES and BX registers point to the NCB.

When a post routine is used while trace is active, three entries are made in the trace table.

Byte 0 contains:

- X'0F' for the entry when the NCB is first issued
- X'1F' when going to the user-supplied post routine
- X'2F' when returning from the post routine.

Trace Table Formats for the PC Network and NDIS Adapters

Three trace entry formats are used. Each trace entry is 16 bytes long.

The SS and SP registers point to 26 bytes of stack space used by the protocol driver when the trace entry is made.

Table 3-42. CCB Trace Entry

Byte	Meaning
0	Adapter number (0/1)
1	Flags
2	CCB command
3	Return code
4-7	SS:SP registers
8-11	Address of the interrupted application program code
12-15	ES:BX registers

The command code (byte 2) is zero if not applicable. If this entry is the result of a request for data by the adapter following a TRANSMIT CCB, the specific TRANSMIT command code is inserted.

Table 3-43. Byte 1: Flags

Bit	Meaning
7	Adapter initialized
6	Initialize in process
5	Adapter opened
4	Open in process
3	Command request block busy
2	Reserved
1	Reserved
0	No adapter

The ES and BX registers point to the CCB.

The interrupt trace entry structure will be described in general followed by each possible entry described separately.

Table 3-44. Interrupt Trace Entry

Byte	PC Network
0	Adapter number (0/1) and activation reason code
1	Command code or X'00'
2	Command dependent information
3	Command dependent information
4-7	SS:SP registers
8-11	Address of the interrupted application program code
12-15	ES:BX registers

The ES and BX registers point to the applicable CCB or buffer if the interrupt is a result of a CCB. Otherwise these bytes are zero.

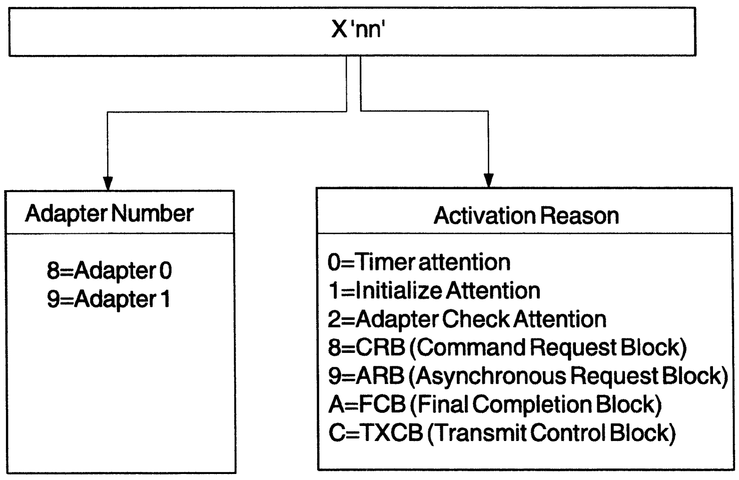

X'nn'
Adapter Number
8=Adapter 0
9=Adapter 1
Activation Reason
0=Timer attention
1=Initialize Attention
2=Adapter Check Attention
8=CRB (Command Request Block)
9=ARB (Asynchronous Request Block)
A=FCB (Final Completion Block)
C=TXCB (Transmit Control Block)

Figure 3-1. Byte 0: Adapter Number (0/1) and Activation Reason Code

Figure 3-2 shows the structure for the Timer Entry, when byte 0 is X'80' or X'90'.

0	1	2-3	4-7	8-11	12-15
X'80' & X'90'	X'00'	Tick counter: Adapter 0 and 1	SS:SP	Address of interrupted code	ES:BX

Figure 3-2. Timer Entry

Note: Timer interrupts with no other interrupts occurring are maintained in one trace entry. The counter is updated for each timer interrupt (either adapter) and the other values in the trace represent the latest trace entry.

Figure 3-3 shows the structure for the Initialize Entry, when byte 0 is X'81' or X'91'.

0	1	2-3	4-7	8-11	12-15
X'81' & X'91'	X'00'	X'0000'	SS:SP	Address of interrupted code	ES:BX

Figure 3-3. Initialize Entry

Figure 3-4 shows the structure for the Command Request Block (CRB) Entry, when byte 0 is X'88' or X'98'.

0	1	2	3	4-7	8-11	12-15
X'88' & X'98'	CRB Command Code	Return Code	Correlator (00 if RC= X'FF')	SS:SP	Address of interrupted code	ES:BX

Figure 3-4. CRB Entry

Figure 3-5 shows the structure for the Asynchronous Request Block (ARB) Entry, when byte 0 is X'89' or X'99'.

0	1	2	3	4-7	8-11	12-15
X'89' & X'99'	ARB Command Code	(see below)		SS:SP	Address of interrupted code	ES:BX

	1	2	3
Receive	X'81'	Station ID	
Transmit	X'82'	Sta Num	Corr
DLC Status	X'83'	DLC Status	
Network Status	X'84'	Network Status	

Figure 3-5. ARB Entry

The contents of the ES:BX registers are as follows:

- For a receive, the ES:BX registers point to the receive CCB if the receive is completing (no RECEIVE.DATA appendage or a bad return code), else it points to the first receive buffer if a RECEIVE.DATA appendage is specified.
- For a transmit, the ES:BX registers point to the transmit CCB.
- For a DLC status, the ES:BX registers point to the DLC status buffer
- For a network status, the ES:BX registers point to a queue of pending CCBs if the adapter has been closed, or it is zero.

Figure 3-6 shows the structure for the Final Completion Block (FCB) Entry, when byte 0 is X'8A' or X'9A'.

0	1	2	3	4-7	8-11	12-15
X'8A' & X'9A'	FCB Command Code	Return Code	Correlator	SS:SP	Address of interrupted code	ES:BX

Figure 3-6. FCB Entry

Figure 3-7 shows the structure for the Transmit Control Block (TXCB) Entry, when byte 0 is X'8C' or X'9C'.

0	1	2	3	4-7	8-11	12-15
X'8C' & X'9C'	TXCB Command Code	Return Code	X'00'	SS:SP	Interrupted	ES:BX

Figure 3-7. TXCB Entry

Table 3-45. NCB Trace Entry

Byte	Meaning
0	X'0F' X'1F' X'2F' X'3F' (see below)
1	Adapter number (0/1)
2-3	NCB command and return code
4-7	SS:SP registers
8-11	Address of the interrupted application program code
12-15	ES:BX registers

The ES and BX registers point to the NCB.

If there is no NCB post routine specified, there will be no X'2F' entry.

Byte 0 contains:

- X'0F' for the entry when the NCB is first issued
- X'1F' when posting the NCB
- X'2F' when returning from a post routine
- X'3F' when NETBIOS indicates finished with NCB.

PURGE.RESOURCES

Hex 36

PURGE.RESOURCES

Command Description: This command is for **CCB3 only**. It allows the application program to purge resources from the protocol driver that are owned by a terminating OS/2 process. This command can also be used to purge resources from the protocol driver that are allocated from memory being returned to the operating system.

The application program must use this command if all the following are true:

- The application program is using the DD interface.
- Control blocks that were allocated from memory owned by one or more OS/2 processes have been passed to the protocol driver.
- After one of its processes terminates, the application program must continue using the services of the protocol driver. In this case the application program should not issue the DIR.CLOSE.ADAPTER command.
- It is necessary for the application program to free memory to the operating system, but it still has CCBs and buffers pending with the protocol driver. The application program should not issue the DIR.CLOSE.ADAPTER command.

Command Specifics:

- Memory passed to a protocol driver (LANDD) in the form of CCBs or buffers can be owned by different processes of an application program or by an application program's device driver. Since the LAN device drivers cannot guarantee that the active process is the owner of memory being passed to it (command invocations at interrupt time) all control blocks and buffers are associated with a resource ID. The resource ID is passed as a parameter with the CCB and is associated with the CCB and all other control blocks (for example, logs and buffers) referenced by the CCB.

In order to have CCBs that have been purged returned to the application program, this command must be used specifying a command completion appendage.

You may want an application program to purge resources when it consists of more than one OS/2 process and each process has resources. For example, an application program has two OS/2 processes that have both been allocated memory (being used in the application program's SAP buffer pool). When one of the processes terminates, the application program should use the PURGE.RESOURCES command to specify to the protocol driver what control blocks should be removed (cleaned up) from the protocol driver's internal queues.

- To identify the resource ID (PURGE_RESOURCE_ID) of the control blocks to be purged, use the CCB_PARAMETER_2 field of the PURGE.RESOURCES CCB.

  All control blocks that have been passed to the protocol driver with a resource ID that matches PURGE_RESOURCE_ID are removed from the protocol driver internal queues and returned to the application program. All pending CCBs that are canceled are chained using the CCB_POINTER fields with the CCB_RETCODE field set to X'67'. When the protocol driver calls the application program's device driver (with the appropriate event appendage offset passed in register DI) and there are no resources associated with the PURGE_RESOURCE_ID, the 12-byte information table referenced by registers DX and DI will contain zeros.
- If the application program does not purge a terminating process' resources immediately from the LAN device driver's internal queues, other application programs can be adversely affected. If the memory owned by the terminating process is reallocated by OS/2 to another process, the memory can be written into by the protocol driver if the terminating process's control blocks are not removed from the protocol driver's internal queues.
- This command does not close stations, SAPs, or application programs. It only cleans up data areas.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

READ

Hex 31

READ

Command Description: This command is for **CCB2 only**. It performs a read function for the direct and DLC interfaces. The station ID is used to define the interface that the read is being issued for.

Command Specifics: When a READ command is issued, it is queued to wait for the specified read event for the station ID indicated in the command. Multiple READ commands can be pending at one time for each station ID. Each READ command specifies an event or set of events and a system semaphore handle (CCB_SEMAPHORE). If an event has already occurred that matches an event specified with a READ command, or at some later time an event occurs that matches an event specified with a READ command, the following takes place:

- Data associated with the event is copied into the READ command's CCB parameter table.

- CCB_RETCODE of the READ command CCB is set to X'00'.
- CCB_SEMAPHORE, if provided, is cleared signaling that the READ command has been completed.

Once the READ command's CCB_RETCODE has been set and CCB_SEMAPHORE cleared, the READ command has been completed and must be issued again to become active.

In some cases, it may be necessary for an application program to guarantee that a READ command is pending for a particular CCB command completion. For example, when a DIR.CLOSE.ADAPTER is executed, all the application program's pending CCBs, SAP buffers, and receive frames are returned in the parameter table of a pending READ command. If no READ command is pending, the application program's data areas are not returned. To guarantee that a READ command is pending, the application program can chain a READ command's CCB to a CCB by placing the address of the READ command's CCB in the CCB_POINTER field and setting the CCB_READ_FLAG field to a non-zero value. When the CCB is completed, the READ command referenced by the CCB_POINTER field is used to post the completion of the command. In addition to updating the READ command's CCB parameter table, the READ command's CCB_RETCODE is set and CCB_SEMAPHORE is cleared (if it exists).

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-46 (Page 1 of 2). CCB Parameter Table for READ

Offset	Parameter Name	Byte Length	8086 Type	Description
For CCB2 only:				
0	STATION_ID	2	DW	SAP station ID
2	OPTION_INDICATOR	1	DB	READ option indicator
3	EVENT_SET	1	DB	Set of events for notification
4	EVENT	1	DB	Posting event *
5	CRITICAL_SUBSET	1	DB	Critical event subset identifier *
6	NOTIFICATION_FLAG	4	DD	Event user notification flag *
10	CCB_COUNT	2	DW	Count of CCBs chained to the EVENT_CCB_POINTER *
12	EVENT_CCB_POINTER	4	DD	Pointer to CCBs *
16	BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR *

* Indicates a returned value.

** Indicates that these parameters can also be defined as an 8086 declaration type of DB.

Table 3-46 (Page 2 of 2). CCB Parameter Table for READ

Offset	Parameter Name	Byte Length	8086 Type	Description
18	FIRST_BUFFER_ADDR	4	DD	Address of first receive buffer *
22	RCV_FRAME_COUNT	2	DW	Count of received frames chained to RCV_FRAME_ADDR *
24	RCV_FRAME_ADDR	4	DD	Address of received frames *
28	EVENT_ERROR_CODE	2	DW	Exception code * **
30	EVENT_ERROR_DATA	6	DW	Exception parameters * **
For DLC Status Changes: For DLC status change events posting, the fields starting at offset 10 are defined differently. The descriptions of those fields starts here.				
10	STATION_ID	2	DW	SAP or link station ID *
12	DLC_STATUS_CODE	2	DW	DLC status code *
14	FRMR_DATA	5	DB	Frame data *
19	ACCESS_PRIORITY	1	DB	The new access priority *
20	REMOTE_NODE	6	DB	Remote node address *
26	REMOTE_SAP	1	DB	Remote SAP address *
27		1	DB	Reserved
28	USER_STAT_VALUE	2	DW	User status value *

* Indicates a returned value.

** Indicates that these parameters can also be defined as an 8086 declaration type of DB.

STATION_ID

Explanation: The station ID to be posted for events.

OPTION_INDICATOR

Explanation: Option indicator for matching READ command requests. The option indicator is used to match events with the READ command for posting of events.

Indicator	Meaning
X'00'	Match READ command using station ID, X'nnss', where "nn" is the SAP number and "ss" is the station number. All events on the completion list associated with the station ID number are checked for a READ command match.
X'01'	Match READ command using the SAP number of the station ID, X'nn00', where "nn" is the SAP number. All events on the completion list associated with the SAP number are checked for a READ command match.
X'02'	Match READ command using all events. All events on the application program's completion list are checked for a READ command match.

If the OPTION_INDICATOR parameter is not X'00' through X'02', the command terminates with CCB_RETCODE set to X'06'.

EVENT_SET

Explanation: Set of events for notification.

There are 7 different types of asynchronous events that can be posted. A single READ command can request to be posted for more than one event. Thus, multiple bits can be set (non-zero). Each event is bit-mapped using the EVENT_SET byte as described below:

Bit **Description**

7 Reserved.

6 System Action (non-critical).

If a non-critical system action occurs while using the System Key defined by the configuration parameters, one of the following is true:

- The DIR.READ.LOG command has been issued to read the adapter or direct interface logs.
- The DIR.SET.FUNCTIONAL.ADDRESS command has been issued to modify the functional address.
- The DIR.SET.GROUP.ADDRESS command has been issued to modify the group address.
- The DLC.RESET command has been issued to reset SAP stations.

In these cases the adapter is not closed or reinitialized, and service can still be provided to application programs once the following occur:

- The functional or group addresses have been reset.

  This is only applicable if an application program was using one of the addresses and the system action resulted in the address not being suitable for the application program.
- The SAP stations have been reopened.

  This is only applicable if a SAP station used by an application program was reset by the system action.

All other non-critical system actions do not require the application program to take further action as service has not been disrupted.

5 Network Status (non-critical).

Notification of a Network Status change.

4 Critical Exception.

A critical exception occurs when the adapter is abruptly closed or reinitialized. The adapter can be closed either by an unrecoverable failure or by the action of a system administrator issuing the DIR.CLOSE.ADAPTER command with the System Key defined by the configuration parameters. The adapter can also be reinitialized by the action of a system administrator issuing the DIR.INITIALIZE command with the System Key defined by the configuration parameters.

Individual sub-events that result in critical exceptions are:

- Network Status (Adapter closed cases only)
- Adapter Check
- PC Error
- System Action (DIR.INITIALIZE and DIR.CLOSE.ADAPTER commands).

These critical exception events are provided in the CRITICAL_SUBSET field when a critical exception occurs.

Warning: If a critical exception occurs and an application program does not have a READ command pending, the only action taken by the protocol driver will be to set the return codes for all pending commands to X'07'. In this case, no pointers to buffers from SAP buffer pools or receive frames and no pointer to a list of outstanding CCBs will be provided to the user.

3 DLC status change.

Notification of a DLC status change.

2 Receive data.

MAC (Token-Ring Network only) and non-MAC frames are received.

1 Transmit completion.

A DLC or DIR TRANSMIT command is completed. The transmit completion bit is used to expedite data transfer operations.

0 Command completion.

A DLC or DIR command is completed.

If EVENT_SET is set to X'00' (no event bit is set), the command terminates with a CCB_RETCODE of X'05'.

EVENT

Explanation: The posting event.

This field contains the event being posted. Bits 0 through 6 are bit-mapped as described above. For each posting event, a single bit is set to indicate the particular event that has occurred. Each type of event that is posted will have different data placed into the CCB of the READ command. Table 3-47 describes the additional CCB fields used for each event:

Table 3-47 (Page 1 of 2). Posted CCB Fields for Each Event

Event	Fields Posted
Command Completion	• CCB_COUNT • EVENT_CCB_POINTER • BUFFER_COUNT (DLC.CLOSE.SAP and DLC.RESET commands) • FIRST_BUFFER_ADDR (DLC.CLOSE.SAP and DLC.RESET commands) • RCV_FRAME_COUNT (DLC.CLOSE.SAP, DLC.CLOSE.STATION, DLC.RESET, and RECEIVE commands) • RCV_FRAME_ADDR (DLC.CLOSE.SAP, DLC.CLOSE.STATION, DLC.RESET, and RECEIVE commands)

Table 3-47 (Page 2 of 2). Posted CCB Fields for Each Event

Event	Fields Posted
Transmit Completion	• CCB_COUNT • EVENT_CCB_POINTER
Receive Data	• RCV_FRAME_COUNT • RCV_FRAME_ADDR
DLC Status Change	• NOTIFICATION_FLAG Starting at offset 10 in the READ command's CCB parameter table, the DLC Status Table is defined below. • STATION_ID • DLC_STATUS_CODE • FRMR_DATA • ACCESS_PRIORITY • REMOTE_NODE • REMOTE_SAP • USER_STAT_VALUE
Critical Exception	• CRITICAL_SUBSET • NOTIFICATION_FLAG • CCB_COUNT • EVENT_CCB_POINTER • BUFFER_COUNT • FIRST_BUFFER_ADDR • RCV_FRAME_COUNT • RCV_FRAME_ADDR • EVENT_ERROR_CODE • EVENT_ERROR_DATA
Network Status (Non-Critical)	• NOTIFICATION_FLAG • EVENT_ERROR_CODE
System Action (Non-Critical)	• NOTIFICATION_FLAG • CCB_COUNT • EVENT_CCB_POINTER • BUFFER_COUNT • FIRST_BUFFER_ADDR • RCV_FRAME_COUNT • RCV_FRAME_ADDR • EVENT_ERROR_CODE • EVENT_ERROR_DATA

CRITICAL_SUBSET

Explanation: Exception event causing a critical exception.

When a critical exception event posts, this field contains the actual event that resulted in the critical exception. The individual events are mapped as follows:

Event	**Meaning**
X'01'	Network Status
X'02'	Adapter Check
X'03'	PC System Error
X'04'	System Action.

NOTIFICATION_FLAG

Explanation: User notification flag for events.

For notification of exception conditions, the user exception flag defined with the DIR.SET.EXCEPTION.FLAGS command is returned.

For notification of a DLC status change, this value will contain the DLC_STATUS_FLAG, as defined with the DLC.OPEN.SAP command, for the link station that has experienced the change.

These flags are preserved across invocations and may be used by the application program for user-specific data.

CCB_COUNT

Explanation: The number of CCBs chained to the EVENT_CCB_POINTER field as described below.

EVENT_CCB_POINTER

Explanation: Points to a CCB when notifying an application program of a single completed command.

When notifying an application program of the DIR.CLOSE.ADAPTER, DIR.TIMER.CANCEL.GROUP, DLC.CLOSE.SAP, and DLC.RESET commands completing, this field points to the completing command with a list of pending commands chained together using the CCB_POINTER parameter of the commands.

If TRANSMIT commands are issued specifying that their CCBs be chained together upon completion, this field points to the first member of a list of completed TRANSMIT command CCBs chained using the CCB_POINTER parameter of the commands.

If an exception occurs that causes the adapter to close (encounters an unrecoverable error) or causes SAP stations to be reset, all pending commands are chained to the EVENT_CCB_POINTER field using the CCB_POINTER parameter of the commands. The CCB_RETCODE of all commands chained to the EVENT_CCB_POINTER will be set as follows:

- If an Adapter Check, Network Status, or PC System Detected Error occurs, the CCB_RETCODE is X'07'.
- If a System Action occurs, the CCB_RETCODE is X'62'.

Note: The adapter can be closed either by an unrecoverable failure or by the action of a system administrator issuing commands with the System Key. In addition, the System Key can be used to reset SAP stations. If a command can use the System Key, this is noted in the description of the command.

BUFFER_COUNT

Explanation: The number of buffers chained to the FIRST_BUFFER_ADDR field.

FIRST_BUFFER_ADDR

Explanation: Address of the first buffer in the buffer pools being returned.

If the adapter encounters an unrecoverable error, a System Action exception occurs, or the DLC.CLOSE.SAP or DLC.RESET commands have been issued that cause SAP or direct stations to be closed or reset, all buffers contained in associated SAP or direct station buffer pools are chained to this field. This field contains the address of the first buffer of the buffer pool with all other buffers chained.

Note: The adapter can be closed either by a catastrophic failure or by the action of a system administrator issuing commands with the System Key. In addition, the System Key can be used to reset SAP stations. If a command can use the System Key, this is noted in the description of the command.

RCV_FRAME_COUNT

Explanation: The number of frames chained to the RCV_FRAME_ADDR field.

RCV_FRAME_ADDR

Explanation: Address of received frames.

When data is received and posted with the READ command requesting receive data, the RCV_FRAME_ADDR field contains the address of the first buffer of a frame with all other frames chained using the first buffer of each frame.

If the adapter encounters an unrecoverable error, a System Action exception occurs, or DLC.CLOSE.SAP, DLC.CLOSE.STATION, and DLC.RESET commands have been issued that cause the station ID to be closed or reset, all received frames associated with the station IDs on the completion list are removed and chained to this field.

Note: The adapter can be closed either by a catastrophic failure or by the action of a system administrator issuing commands with the System Key. In addition, the System Key can be used to reset SAP stations. If a command can use the System Key, this is noted in the description of the command.

EVENT_ERROR_CODE

Explanation: Reason, Status, or Error Codes.

Codes used to identify error conditions and status changes for exceptions are placed into the EVENT_CODE field. See “Exception Indications” on page B-46 for all exception conditions.

EVENT_ERROR_DATA

Explanation: Exception parameters.

The Adapter Check exception and a PC System Detected Error exception have 3 parameters defined for passing maintenance and RAS data. These parameters are placed into this field. See “Exception Indications” on page B-46 for all exception conditions.

DLC Status Change Events

For DLC status change events posting, the fields starting at offset 10 are defined differently. The descriptions of those fields are as follows:

STATION_ID

Explanation: The SAP or link station ID for DLC status change.

DLC_STATUS_CODE

Explanation: The DLC status code.

Note: See Appendix B, "Return Codes" for the DLC status codes.

Multiple bits may be set when a DLC status is posted.

FRMR_DATA

Explanation: Frame data.

Five bytes of reason code that are applicable when an FRMR is either transmitted or received. Refer to the *IBM Token-Ring Network Architecture Reference* for a complete description of this information.

ACCESS_PRIORITY

Explanation: The new access priority that is applicable when status bit 5 is on. The format is B'nnn00000' where "nnn" is the access priority. This byte is ignored and is set to zero when PC Network or Ethernet adapters are used.

REMOTE_NODE

Explanation: The 6-byte node address of the remote partner for a newly opened link station. Applicable when status bit 10 is on.

REMOTE_SAP

Explanation: The 1-byte remote SAP address for a newly opened link station. Applicable when status bit 10 is on.

USER_STAT_VALUE

Explanation: User status value as defined in the DLC.OPEN.SAP command.

READ.CANCEL

Hex 32

READ.CANCEL

Command Description: This command is for **CCB2 only**. It cancels a pending READ command for an individual application program.

Command Specifics: This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver. The canceled READ CCB (if one is pending) is terminated with the CCB_RETCODE set to X'0A'. The READ command is not posted using the post semaphore. The post semaphore for the READ.CANCEL command is cleared to signal the completion of the READ.CANCEL and the READ command is not posted except to set the return code.

The fields CCB_PARM_OFFSET and CCB_PARAMETER_1 are combined to form a 4-byte (8086 declaration type DD) address of the READ CCB to be canceled.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

RECEIVE

Hex 28

RECEIVE

Command Description: This command receives data for the station defined in the STATION_ID field of the CCB.

Command Specifics: When a RECEIVE command is issued, it is queued in the workstation, awaiting received data for the specified station. Multiple RECEIVE commands can be active at one time, but there can be only one for each specific station ID.

For CCB1: Once data is received for a pending RECEIVE command and there are adequate receive buffers available in the pool, the following takes place:

- The protocol driver fills receive buffers from the appropriate buffer pool and places the address of the first buffer in the RECEIVE command parameter table.
- If the optional RECEIVED_DATA user appendage is defined and the return code is X'00', the following happens:
  - The address of the CCB is placed in registers DS and SI.
  - The address of the first receive data buffer is placed in registers ES and BX.
  - The RECEIVE_DATA user appendage exit is taken.

  The RECEIVE command remains active to receive any data that follows. Therefore, once a RECEIVE command is issued for a station, it continues to be active until terminated by an exception condition, until a RECEIVE.CANCEL is issued for the RECEIVE, or until the SAP or link station is closed.

  If the optional RECEIVED_DATA user appendage is not defined or the return code is not X'00', the command is completed in the same way as any other command. The CCB_CMD_CMPL user exit is taken with the CCB address in registers ES and BX.

When a DLC.CLOSE.SAP or DLC.CLOSE.STATION is issued, the RECEIVE command associated with that SAP or station is terminated with a CCB_RETCODE of X'0A', and the address of the RECEIVE command CCB is placed in the CCB_POINTER field of the command causing the SAP or station to close. The RECEIVE command's completion appendage is not taken.

When the RECEIVE command has been issued with RECEIVE_DATA set, and when data has been received successfully, this command never actually completes with a return code of X'00'. However, when data is received successfully, there is an implied return code of X'00'. The actual return code remains X'FF'. A return code is set only when the command is terminated, for example, when a lost data condition occurs or if the RECEIVE_DATA was not set when the command was issued.

For CCB2: Once data is received for a pending RECEIVE command and there are adequate receive buffers available in the pool, the following takes place:

- The protocol driver fills receive buffers from the appropriate buffer pool and places the address of the first buffer in the RECEIVE command parameter table.
- If the optional RECEIVED_FLAG is not set or the return code is not X'00', the command is completed in the same way as any other command. The CCB_CMD_FLAG is used to determine how the command completion notification should be handled.
- If the optional RECEIVED_FLAG is set and the return code is X'00', the following happens:
  - If a READ command defined for notification of receive data is pending, the addresses are copied into the READ command's parameter table, and the READ semaphore is cleared.
  - If no READ command defined for notification of receive data is pending when the receive data event occurs, the address of the first receive buffer is placed onto a completion list. Upon reception of a READ command, the completion list is scanned. If the RCV_READ_OPTION field of the RECEIVE command is set for chaining, all data received for the station ID is chained and posted at once. The first receive buffer is copied into the READ command's parameter table, and the READ semaphore is cleared.

  The RECEIVE command remains active to receive any data that follows. Therefore, once a RECEIVE command is issued for a station, it continues to be active until terminated by an exception condition, until a RECEIVE.CANCEL is issued for the RECEIVE, or until the SAP or link station is closed.

When a DLC.CLOSE.SAP or DLC.CLOSE.STATION command is issued, the RECEIVE command associated with that SAP or station is terminated with a CCB_RETCODE of X'0A', and the address of the RECEIVE command CCB is placed in the CCB_POINTER field of the command causing the SAP or station to close. Any READ commands that may be pending that request receive data posting or clearing of semaphores defined by the RECEIVE command are not completed.

When the RECEIVE command has been issued with RECEIVE_FLAG set, and data has been received successfully, this command never actually completes with a return code of X'00'. However, when data is received successfully, there is an implied return code of X'00'. The actual return code remains X'FF' (command in process). A return code is set only when the command is terminated, for example, when a lost data condition occurs or if the RECEIVE_FLAG was not set when the command was issued.

For CCB3: Once data is received for a pending RECEIVE command and there are adequate receive buffers available in the pool, the following takes place:

- The protocol driver fills receive buffers from the appropriate buffer pool and places the address of the first buffer in the RECEIVE command parameter table.
- If the optional RCV_DATA_APPNDG user appendage is defined and the return code is X'00', the protocol driver calls the application program's device driver with the following set:
  - Register DI contains the RCV_DATA_APPNDG as defined by the RECEIVE command.
  - An invocation code of X'0001' has been pushed onto the stack. Before returning control to the protocol driver, the application program must remove the invocation code from the stack.
  - Register DS contains the application program device driver's protect mode data segment selector.
  - Register CX contains the adapter number.
  - Registers ES and BX contain a virtual address to the first SAP buffer of the receive information.
  - Registers AX and SI contain a virtual address of the RECEIVE command's CCB for which receive data has been processed.

  The RECEIVE command remains active to receive any data that follows. Therefore, once a RECEIVE command is issued for a station, it continues to be active until terminated by an exception condition, until a RECEIVE.CANCEL is issued for the RECEIVE, or until the SAP or link station is closed.

When a DLC.CLOSE.SAP or DLC.CLOSE.STATION command is issued, the RECEIVE command associated with that SAP or station is terminated with a CCB_RETCODE of X'0A', and the address of the RECEIVE command CCB is placed in the CCB_POINTER field of the command causing the SAP or station to close. The RECEIVE command's completion appendage is not taken.

When the RECEIVE command has been issued with RCV_DATA_APPNDG set, and data has been received successfully, this command never actually completes with a return code of X'00'. However, when data is received successfully, there is an implied return code of X'00'. The actual return code remains X'FF' (command in process). A return code is set only when the command is terminated, for example, when a lost data condition occurs or if the RCV_DATA_APPNDG was not set when the command was issued.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-48. CCB Parameter Table for RECEIVE

Offset	Parameter Name	Byte Length	8086 Type	Description
0	STATION_ID	2	DW	Defines the station receiving data.
2	USER_LENGTH	2	DW	Length of user data in buffers.
For CCB1:				
4	RECEIVED_DATA	4	DD	Address of the receive data appendage.
For CCB2:				
4	RECEIVE_FLAG	4	DD	Optional receive user flag.
For CCB3:				
4	RCV_DATA_APPNDG	2	DW	Offset to receive data appendage.
6		2	DW	Reserved for the application program. Can be used for segment or selector.
For all CCBs:				
8	FIRST_BUFFER	4	DD	First receive buffer address from the adapter. *
12	OPTIONS	1	DB	RECEIVE options.
For CCB2:				
13		3	DB	Reserved.
16	RCV_READ_OPTION	1	DB	Read posting option.
For CCB3:				
13		3	DB	Reserved.
16		1	DB	Reserved for the application program.

* Indicates a returned value.

STATION_ID

Explanation: Defines the station and the kind of data the station will receive. The station ID is defined in "Stations, SAPs, and IDs" on page 2-27. It identifies the station to receive data as follows:

X'0000'	Direct station, receive both MAC and non-MAC frames. Direct station receives only non-MAC frames on the PC Network or Ethernet.
X'0001'	Direct station, receive MAC frames. Not used on the PC Network or Ethernet. Reserved for the Token-Ring Network.
X'0002'	Direct station, receive non-MAC frames.

X'nn00' SAP, receive data for SAP 'nn'.

X'nnss' Link station, receive data for SAP 'nn', station 'ss'.

Every station that is defined to the adapter can have a RECEIVE command pending, but there can be only one RECEIVE command for any specific station.

Note: If no RECEIVE command is active for link station X'nnss', the frame received by the adapter for the link station is received by SAP X'nn00' if it has a RECEIVE command active.

USER_LENGTH

Explanation: This field specifies the length of a user space in the buffer for private data. The data placed in the receive buffer starts at an offset specified by the USER_OFFSET field of the receive buffer. The information placed in the user space is not altered by the protocol driver nor is it overwritten by the received frame data. See "Buffer Pools" on page 2-40 for the receive buffer.

RECEIVED_DATA

Explanation: The address of a user-provided appendage routine that is taken when data is received. By coding this parameter, the application program can receive data and keep the same RECEIVE command active to receive subsequent data.

RECEIVE_FLAG

Explanation: This is a user flag that specifies whether or not received data should be posted using the READ command specifying notification for receive data. By setting this flag to a non-zero value, the application program can receive data and keep the same RECEIVE command active to receive subsequent data.

RCV_DATA_APPNDG

Explanation: This is the offset of the receive data appendage where received data is posted to the application program. This address offset is passed to the application program's device driver intercommunication entry point in register DI when the protocol driver calls the application program's device driver.

FIRST_BUFFER

Explanation:

For CCB1 and CCB3: A returned value indicating the address of the first buffer. This is the same address value that is placed in registers ES and BX. If the address is X'00000000', there is no receive data.

For CCB2: A returned value. If no RECEIVE_FLAG is specified, the address of the first receive buffer is placed into this field. This is the same value placed in the READ command's CCB parameter table when a READ command requesting notification of receive data is completed. If the address is X'00000000', there is no receive data.

OPTIONS

Explanation: Options set by the application program to inform the protocol driver how to present received information to the application program.

Bits	Description
7	Contiguous MAC. Not used on PC Network and is applicable only if the received frame is a MAC frame. • If this bit is on, the entire frame is placed into the buffers as a continuous data string after the USER_SPACE. See "Receive Buffers" on page 2-41 for a discussion of contiguous and non-contiguous buffers. • If this bit is off, the 32-byte LAN header is removed from the frame and is placed in a special location in buffer 1. In this case, the data is stored in a non-contiguous buffer.
6	Contiguous data and is applicable only if the received frame is a non-MAC frame. • If this bit is on, the entire frame is placed into the buffers as a continuous data string. Buffer 1 contains contiguous data in this case. • If this bit is off, the 32-byte LAN header is placed in buffer 1, followed by the DLC header and all the received data. In this case, the non-contiguous buffer data format is used.
5	Break. If this bit is on, the first received data is placed in the second receive buffer. The first buffer contains only the buffer header data.
4-0	These bits are reserved and should be zero, but they are not checked.

RCV_READ_OPTION

Explanation: This field is used only when the RECEIVE_FLAG is set to a non-zero value.

Frames that have been received are destined for SAP stations, link stations, or one of the direct stations. If a RECEIVE command has been issued for a SAP station but not for any of the link stations opened under the SAP, then data received for the link station is received using the SAP's RECEIVE command. In this case, it is possible for the application program to issue a READ command requesting receive data for a link station. If RECEIVE commands have been issued for all link stations opened under a SAP, it is also possible for a READ command to be issued requesting receive data for a SAP and its link stations. To prevent the application program from having to issue a READ command for each frame received, received frames can be chained together. To do this, the application program must specify ahead of time how received frames are to be chained. The RCV_READ_OPTION field should be set as described below to allow these chained frames:

- If the RCV_READ_OPTION field contains a X'00', received frames are not chained.

  This option specifies that received frames for the station are to be placed separately onto the protocol driver's completion list. A READ command will have to be issued to retrieve each frame from the completion list.

- If the RCV_READ_OPTION field contains a X'01', all frames received for a link station are chained.

  This option specifies that received frames for the station are to be chained onto the protocol driver's completion list for the specified station ID. A single READ command can be issued to retrieve all chained frames from the completion list for this station ID. If the RECEIVE command is for a SAP station, this option has the same effect as a RCV_READ_OPTION field containing X'02' for frames received for the SAP station.

- If the RCV_READ_OPTION field contains X'02', all frames received for a SAP are chained.

  This option specifies that received frames for the station ID are to be chained onto the protocol driver's completion list. All frames received for a SAP and its link stations are chained together. A single READ command can be issued to retrieve all chained frames from the completion list for this link station.

  If RECEIVE commands have been issued for link stations, the options specified with the link station's RECEIVE commands are used to determine how receive data is chained. If a link station has a RECEIVE command pending specifying that its received frames should be chained on a link station basis, its frames would not be chained together with the SAP's frames. All link stations that do not have a RECEIVE command pending will have their received frames chained as specified by the SAP's RECEIVE command. For example, if a link station did not have a RECEIVE command pending, but the SAP did have one specifying that no received frames be chained, all frames received for the link station and the SAP would be placed onto the protocol driver's completion list individually. If neither the SAP nor the link station had a RECEIVE command pending, the direct station would be used. See "Stations, SAPs, and IDs" on page 2-27.

Notes:

1. The RCV_READ_OPTION for the SAP and the link station are independent of each other. The option specified for a link station's RECEIVE command is considered first when determining if received frames are to be chained. A RECEIVE command does not have to be outstanding for a SAP in order to have received frames for link stations chained on a SAP basis. Consequently, this requires that link stations have RECEIVE commands outstanding with RCV_READ_OPTION set to X'02'.
2. If the value of the RCV_READ_OPTION field is not within the range of X'00' to X'02', the command terminates with CCB_RETCODE of X'06'.
3. The RCV_READ_OPTION of a SAP's RECEIVE command is used to determine how the SAP's receive data is chained.
4. The RCV_READ_OPTION of a link station's RECEIVE command is used to determine how the link station's receive data is chained.
5. The RCV_READ_OPTION of a SAP's RECEIVE command is used to determine how link station receive data for the SAP is chained when its link stations do not have RECEIVE commands outstanding.
6. A single RECEIVE command for a SAP (RCV_READ_OPTION = X'00') can be used to receive all data for a SAP and its link stations with each received frame being placed separately on the completion list.

7. A single RECEIVE command for a SAP (RCV_READ_OPTION = X'01') can be used to chain all receive data for a SAP and its link stations on individual station ID queues using a separate completion list entry for each station ID.
8. A single RECEIVE command for a SAP (RCV_READ_OPTION = X'02') can be used to chain all receive data for a SAP and its link station on a single SAP queue using one completion list entry.
9. READ commands need to match RECEIVE commands' RCV_READ_OPTION settings.

For information on receive buffers, see "Buffer Pools" on page 2-40.

RECEIVE.CANCEL

Hex 29

RECEIVE.CANCEL

Command Description: This command cancels a RECEIVE command on any specific SAP or link station, including the direct station.

Command Specifics: The station ID specifies the SAP or station of a pending receive that is to be canceled. This command is executed entirely in the workstation. The return code is available to the application program upon return from the protocol driver.

The canceled RECEIVE CCB, if there is one, is terminated with a CCB_RETCODE value of X'0A' (command canceled by user request).

For CCB1 and CCB3: The RECEIVE command's completion appendage is not taken. However, the RECEIVE.CANCEL command's completion appendage is taken, if provided.

For CCB2: If a READ command is pending that requests notification of receive data, the cancellation of a RECEIVE will not affect the READ. The READ command is not executed, and the READ command is not posted except for the setting of the return code. The RECEIVE.CANCEL commands are posted as defined by the CCB_CMPL_FLAG and CCB_SEMAPHORE parameters. After completion of the RECEIVE.CANCEL, the CCB_POINTER contains the address of the canceled RECEIVE command.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

For CCB1: The station ID is located at the CCB_PARM_TAB field of the control block.

For CCB2 and CCB3: The station ID is located at the CCB_PARM_OFFSET field of the control block.

RECEIVE.MODIFY

Hex 2A

RECEIVE.MODIFY

Command Description: This command is for **CCB1 and CCB3 only**. It receives data and puts some of the data into a buffer not taken from the SAP buffer pool.

Command Specifics: This command operates in the same way as the RECEIVE command, with the following exceptions:

- There are no receive options in the parameter table.
- Only data (non-MAC) frames can be received.
- Data is received into one SAP buffer and one user buffer.
- The format of received data (that is, the data following the DLC header) is assumed to be:

  `llllhh...hhdd...dd`

  where:

  - `llll` is a 2-byte field whose value is its own length (2 bytes) plus the length in bytes of the `hh...hh` field. The length field (`llll`) has a format defined as DW.
  - `hh...hh` is a message header.
  - `dd...dd` is message data.
- When data is received, a SAP buffer is obtained by the protocol driver.

For CCB1:

- The first 58 bytes of the SAP buffer are prepared exactly as when executing a RECEIVE command with the option not continuous data.
- At byte 58 (plus user length, if applicable), the received `llllhh...hh` is placed into the SAP buffer. (If the data [`llllhh...hhdd...dd`] exceeds the length of the buffer, the frame is discarded by the protocol driver and no indication is given to the application program.)
- The protocol driver calls a subroutine defined in the SUBROUTINE@ parameter to obtain the length and location of an application program buffer.

  An option is to call the RECEIVE_DATA appendage once data has been placed into the application program's buffer.

For CCB3:

- The first 62 bytes of the SAP buffer are prepared exactly as when executing a RECEIVE command with the option not continuous data.
- At byte 62 (plus user length, if applicable), the received `llllhh...hh` is placed into the SAP buffer. (If the data [`llllhh...hhdd...dd`] exceeds the length of the buffer, the frame is discarded by the protocol driver and no indication is given to the application program.)
- The protocol driver calls the application program's device driver intercommunication entry point, passing the offset of a subroutine address (SUBROUTINE@) in the DI register. This call is made by the

protocol driver to obtain the length and location of an application program buffer.

An option is to call the application program's device driver intercommunication entry point with the RCV_DATA_APPNDG parameter passed in the DI register, once data has been placed into the application program's buffer.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-49. CCB Parameter Table for RECEIVE.MODIFY

Offset	Parameter Name	Byte Length	8086 Type	Description
0	STATION_ID	2	DW	Defines the station receiving data
2	USER_LENGTH	2	DW	Length of user data in buffers
For: CCB1:				
4	RECEIVED_DATA	4	DD	Optional user exit for received data
For: CCB3:				
4	RCV_DATA_APPNDG	2	DW	Received data appendage offset
6		2	DW	Reserved for the application program
For CCB1 and CCB3:				
8	FIRST_BUFFER	4	DD	First receive buffer address *
For CCB1:				
12	SUBROUTINE@	4	DD	The address of a subroutine
For CCB3:				
12	SUBROUTINE@	2	DW	Offset to the subroutine
14		2	DW	Reserved for the application program

* Indicates a returned value.

@ Indicates an address throughout this book.

STATION_ID

Explanation: Defines the station and the kind of data the station will receive. The station ID is defined in "Stations, SAPs, and IDs" on page 2-27. It identifies the station to receive data as follows:

X'0000'	Direct station, receive non-MAC frames only
X'0002'	Direct station, receive non-MAC frames
X'nn00'	SAP, receive data for SAP 'nn'
X'nnss'	Link station, receive data for SAP 'nn', station 'ss'.

Every station that is defined for this adapter can have a RECEIVE or RECEIVE.MODIFY command pending, but there can be only one RECEIVE or RECEIVE.MODIFY command for any specific station.

USER_LENGTH

Explanation: This field specifies the length of a user space in the buffer for private data. The data placed in the receive buffer starts at an offset specified by the USER_OFFSET field of the receive buffer. The information placed in the user space is not altered by the protocol driver or the received frame data. See "Buffer Pools" on page 2-40 for the receive buffer.

RECEIVE_DATA

Explanation: This is the address of an appendage routine provided by the application program that is used to receive data. By coding this parameter the application program can receive data and keep the same RECEIVE command active to receive subsequent data.

When the protocol driver has updated the application program buffer obtained from the SUBROUTINE@ call, the receive appendage is called to post the reception of data. See "RECEIVE" on page 3-95 for register usage when the receive appendage is called to post the reception of data.

When the application program's receive appendage is called to post the reception of data, bytes 6 and 7 (LENGTH_IN_BUFFER) of the SAP buffer referenced by ES and BX is set to the length of the data moved into the application program's receive buffer starting at offset 30 (SOURCE_ADDRESS). If the received data is more than what will fit in the application program's receive buffer, bytes 6 and 7 are set to X'FFFF', and any excess data is lost.

Upon return, the SAP buffer is returned to the available pool.

RCV_DATA_APPNDG

Explanation: This is an address offset of the receive data appendage where receive modify data information is posted to the application program.

When the protocol driver has updated the user buffer obtained from the SUBROUTINE@ call with the received data, the application program's device driver is called with this receive appendage offset passed in register DI. The protocol driver enters the application program device driver's intercommunication entry point with a Call Far instruction, and the application program's device driver must return with a Return Far instruction. This call is made using the application program's device driver entry point obtained when the DIR.OPEN.ADAPTER command is issued. See "RECEIVE" on page 3-95 for register usage when the application program is called to post the reception of data.

When the application program device driver's receive data appendage is called to post the reception of data, bytes 6 and 7 (LENGTH_IN_BUFFER) of the SAP buffer referenced by ES and BX is set to the length of the data moved into the user's receive buffer starting at offset 34 (SOURCE_ADDRESS). If the received data is more than what will fit in the user's receive buffer, bytes 6 and 7 are set to X'FFFF', and any excess data is lost.

Upon return, the SAP buffer is returned to the available pool.

FIRST_BUFFER

Explanation: This is a returned value indicating the address of the first buffer. This is the same address value that is placed in registers ES and BX. If the address is X'00000000', there is no receive data.

SUBROUTINE@

Explanation:

For CCB1: The address of a subroutine or appendage that the protocol driver calls to obtain the address and length of an additional user buffer. The protocol driver enters the subroutine with a Call Far instruction, and the subroutine must return with a Return Far instruction. This field must be provided. When the SUBROUTINE@ is entered, the following parameters are set:

- Registers ES and BX point to the SAP buffer.
- Registers ES and DI point to offset 30 of the SAP buffer (the source address of the frame).
- Registers AX and SI point to this adapter's node address in shared RAM.
- Register CX contains the adapter number.
- Register DX contains the number of bytes left in the frame.

When the appendage subroutine is completed, it must set the AL register and issue a Return Far instruction.

If the AL register is set to zero, then:

- Registers ES and DI point to a receive buffer.
- Register CX indicates the length of the receive buffer.

If the AL register is not set to zero, then:

- The protocol driver returns the SAP buffer to the pool.
- The received data is discarded.
- The received data appendage is not taken.

If the frame is an I frame, it is treated by the adapter's DLC logic as if it had been successfully received.

For CCB3: This is the address offset of the application program's subroutine that the protocol driver calls to obtain a user's buffer length and address. The protocol driver enters the application program's device driver with a Call Far instruction, and the application program's device driver must return with a Return Far instruction.

When data is received, the application program's device driver is called to obtain a buffer. This call is made using the application program's device driver entry point obtained when the DIR.OPEN.ADAPTER command is issued. For this call, the following parameters are set:

- Register DI contains the offset of a subroutine within the application program's device driver code segment.
- Register DS contains the called device driver's protect mode data segment selector.

- Registers ES and BX contain a virtual address to a SAP buffer.
- Registers AX and SI contain a virtual address to this adapter's node address.
- Register CX contains the adapter number.
- Register DX contains the number of bytes of the frame that have not been removed from the adapter's receive buffers.

Before returning from the SUBROUTINE@ call, the application program's device driver must set the AL register to indicate to the protocol driver what action is to be taken.

If the AL register is set to zero, then:

- Registers DX and BX contain a 32-bit physical address of a user receive buffer, where DX contains the most significant word of the address.
- Register CX contains the length of the user receive buffer.

If AL register is not set to zero, then:

- The protocol driver returns the SAP buffer to the available pool.
- The received frame is discarded. If the discarded frame is an I-frame, the frame is treated by the adapter's DLC logic as if it had been successfully received.
- The received data appendage is not taken.

See "Buffer Pools" on page 2-40 for more information.

TRANSMIT.DIR.FRAME

Hex 0A

TRANSMIT.DIR.FRAME

Command Description: This command transmits data for the *direct station.*

Command Specifics: This command can be used only for the direct stations. It is invalid for all station IDs except direct stations.

The entire transmission frame must be prepared by the application program, including the LAN header and any required data headers.

The LAN header in the user's buffer must reserve the space for the adapter to insert the source address. The user's buffer is not altered. The adapter sets the source address into the transmit buffer in shared RAM. However, the protocol driver passes the high-order bit of the source address, used to indicate the presence of routing information as supplied by the application program, to the adapter. The high-order bit is loaded, as it was supplied, into shared RAM. The adapter verifies that the access priority and source class (MAC frames only) are valid. MAC frames do not exist on the PC Network or on Ethernet. See Figures 2-7, 2-8, and 2-9 starting on page 2-49 for the configuration of the header.

See "Transmit Command Specifics" on page 3-110 for information common to all TRANSMIT commands.

TRANSMIT.I.FRAME

Hex 0B

TRANSMIT.I.FRAME

Command Description: This command transmits "information" data for a *link station.*

Command Specifics: This command can be used only for a link station. The only data supplied by the application program is the actual data portion of the message. The LAN and DLC headers are not built by the application program. The DLC code handles all transmission retries. The maximum size of the user data provided with this command is limited to the DHB size minus 6 bytes.

See "Transmit Command Specifics" on page 3-110 for information common to all TRANSMIT commands.

Transmit Completion

Under normal conditions, this command terminates when verification of its receipt has been received from the receiving link station. Since link stations are controlled by the DLC functions, DLC information is exchanged to verify delivery at the link level in addition to the verification at the physical level.

For CCB1:

- If the return code is X'00' and the link station's MAXOUT parameter is not a value of 1, multiple TRANSMIT.I.FRAME commands for a given link station may be completed at the same time. All frames acknowledged by a specific received acknowledgment are completed at the same time for a given link station. The completed TRANSMIT.I.FRAME commands for a given link station are queued in the order in which they were issued, each one pointing to the next by the CCB_POINTER field until the last CCB, which has zeros in the field. The adapter takes the appendage exit of the first CCB in the queue.
- In a case where TRANSMIT commands are queued and an abortive condition occurs, all pending TRANSMIT.I.FRAME commands for a given link station are queued and ended with appropriate return codes, as described in the previous step.

If the return code is X'28', the link station enters the disconnected mode. Once the station is in the disconnected mode, the application program will have to reestablish the connection before continuing transmission. See "Link Station States" on page 2-33.

For CCB2:

If the application program has specified that each TRANSMIT.I.FRAME's CCB be placed onto the protocol driver's internal completions list (setting CCB_CMPL_FLAG), the following occurs:

- If a READ command is pending, or when a READ command is issued requesting notification of completed TRANSMIT commands or command completions, then the READ command is posted. The READ command's CCB parameter table will contain a pointer to a single TRANSMIT.I.FRAME CCB or a chain of TRANSMIT.I.FRAME CCBs (if chaining is specified in the TRANSMIT commands).

- In a case where TRANSMIT commands are chained and an abortive condition occurs (for example, link lost), all pending TRANSMIT.I.FRAME commands are chained and completed, as previously explained.

If the return code is X'28', the link station enters the disconnected mode. Once the station is in the disconnected mode, the application program will have to reestablish the connection before continuing transmission. See "Link Station States" on page 2-33.

TRANSMIT.TEST.CMD

Hex 11

TRANSMIT.TEST.CMD

Command Description: This command requests the adapter to transmit a test command frame with the poll bit set. This command can be used only for a SAP.

Command Specifics: The adapter provides the DLC header. The application program must provide the LAN header and the optional test information. The first buffer must contain only the LAN header.

See "Transmit Command Specifics" on page 3-110 for information common to all TRANSMIT commands.

TRANSMIT.UI.FRAME

Hex 0D

TRANSMIT.UI.FRAME

Command Description: This command transmits unnumbered information data for a SAP. This command can be used only for a SAP.

Command Specifics: The adapter provides the DLC header information. The application program must provide the LAN header and data portions of the message. The first buffer must contain only the LAN header.

See "Transmit Command Specifics" on page 3-110 for information common to all TRANSMIT commands.

TRANSMIT.XID.CMD

Hex 0E

TRANSMIT.XID.CMD

Command Description: This command transmits an XID command with the poll bit set on.

Command Specifics: This command can be used only for a SAP. The user provides the LAN header and data field. The user must also leave room for the DLC header that is provided by the protocol driver. If the SAP option indicates that the adapter handles XID commands, the adapter will provide the data. The first buffer must contain only the LAN header.

See "Transmit Command Specifics" on page 3-110 for information common to all TRANSMIT commands.

TRANSMIT.XID.RESP.FINAL

Hex 0F

TRANSMIT.XID.RESP.FINAL

Command Description: This command transmits an XID response with the final bit on.

Command Specifics: This command can be used only for a SAP opened with the SAP option specifying that the application program will handle XID commands. The user provides the LAN header and data field. The user must also leave room for the DLC header that is provided by the protocol driver. The first buffer must contain only the LAN header.

See "Transmit Command Specifics" on page 3-110 for information common to all TRANSMIT commands.

TRANSMIT.XID.RESP.NOT.FINAL

Hex 10

TRANSMIT.XID.RESP.NOT.FINAL

Command Description: This command transmits an XID response with the final bit off.

Command Specifics: This command can be used only for a SAP opened with the SAP option specifying the application program will handle XID commands. The user provides the LAN header and data field. The user must also leave room for the DLC header that is provided by the protocol driver. The first buffer must contain only the LAN header.

See "Transmit Command Specifics" on page 3-110 for information common to all TRANSMIT commands.

Transmit Command Specifics

The seven TRANSMIT commands are variations of the same basic TRANSMIT command. The command completion, parameter table and field explanations, and return codes are explained here. All differences are noted with the specific command description.

Command Completion: The TRANSMIT command terminates when the frame has been transmitted and read back in by the adapter.

If the FS field is X'CC' (both "address recognized" and both "frame copied" bits on) and there are no other error conditions, the CCB_RETCODE will be X'00' for all TRANSMIT commands. If the FS field is anything other than X'CC', the command terminates with a CCB_RETCODE of X'22'. Except for the TRANSMIT.I.FRAME command, if the FS field is not X'CC', the protocol driver and adapter will not retransmit the frame based on its MAX_RETRY_CNT parameter. For more information on the FS field, see Table 3-50 and its field descriptions or, "Frame Status" on page B-46.

For CCB2: To reduce the number of READ commands that must be issued to receive notification of completed TRANSMIT commands, the application program can specify that completed TRANSMIT CCBs be chained together. Once

TRANSMIT CCBs are chained, a single READ command can be issued to retrieve all completed TRANSMIT CCBs at a given time.

Valid Return Codes: See "CCB Return Codes Listed by Command" on page B-5.

Parameters:

Table 3-50. CCB Parameter Table for TRANSMIT Commands

Offset	Parameter Name	Byte Length	8086 Type	Description
0	STATION_ID	2	DW	Defines the station sending data
2	TRANSMIT_FS	1	DB	Stripped FS field *
3	RSAP	1	DB	Remote SAP value
4	XMIT_QUEUE_ONE	4	DD	Address of the first transmit queue
8	XMIT_QUEUE_TWO	4	DD	Address of the second transmit queue
12	BUFFER_LEN_ONE	2	DW	Length of transmit buffer BUFFER_ONE
14	BUFFER_LEN_TWO	2	DW	Length of transmit buffer BUFFER_TWO
16	BUFFER_ONE	4	DD	The address of the first transmit buffer
20	BUFFER_TWO	4	DD	The address of the second transmit buffer
For CCB2:				
24	XMIT_READ_OPTION	1	DB	Read posting option
For CCB3:				
24		1	DB	Reserved for the application program

* Indicates a returned value.

STATION_ID

Explanation: Defines what station is sending the data. The station ID is explained in "Stations, SAPs, and IDs" on page 2-27. It identifies the station that is transmitting data as follows:

X'00nn' Direct station, transmit both MAC and non-MAC frames (TRANSMIT.DIR.FRAME only). MAC frames do not exist on the PC Network or on Ethernet.

X'nn00' SAP, transmit data for SAP 'nn' (any transmit except TRANSMIT.DIR.FRAME and TRANSMIT.I.FRAME).

X'nnss' Link station, transmit data for SAP 'nn', station 'ss' (TRANSMIT.I.FRAME only).

Note: A station can have more than one TRANSMIT command pending at one time. The protocol driver will queue TRANSMIT commands.

TRANSMIT_FS

Explanation: This is the FS field as returned by the adapter or the protocol driver. It is a copy of the FS field after the frame has gone around the ring and has been read back by the ring interface of the adapter. See "Frame Status" on page B-46.

On the PC Network or Ethernet, this field is set to X'00' if the adapter detects an error on transmission, and the command terminates with a return code of X'22'. If the adapter does not detect an error, the field is set to X'CC', and the command terminates with a return code of X'00'.

The TRANSMIT_FS field can be interrogated for the reason for transmission failure if the command terminates with CCB_RETCODE of X'22'.

This field is not returned when the TRANSMIT command is for a link station, because all transmission retry is handled by the link facilities. The FS field is explained in *IBM Token-Ring Network Architecture Reference*.

RSAP

Explanation: The SAP value of the remote SAP that the sending (local) SAP is communicating with.

This value is ignored if the sending station is a link station or a direct station.

XMIT_QUEUE_ONE

Explanation: The address of the first (or only) buffer in a queue of buffers to be transmitted. The data in all the buffers is transmitted as one frame.

The buffers in this queue are not returned to the SAP buffer pool upon command completion.

This transmit queue of buffers is not used if the value is X'00000000'.

If the NEXT_BUF_POINTER field of the first buffer is not zero, there are additional buffers in this XMIT_QUEUE_ONE queue.

See "Transmit Buffers" on page 2-46 for details of transmit queues and buffers.

XMIT_QUEUE_TWO

Explanation: The address of the second queue of buffers to be transmitted.

If there are buffers in XMIT_QUEUE_ONE, the data in XMIT_QUEUE_TWO buffers is transmitted following the data in XMIT_QUEUE_ONE buffers as one frame.

Before taking an appendage exit or posting completion, the buffers in this queue are returned to the SAP buffer pool and this field set to zero upon command completion if the return code equals zero (X'00').

This transmit queue of buffers is not used if the value is X'00000000'.

If the NEXT_BUF_POINTER field of the buffer is not zero, there are additional buffers in this XMIT_QUEUE_TWO queue.

See "Transmit Buffers" on page 2-46 for details of transmit queues and buffers.

BUFFER_LEN_ONE

Explanation: The length of the transmit buffer containing the data to be transmitted, located by the contents of the BUFFER_ONE field.

If this field is 0, all the following fields (BUFFER_LEN_TWO, BUFFER_ONE, BUFFER_TWO, XMIT_READ_OPTION) are ignored.

BUFFER_LEN_TWO

Explanation: The length of the transmit buffer containing the data to be transmitted, located by the contents of the BUFFER_TWO field.

If this field is 0, this buffer is not used.

BUFFER_ONE

Explanation: The address of the buffer containing data to be transmitted. The data is transmitted as one frame following the data in XMIT_QUEUE_ONE and XMIT_QUEUE_TWO.

The length of the buffer is defined by BUFFER_LEN_ONE.

The buffer is not used if BUFFER_LEN_ONE is 0.

See "Transmit Buffers" on page 2-46 for details of transmit queues and buffers.

For CCB3: If this field is used to reference transmit data, it must be a 32-bit physical address, not a virtual address.

BUFFER_TWO

Explanation: The address of the buffer containing data to be transmitted.

The length of the buffer is defined by BUFFER_LEN_TWO. The data is transmitted as one frame following the data in XMIT_QUEUE_ONE, XMIT_QUEUE_TWO, and BUFFER_ONE.

The buffer is not used if BUFFER_LEN_TWO is 0.

See "Transmit Buffers" on page 2-46 for details of transmit queues and buffers.

For CCB3: If this field is used to reference transmit data, it must be a 32-bit physical address, not a virtual address.

XMIT_READ_OPTION

Explanation: To eliminate the need to issue an individual READ command to receive notification for each completion of a TRANSMIT command, the application program can specify that completed TRANSMIT commands be chained together using the CCB_POINTER of each CCB. The XMIT_READ_OPTION must be set for each individual TRANSMIT command. Following is a list of the various XMIT_READ_OPTIONs:

X'00' Chain this TRANSMIT command on a link station basis when this command is completed. This option is valid only for TRANSMIT.I.FRAME commands.

X'01' Do not chain this TRANSMIT command when it completes.

X'02' Chain this TRANSMIT command on a SAP station basis when this command completes.

Chapter 4. NETBIOS

About This Chapter 4-2
NETBIOS Overview 4-2
The Network Control Block 4-3
  NCB Field Explanations 4-3
NETBIOS Operational Parameters 4-8
  Local Area Network Support Program Old Parameters 4-8
  Local Area Network Support Program New Parameters 4-13
  NETBIOS 3.0 (OS/2 EE) 4-13
  NETBIOS 4.0 4-13
  NETBIOS DLC Timers 4-14
  NETBIOS Calling Conventions Using the Local Area Network Support Program 4-14
  NETBIOS Calling Conventions Using the Dynamic Link Routine Interface 4-14
  NETBIOS Calling Conventions Using the Device Driver Interface 4-14
  NCB Completion with Wait Type Commands 4-15
  NCB Completion with No-Wait Type Commands 4-15
NETBIOS Command Descriptions 4-15
  NCB.ADD.GROUP.NAME 4-15
  NCB.ADD.NAME 4-16
  NCB.CALL 4-16
  NCB.CANCEL 4-17
  NCB.CHAIN.SEND 4-18
  NCB.CHAIN.SEND.NO.ACK 4-19
  NCB.DELETE.NAME 4-20
  NCB.FIND.NAME 4-21
  NCB.HANG.UP 4-22
  NCB.LAN.STATUS.ALERT 4-23
  NCB.LISTEN 4-24
  NCB.RECEIVE 4-25
  NCB.RECEIVE.ANY 4-26
  NCB.RECEIVE.BROADCAST.DATAGRAM 4-27
  NCB.RECEIVE.DATAGRAM 4-27
  NCB.RESET 4-28
  NCB.SEND 4-32
  NCB.SEND.BROADCAST.DATAGRAM 4-33
  NCB.SEND.DATAGRAM 4-34
  NCB.SEND.NO.ACK 4-34
  NCB.SESSION.STATUS 4-35
  NCB.STATUS 4-37
  NCB.TRACE 4-41
  NCB.UNLINK 4-45

About This Chapter

This chapter describes how NETBIOS can be used with any supported adapters to extend the adapter support software by permitting NETBIOS application programs to operate on the network.

NETBIOS Overview

NETBIOS provides a communication interface between the application program and the attached medium. All communication functions from the physical layer through the session layer are handled by NETBIOS, the adapter support software, and the adapter card.

A NETBIOS session is a logical connection between any two names on the network. A session is established by having an NCB.LISTEN issued from one name and an NCB.CALL issued from the other name. Once a session is established, two-way guaranteed-delivery communication is possible between the two names.

Two basic types of data transfer are supported. Reliable data transfer is provided by the *session* layer. If data is lost or errors occur, NETBIOS returns an error code to the application program through the NCB. Data transfer using datagram support goes directly to the *link* layer. This type of data transfer is "best effort" and receipt of data is not guaranteed.

The following are needed to use NETBIOS application programs on an IBM network:

- An IBM-supported network adapter
- Adapter support software (provided by the Local Area Network Support Program, OS/2 EE 1.3 or LAPS from NTS/2, ES 1.0, or LAN Server)
- NETBIOS (provided by the Local Area Network Support Program, OS/2 EE 1.3 or LAPS from NTS/2, ES 1.0, or LAN Server).

NETBIOS maintains a table of names that the node is known by on the network. These names are provided to NETBIOS by the application program. A name can be a unique name or a group name. NETBIOS checks the network to verify that a unique name is not already in use at another adapter. A group name can be used by several adapters. Names are used as the basis for communication between application programs. If the name is in the NETBIOS name table, a session can be established. NETBIOS can have from 1 to 254 selectable names and one NAME_NUMBER_1. The default is 16 plus 1. All names are 16 characters long. The NAME_NUMBER_1 is always present and consists of 10 bytes of binary zeros followed by the adapter's universally administered address. This NETBIOS name is referred to as NETBIOS_NAME_NUMBER_1.

For OS/2 EE 1.3: A maximum of 255 application programs (processes) can use NETBIOS 3.0 simultaneously. This number is the sum of application programs using the DLR interface and DD interface.

OS/2 EE 1.3 supports adapter numbers 0 and 1. Adapter 00 in the NCB_ADPTR_NUM field indicates the primary adapter and 01 indicates the alternate adapter.

For LAPS from NTS/2, ES 1.0, or LAN Server: LAPS from NTS/2, ES 1.0, or LAN Server supports up to four adapters. These adapters can be numbered with numbers in the range of 00–15.

For all versions of OS/2 NETBIOS: Application programs cannot share resources across process boundaries. The application program or process that allocates the resource is the sole owner of that resource.

The Network Control Block

NETBIOS is operated using a control block called the Network Control Block (NCB). (The Token-Ring Network NCB is the same as the NCB in a PC Network.)

NCB Field Explanations

Note: Throughout this manual field names ending with @ indicate an address.

The following table shows the contents of the NCB.

Table 4-1 (Page 1 of 2). Network Control Block (NCB)

Offset	Parameter Name	Byte Length	8086 Type	Description
0	NCB_COMMAND	1	DB	Command field
1	NCB_RETCODE	1	DB	Return code
2	NCB_LSN	1	DB	Local session number
3	NCB_NUM	1	DB	Number of application program name
4	NCB_BUFFER@	4	DD	Pointer to message buffer address (segment:offset)
8	NCB_LENGTH	2	DW	Buffer length in bytes
10	NCB_CALLNAME	16	DB	Name on local or remote NETBIOS. This field has a different use for the NCB.CHAIN.SEND and the NCB.RESET commands.
26	NCB_NAME	16	DB	Name on the local NETBIOS session. This field has a different use for the NCB.RESET command.
42	NCB_RTO	1	DB	Receive timeout
43	NCB_STO	1	DB	Send timeout
For all NETBIOS except the OS/2 Device Driver interface				
44	NCB_POST@	4	DD	Pointer to post routine (segment:offset) or X'00000000'

Table 4-1 (Page 2 of 2). Network Control Block (NCB)

Offset	Parameter Name	Byte Length	8086 Type	Description
For NETBIOS using the OS/2 Device Driver interface				
44	NCB_POST@	2	DB	A value returned to the post routine in register DI (0 = do not post)
46	NCB_DD_ID	2	DB	Device driver identification
For all NETBIOS				
48	NCB_ADPTR_NUM	1	DB	Adapter number
49	NCB_CMD_CMPL	1	DB	Command status
50	NCB_RESERVE	14	DB	Reserved area for all commands except NCB.RESET

NCB_COMMAND

Explanation: The command to be performed by the adapter. The high-order bit defines the wait/no-wait option. If the bit is set, the command uses the no-wait option; if the bit is clear, the command is a wait command.

When an application program issues a "wait" command to NETBIOS, control is not returned to the application program until the command is complete. When the command has been executed, check either the AL register or the NCB_RETCODE field for the completion code. The memory allocated for the NCB should not be disturbed until the NCB has been completed.

For NETBIOS using the OS/2 DD interface: With the exception of RESET and CANCEL, the NETBIOS DD interface does not have wait commands.

When an application program issues a "no-wait" command to NETBIOS, control is returned to the application program at the earliest possible time, typically before the command is completed. With the no-wait option, two return codes are returned. An immediate return code is returned when the command is accepted by NETBIOS and the final return code is returned with the command completion. If the immediate return code in the AL register is not X'00', NETBIOS does not proceed and, therefore, does not provide a final return code.

When a no-wait command is completed, if the NCB_POST@ is zero, the completion code is set by NETBIOS. It is the responsibility of the application program to determine when NETBIOS is finished by periodically checking the completion code for a change from X'FF'. If NCB_POST@ is not zero, NETBIOS takes the address to be a post routine and after command completion, control is given to the post routine. See the description for the NCB_POST@ field on page 4-6.

NCB_RETCODE

Explanation: The completion code as provided by NETBIOS.

While the value is X'FF', the application program must not change either the control block or any data associated with the command. Below are descriptions of the other return code values:

- A return code value of X'00' indicates successful completion of the command.
- Return code values of X'01' through X'2F' indicate terminations that are described in "NCB Return Codes" on page B-32.
- Return code values of X'30' through X'40' indicate user errors that are described in "NCB Return Codes" on page B-32. These return codes are only returned when you are using OS/2 EE 1.3.
- Return code values of X'41' through X'4F' indicate user errors that are described in "NCB Return Codes" on page B-32.
- Return code values of X'50' through X'FE' indicate a workstation error or an adapter error and are described in "NCB Return Codes" on page B-32.

NCB_LSN

Explanation: A 1-byte field indicating the local session number. This is the number of the session the application program has with another name on the network. This is valid only after an NCB.CALL or NCB.LISTEN command has been completed successfully. For send and receive commands under session support, this field must be provided. NETBIOS uses a modulo 254 technique to provide numbers from X'01' to X'FE'.

The RESET command uses this field. This field is not used for datagram support.

NCB_NUM

Explanation: A 1-byte number provided by NETBIOS after an NCB.ADD.NAME or NCB.ADD.GROUP.NAME command is executed. This number, not the name, must be used with all datagram support commands and for NCB.RECEIVE.ANY commands.

The number for NETBIOS_NAME_NUMBER_1 is always X'01'. NETBIOS uses a modulo 253 technique to provide numbers from X'02' to X'FE' for the remaining names.

The RESET command uses this field.

NCB_BUFFER@

Explanation: A 4-byte field containing the address of the buffer area assigned by the application program. This field is in define double-word (DD) format (segment:offset) and must be a valid address in workstation memory.

For NETBIOS using the OS/2 DD interface: The address specified in the NCB_BUFFER@ field is the physical address. For exceptions see specific commands (such as NCB.RESET).

NCB_LENGTH

Explanation: This 2-byte field indicates the length in bytes of the data buffer. For receive commands, the field is updated by NETBIOS to indicate the number of bytes actually received.

NCB_CALLNAME

Explanation: This is a 16-byte name of a device that the application program needs to communicate with. The name can either be on your adapter or any other adapter.

This field is also used by the NCB.RESET command.

For an NCB.CHAIN.SEND or NCB.CHAIN.SEND.NO.ACK command, the first 6 bytes are used to specify the second buffer. The first 2 bytes are the length of the buffer and the remaining 4 bytes are the address of the buffer in memory.

For NETBIOS using the OS/2 DD interface: The address specified in the NCB_CALLNAME field is a physical address.

NCB_NAME

Explanation: A name that the node is known by on the network. The name is 16 bytes long. The NETBIOS_NAME_NUMBER_1 can be used as a name. The NETBIOS_NAME_NUMBER_1 is 10 bytes of zeros followed by the 6 bytes of the NODE_ADDRESS.

For OS/2 EE 1.3: Also used by NCB.RESET for returned values.

NCB_RTO

Explanation: A 1-byte field used by the NCB.CALL and NCB.LISTEN commands to specify a timeout period for all receives associated with that session. The timeout value is specified in increments of 500 ms. If X'00' is specified, the default is no timeout. The timeout period can be different for each session, but is fixed when the session is established. The timeout period at the other end of the session can also be different.

NCB_STO

Explanation: A 1-byte field used by the NCB.CALL and NCB.LISTEN commands to specify a timeout period for all sends associated with that session. The timeout value is specified in increments of 500 ms. If X'00' is specified, the default is no timeout. The timeout period can be different for each session, but is fixed once the session is established. The timeout period at the other end of the session can also be different. Send timeouts should be used with caution because they always end the session if they expire.

NCB_POST@

Explanation: If this field is not zero, it is an indication to NETBIOS that the application program has a post routine that gets control when the NCB is completed.

The post routine should be as short in duration as possible. The register contents upon entry are listed below:

- AX = the NCB completion code (AH is always zero)
- ES and BX point to the NCB.

If the post address is all zeros, the post routine is not called by NETBIOS and the application program must check the return code field for a change from X'FF'.

For NETBIOS used with DOS: NCB_POST@ is a 4-byte address (segment:offset) that gets control. The application program returns control by issuing an "interrupt return" instruction.

For NETBIOS using the OS/2 DLR interface: NCB_POST@ is a 4-byte address (selector:offset) that gets control. The application program returns control by issuing a "return" instruction (return from a "call").

For NETBIOS using the OS/2 DD interface: NCB_POST@ is a 2-byte number. NETBIOS gives control to the post routine by issuing a Call Far instruction with the value X'0002' pushed on the stack and with the value of NCB_POST@ in register DI. Register DS contains the application program's device driver data segment. Note that the Call Far instruction is made to the application program's device driver entry point. Control is returned to NETBIOS by issuing a "Return Far 2" instruction (to account for the X'0002' that was pushed on the stack).

NCB_DD_ID

Explanation: The identification number of the device driver.

This field is supplied to the device driver application program by NETBIOS when the first NCB.RESET is completed. The application program must provide this value in all subsequent commands.

NCB_ADPTR_NUM

Explanation: Defines which adapter is to be used. For the Local Area Network Support Program and OS/2 EE 1.3, use X'00' for the primary adapter and X'01' for the alternate adapter. For LAPS, this field can be set to numbers in the range X'00'–X'0F', but only four adapters are supported. The adapter must have the corresponding (primary/alternate) switch set correctly. Other values are reserved.

NCB_CMD_CMPL

Explanation: This value is the same as the NCB_RETCODE.

NCB_RESERVE

Explanation: A 14-byte reserved field. Used as a work area by NETBIOS. Also contains certain system information when an NCB is completed with either a X'4X' or X'FX' return code.

Table 4-2. NCB_RESERVE for the Local Area Network Support Program

Offset from NCB_RESERVE	**Length in Bytes**	**Meaning**
0	2	Value of last adapter bring-up code
2	2	Return code of last CCB open issued
4	2	Last network status
6	2	Last adapter check status
8	2	Last PC error code
10	1	Last CCB code generated by NETBIOS during NCB.RESET or initialization
11	1	Return code of CCB in offset 10

Table 4-3. NCB_RESERVE for OS/2 EE

Offset from NCB_RESERVE	**Length in Bytes**	**Meaning**
0	2	Value of last adapter bring-up code
2	2	Return code of last CCB open issued
4	2	Last network status
6	2	Last adapter check status
8	2	OS/2 System Action Exception
10	2	Last PC error code
12	1	Last CCB code generated by NETBIOS during NCB.RESET or initialization
13	1	Return code of CCB in offset 12

NETBIOS Operational Parameters

There are three different methods of setting the NETBIOS operational parameters:

- Local Area Network Support Program old parameters
- Local Area Network Support Program parameters (see Appendix D, "The Local Area Network Support Program Interrupt Arbitrator")

  This allows the user to modify the NETBIOS operational parameters at load time by adding the parameters to the device driver command line.
- OS/2 NETBIOS 3.0 parameters (see Appendix E, "Operating System/2 Extended Edition Information").

  This allows the user to modify the NETBIOS operational parameters by changing the Communications Manager configuration parameters.

Local Area Network Support Program Old Parameters

This allows the user to modify the NETBIOS operational parameters by using the DIR.OPEN.ADAPTER command.

The parameters must be provided at open time and for each NCB.RESET command. If any of these parameters causes an error, the command terminates with a CCB_RETCODE or NCB_RETCODE as follows:

- If it is a DIR.OPEN.ADAPTER command, the CCB_RETCODE is X'10'. This can occur if the NCB_MAX_NAMES or NCB_MAX_SESSIONS values are not less than 255, or if there is insufficient work space available to satisfy the values of NCB_STATIONS, NCB_MAX_NAMES, NCB_MAX, and NCB_MAX_SESSIONS.
- If it is a NETBIOS command, the NCB_RETCODE is X'FC'. This can occur if any of the remaining parameters are found to be invalid when NETBIOS issues its DLC.OPEN.SAP command.

Table 4-4 (Page 1 of 2). Local Area Network Support Program Old Parameters

Offset	Parameter Name	Byte Length	8086 Type	Description
0		4	--	Adapter work area
4	NCB_TIMER_T1	1	DB	T1 value (response timer)
5	NCB_TIMER_T2	1	DB	T2 value (acknowledgment timer)
6	NCB_TIMER_TI	1	DB	Ti value (inactivity timer)
7	NCB_MAXOUT	1	DB	Maximum transmits without a receive acknowledgment
8	NCB_MAXIN	1	DB	Maximum receives without a transmit acknowledgment
9	NCB_MAXOUT_INCR	1	DB	Dynamic window increment value
10	NCB_MAX_RETRY	1	DB	N2 value
11		1	DB	Adapter work area
12		3		Reserved
15	NCB_ACCESS_PRI	1	DB	Ring access priority
16	NCB_STATIONS	1	DB	Maximum NETBIOS link stations
17		19	--	Adapter work area
36	NCB_MAX_NAMES	1	DB	Maximum names in names table
37	NCB_MAX	1	DB	Maximum outstanding NCBs
38	NCB_MAX_SESSIONS	1	DB	Maximum sessions
39		1	DB	Reserved
40		1	DB	Reserved
41	NCB_OPTIONS	1	DB	Various options
42	NCB_POOL_LENGTH	2	DW	Length of area at NCB_POOL_ADDRESS
44	NCB_POOL_ADDRESS	4	DD	Starting segment of message work area

Table 4-4 (Page 2 of 2). Local Area Network Support Program Old Parameters

Offset	Parameter Name	Byte Length	8086 Type	Description
48	NCB_TRANSMIT_TIMEOUT	1	DB	Time to wait for one query
49	NCB_TRANSMIT_COUNT	1	DB	Maximum number of times to transmit queries

NCB_TIMER_T1

Explanation: Specifies the time period between 1 and 10 used to determine an inoperative condition on a link. The time intervals are defined by the DIR.OPEN.ADAPTER command. If the value is zero, the default of 5 is used. See "NETBIOS DLC Timers" on page 4-14.

NCB_TIMER_T2

Explanation: Specifies the time period between 1 and 10 used to delay transmission of an acknowledgment for a received I-LPDU for link stations using this SAP. The time intervals are defined by the DIR.OPEN.ADAPTER command. If the value is zero, the default of 2 is used. If the value is greater than 10, the acknowledgment timer is not implemented and acknowledgments will be sent at the earliest opportunity. See "NETBIOS DLC Timers" on page 4-14.

NCB_TIMER_Ti

Explanation: Specifies the time period between 1 and 10 used to determine an inactive condition on a link. The time intervals are defined by the DIR.OPEN.ADAPTER command. If the value is zero, the default of 3 is used. See "NETBIOS DLC Timers" on page 4-14.

NCB_MAXOUT

Explanation: Specifies the maximum number of sequentially numbered transmitted I-LPDUs that a link station on NETBIOS SAP can have pending at one time. The maximum valid value is 127. If the value is zero, the default of 2 is used.

NCB_MAXIN

Explanation: Specifies the maximum number of sequentially numbered received I-LPDUs that a link station on NETBIOS SAP can receive before sending an acknowledgment. The maximum valid value is 127. If the value is zero, the default of 1 is used.

NCB_MAXOUT_INCR

Explanation: This dynamic window increment value is used to reduce bridge congestion. If the two end points of a session are on different rings, and the adapter detects an error condition requiring retransmission, the MAXOUT counter is set to 1. It is then incremented by 1 each time MAXOUT_INCR frames are acknowledged by the remote station, until it reaches the value of this field. If this field is set to a value of zero, the default of 1 is used. For more details, refer to *IBM Token-Ring Network Architecture Reference.*

NCB_MAX_RETRY

Explanation: Specifies the number of retries for an unacknowledged command LPDU, or in the case of an I-LPDU timeout, the number of times that the non-responding remote link station will be polled with an RR/RNR command LPDU. The maximum valid value is 255. If the value is zero, the default of 8 is used.

NCB_ACCESS_PRI

Explanation: The transmit access priority value to be placed in the AC byte of all transmissions from the link station and SAP. The format is B'nnn00000', where 'nnn' is the access priority value. No checking is done and the low-order 5 bits are ignored. If the access priority is higher than allowed for the adapter, the error will be detected on the first transmission.

NCB_STATIONS

Explanation: The number of link stations that can be active at one time. This value must not exceed the value of DLC_MAX_STATIONS. During execution of an NCB.RESET command, this value must not exceed the current number of available link stations. If the value is zero, the default of 6 is used.

NCB_MAX_NAMES

Explanation: The maximum number of names that can be in the name table. The adapter itself is entered as a name, using one of the positions. The maximum valid value is 254. If the value is zero, the default of 17 is used.

NCB_MAX

Explanation: The number of NCBs that can be pending at one time.

Note: This value indicates the number of "no-wait" and "wait" commands that can be issued. The maximum valid value is 255. If the value is zero, the default of 12 is used.

NCB_MAX_SESSIONS

Explanation: The maximum number of sessions that can be active at one time. The maximum valid value is 254. If the value is less than the value of NCB_STATIONS, the NCB_STATIONS value is used.

NCB_OPTIONS

Explanation: Various options that are each represented by a bit. If the bit has a value of B'1', the option is active. The high-order bit (bit 7) is the leftmost bit. The bits are described below:

Bits	**Description**
Bit 7	Auto open. When received by the adapter system interface, bit 7 causes an NCB.RESET command to close and then open the adapter. If bit 7 is zero, the NCB.RESET command does not perform the close and ignores the fields containing the "number of sessions" and the "number of commands." (They remain as defined in the DIR.OPEN.ADAPTER command.)
Bit 6	Reserved.
Bit 5	This ring only. When this bit is set on, NETBIOS assumes that all nodes are on the same ring.
Bits 4-0	Reserved.

NCB_POOL_LENGTH

Explanation: The number of bytes of system memory assigned by the application program for the NETBIOS work area pool. If the value is zero, NETBIOS internal work area in system memory as defined at load time is used.

NCB_POOL_ADDRESS

Explanation: The starting address of the NETBIOS work-area pool for the adapter support software to build tables, buffers, and control blocks. If the NCB_POOL_LENGTH value is zero, this parameter is ignored.

NCB_TRANSMIT_TIMEOUT

Explanation: A value to define the amount of time that NETBIOS will wait for a response to a private query on the network (such as an "add name query"). The value is in half-second increments. A value of 10 represents a time of 5 seconds. If the value is zero, the default of 1 (1/2 second) is used. If the value is greater than 20, 20 (10 seconds) is used.

NCB_TRANSMIT_COUNT

Explanation: A value to define the number of times that private network queries, such as an "add name query," will be transmitted for a given command. If a query is transmitted more than one time, the next one is transmitted after the NCB_TRANSMIT_TIMEOUT expires. If the value is zero, the default of 6 is used. If the value is greater than 10, 10 is used.

NETBIOS Notes: The following information is specific to NETBIOS versions lower than NETBIOS 3.0.

The number of stations determines how many physical nodes the adapter can establish connection with at one time.

The number of sessions determines how many different NETBIOS sessions can be active at one time.

There can be multiple sessions on one station.

NETBIOS provides internal work area in system memory as defined at load time for buffers and work area. The application program can define its own space by properly coding the NCB_POOL_ADDRESS and NCB_POOL_LENGTH parameters.

To calculate whether additional work space in bytes is needed, use the following formula by adding the subtotals to 1900:

```
Maximum names (NCB_MAX_NAMES) x 20 =
  Maximum stations (NCB_STATIONS) x 40 =
 Maximum sessions (NCB_SESSIONS) x 44 =
    Maximum commands (NCB_MAX) x 100 =
            Number of transmit buffers x 100 =
                                            +   1900
                                                ------
                                        Total =
```

The remainder of the defined work area is configured as transmit buffers and there must be room for at least 10.

All tables and buffers, except the receive buffers, must be in the same segment. Therefore, if these tables use more than the defined work area, the area defined by the application program must be large enough to hold all this information.

The receive buffers are allocated by the adapter from the defined work area and any application program assigned memory after all other memory requirements are complete.

Local Area Network Support Program New Parameters

The new parameters are added to the NETBIOS driver command line. When the NETBIOS driver is loaded, the NETBIOS parameters become operational. Refer to the *LAN Support Program User's Guide* for a description of these parameters.

NETBIOS 3.0 (OS/2 EE)

This allows the user to modify the NETBIOS operational parameters with the Communications Manager configuration. See Appendix E, "Operating System/2 Extended Edition Information," for a complete description of these parameters.

NETBIOS 4.0

For complete information about the parameters for NETBIOS 4.0, refer to the *NTS/2 LAN Adapter and Protocol Support Configuration Guide.*

NETBIOS DLC Timers

The LAPS implementation of NETBIOS allows the user to specify timer values directly in milliseconds. Timer values in previous releases must be calculated using tables. The following directory indicates the timer values (in seconds) when the various parameter values are used.

These values are using the default DLC tick values.

DLC.T1		**DLC.T2**		**DLC.TI**	
Parameter	**Seconds**	**Parameter**	**Seconds**	**Parameter**	**Seconds**
1	.20-.40	1	.04-.08	1	1-2
2	.40-.60	2	.08-.12*	2	2-3
3	.60-.80	3	.12-.16	3	3-4*
4	.80-1.0	4	.16-.20	4	4-5
5	1.0-1.2*	5	.20-.40	5	5-6
6	1-2	6	.40-.80	6	5-10
7	2-3	7	.80-1.2	7	10-15
8	3-4	8	1.2-1.6	8	15-20
9	4-5	9	1.6-2.0	9	20-25
10	5-6	10	2.0-2.4	10	25-30
		11	-disabled		

* This value is the default.

NETBIOS Calling Conventions Using the Local Area Network Support Program

An application program using the Local Area Network Support Program should use the following conventions to initiate an NCB:

- Register ES and BX point to the NCB.
- Execute instruction interrupt X'5C'.

NETBIOS Calling Conventions Using the Dynamic Link Routine Interface

An application program using the OS/2 DLR interface should use the following conventions to initiate an NCB:

- Push the NCB's selector value onto the stack.
- Push the NCB's offset value onto the stack.
- Use the Call Far instruction to call NETBIOS.

NETBIOS Calling Conventions Using the Device Driver Interface

An application program using the OS/2 DD interface should use the following conventions to initiate an NCB:

- Registers ES and BX point to the NCB.
- Register DS is the value returned on the ATTACHDD call to the NETBIOS device driver.
- Push the value X'0000' onto the stack.
- Use the Call Far instruction for the address obtained when the application program executed an OS/2 'ATTACHDD' to name NETBIOS$.

Note: When you are using the OS/2 DD interface NETBIOS, the NCB and any buffers must be locked by the application program.

NCB Completion with Wait Type Commands

If the command issued is a wait type command, control is not returned to the next instruction until the adapter has completed the command. When the command does complete, the return code will be in both the AL register and the NCB_RETCODE field.

NCB Completion with No-Wait Type Commands

If the command is a no-wait type, NETBIOS presents two return codes. An immediate return code is posted to the AL register. If the immediate return code in the AL register is X'00', processing is complete. An appendage will be enabled to run if it is present. If the immediate return code is X'FF', processing is continuing; control will be passed to an appendage when processing is complete, if an appendage is present. If the immediate return code is neither X'00' nor X'FF', processing is complete, but an error has occurred. The appendage, if present, will not be activated.

If the NCB_POST@ field of the NCB is zero, the final return code is placed in the NCB_CMD_CMPL field, which must be checked for change by the application program. If the application is checking the NCB_CMD_CMPL field, a change of value from X'FF' indicates command completion. This value is the final return code.

If the NCB_POST@ field is not zero, NETBIOS gives control to the post routine after setting the final return code in both the AL register and the NCB_RETCODE field. The application program continues with the instruction located by the contents of the NCB_POST@ field (the command completion appendage).

NETBIOS Command Descriptions

Each command description in this section begins with a box containing the command name. The hexadecimal number at the top of the box is the command code value. Both the wait and no-wait values are supplied when applicable.

NCB.ADD.GROUP.NAME

Hex 36 Wait B6 No-Wait

NCB.ADD.GROUP.NAME

Command Description: This command adds a 16-character name to the table of names. The name cannot be used by any other station across the network as a unique name, but can be added by any station as a group name. This is a name by which this station will be known.

Command Specifics: When NETBIOS processes this command, it sends name query requests on the network. If no reply is received, it is assumed that no other node on the network has defined the name as a unique name, and the name is added to that node. The adapter returns the number of the name in the NCB_NUM field. This number is used in datagram support and for NCB.RECEIVE.ANY commands.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_NAME
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface).

Returned Fields:

NCB_RETCODE
NCB_NUM
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.ADD.NAME

Hex 30 Wait B0 No-Wait

NCB.ADD.NAME

Command Description: This command adds a 16-character name to the table of names. The name must be unique across the network. This is a name that this station will be known by. The first byte of the name parameter, NCB_NAME, cannot be X'00' or *.

Command Specifics: When NETBIOS processes this command, it sends name query requests on the network. If no reply is received, the name is assumed to be unique and is added to the table of names. The adapter returns the number of the name in the NCB_NUM field. This number is used in datagram support and for NCB.RECEIVE.ANY commands.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_NAME
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface).

Returned Fields:

NCB_RETCODE
NCB_NUM
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.CALL

Hex 10 Wait 90 No-Wait

NCB.CALL

Command Description: This command opens a session with another name specified by the NCB_CALLNAME field using the local name specified by the supplied NCB_NAME field.

Command Specifics: The destination name station must have an NCB.LISTEN command pending in order for the session to be established. Sessions can be established with either a local or remote name. Multiple sessions can be established with the same pair of names. The timeout intervals are specified in 500-ms units. (A value of zero means that no timeout will occur.) The system timeout intervals and retry count are constants in NETBIOS. The NCB.CALL command itself aborts if unsuccessful after the system timeout interval (retry count exhausted). When the call is completed, a local session number (LSN) is assigned and is used thereafter to refer to the established session.

Local session numbers (NCB_LSN) are assigned in a round-robin technique, starting from the next available value within the range of 1 to 254.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_NAME
NCB_CALLNAME
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_RTO (500-ms increments. If the field is set at X'00', no receive timeout will occur.)
NCB_STO (500-ms increments. If the field is set at X'00', no send timeout will occur.).

Returned Fields:

NCB_RETCODE
NCB_LSN
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.CANCEL

Hex 35 Wait

NCB.CANCEL

Command Description: The active command has its NCB at the address given by NCB_BUFFER. The NCB.CANCEL command requests that this active command be canceled. NCB.CANCEL can be issued by an application program even though the total number of NCBs allocated to the application program are in use.

Command Specifics: Use caution when canceling session commands. Cancellation always ends the session.

The following commands can be canceled:

- NCB.CALL
- NCB.CHAIN.SEND
- NCB.CHAIN.SEND.NO.ACK
- NCB.HANG.UP
- NCB.LAN.STATUS.ALERT
- NCB.LISTEN
- NCB.RECEIVE
- NCB.RECEIVE.ANY
- NCB.RECEIVE.BROADCAST.DATAGRAM
- NCB.RECEIVE.DATAGRAM

- NCB.SEND
- NCB.SEND.NO.ACK
- NCB.STATUS.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_BUFFER@ (address of the NCB to be canceled)
NCB_DD_ID (if using the OS/2 DD interface).

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'5X' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.CHAIN.SEND

Hex 17 Wait 97 No-Wait

NCB.CHAIN.SEND

Command Description: This command sends data to the session partner as defined by the session number in the NCB_LSN field. The data to send is in the buffers pointed to by the NCB_BUFFER@ field and the NCB_CALLNAME field.

Command Specifics: The data in the second buffer is concatenated to the data in the first buffer and sent as a single message. The NCB.CALLNAME field is used to specify the length and address of the second buffer. The length is specified in the first 2 bytes and the second buffer address in the next 4 bytes.

When the remote side closes a session, all NCB.CHAIN.SEND commands pending for the closed session are returned with a "session closed" status. If a local NCB.HANG.UP command is issued with any pending NCB.CHAIN.SEND commands, the NCB.CHAIN.SEND commands are completed.

If a session is aborted, a "session ended abnormally" status is returned. If the NCB.CHAIN.SEND timeout expires, the session is aborted and a "command timed out" status is returned. Timeout values for the NCB.CHAIN.SEND are associated with the session when an NCB.CALL or NCB.LISTEN is issued and cannot be specified with this command.

Message size must be between 0 and 131,070 bytes long.

If more than one NCB.SEND or NCB.CHAIN.SEND is pending, the data is transmitted in a first-in, first-out order within a session.

If the NCB.CHAIN.SEND command cannot be completed for any reason, the session ends abnormally and the session is dropped. This is done to guarantee data integrity.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_LSN

NCB_CALLNAME (specifies the second buffer)
- Bytes 0 - 1 = NCB_LENGTH2
- Bytes 2 - 5 = NCB_BUFFER2@.

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.CHAIN.SEND.NO.ACK

Hex 72 Wait F2 No-Wait

NCB.CHAIN.SEND.NO.ACK

Command Description: This command has the same benefits as the NCB.SEND.NO.ACK in that it does not require an acknowledgment.

This command sends data to the session partner as defined by the session number in the NCB_LSN field. The data to send is in the buffer pointed to by the NCB_BUFFER@ field. Two buffers can be chained together with this command.

The data in the second buffer is concatenated to the data in the first buffer and sent as a single message. The NCB_CALLNAME field specifies the length and address of the second buffer. The length is specified in the first 2 bytes and the second buffer address in the next 4 bytes.

When the remote side closes a session, all NCB.CHAIN.SEND.NO.ACK commands pending for the closed session are returned with a "session closed" status. If a local NCB.HANG.UP command is issued with any pending NCB.CHAIN.SEND.NO.ACK commands, the NCB.CHAIN.SEND.NO.ACK commands are completed.

If a session is aborted, a "session ended abnormally" status is returned.

Message size must be between 0 and 65,535 bytes long.

If more than one send type of command is pending, the data is transmitted in a first-in, first-out order within a session.

Command Specifics: Lost data (indicated by a X'07' return code), as well as the possible need for retransmission, are both the responsibility of the application program.

If the remote node cannot process the NCB.CHAIN.SEND.NO.ACK, it is handled as an NCB.CHAIN.SEND; a DATA_ACK must be received before the NCB.CHAIN.SEND.NO.ACK will be executed.

This command should be used for transactions and not for data transfers.

Notes:

1. Note the following on a X'07' return code.
   - The return code is generated for the node that issued NCB.CHAIN.SEND.NO.ACK.
   - The command terminating with the X'07' must be reissued.

- The return code is issued if the receiver either had no NCB.RECEIVE or NCB.RECEIVE.ANY up, or the transmitted data exceeded the length of the receive buffer.
- Unless the user is aware of the number of NCB.CHAIN.SEND.NO.ACKs issued, he has no indication of how many were not completed successfully.
- All other NCB.CHAIN.SEND return codes apply.

2. For data transmitted with NCB.CHAIN.SEND.NO.ACK, only one receive command will be satisfied. The NCB.CHAIN.SEND.NO.ACK data buffer must be no longer than the remote receive buffer for successful completion.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_LSN
NCB_CALLNAME (specifies the second buffer)
- Bytes 0 - 1 = NCB_LENGTH2 DW format
- Bytes 2 - 5 = NCB_BUFFER2@ DD format.

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.DELETE.NAME

Hex 31 Wait B1 No-Wait

NCB.DELETE.NAME

Command Description: This command deletes a 16-character name from the table of names.

Command Specifics: If the name has active sessions when the NCB.DELETE.NAME command is issued, the name is flagged as *deregistered* and the "command completed, name has active sessions" status is returned to the user. The delete is delayed until the sessions associated with the name are closed. A deregistered name is not usable by subsequent NCBs.

If the name has only pending non-active session commands when the NCB.DELETE.NAME command is issued, the name is removed and the "command completed" status is returned to the user. The pending non-active session commands are terminated immediately with the "name was deleted" status. Non-active session commands are:

- NCB.LISTEN
- NCB.RECEIVE.ANY
- NCB.RECEIVE.DATAGRAM
- NCB.RECEIVE.BROADCAST.DATAGRAM.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_NAME
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface).

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.FIND.NAME

Hex 78 Wait F8 No-Wait

NCB.FIND.NAME

Command Description: This command finds the location on the network of a 16-character name. The name is specified in the NCB_CALLNAME field.

Command Specifics: NETBIOS sends a name query request on the network. If any remote nodes have the requested name registered, they respond with an indication of how they have the name registered (unique/group).

If no response is received within the system timeout period, NETBIOS returns a CCB_RETCODE of X'05'. If responses are received, NETBIOS returns the number of nodes that responded, followed by every unique LAN header from each responding node. The LAN header contains the adapter address of the remote node where the name is located. The returned data is located at the buffer address specified by the NCB_BUFFER@ field, and the NCB_LENGTH indicates the number of bytes of data stored. There is no guarantee that all nodes will respond within the given timeout period.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_CALLNAME.

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs)
NCB_LENGTH.

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

Table 4-5. Data Areas Returned for the NCB.FIND.NAME Command

Offset	Byte Length	8086 Type	Destination
0	2	DW	Number of nodes responding (will be greater than 1 only if used as a group name)
2	1	DB	Reserved
3	1	DB	Status: **X'00'** unique name **X'01'** group name
4	*	DB	The LAN header length and the LAN header from the remote nodes where the name is located (contains the address of the remote node). Each entry includes the header length and the LAN header for a total of 33 bytes for each entry. See the example of a LAN header below.

Length	Access Control	Frame Control	Dest. Addr.	Source Addr.	Routing Info
0	1	2	3◄──►8	9◄──►14	15◄──►32

Refer to Figures 1-3 and 1-7 starting on page 1-11 for a description of the header data.

NCB.HANG.UP

Hex 12 Wait 92 No-Wait

NCB.HANG.UP

Command Description: This command closes the session specified by the local session number.

Command Specifics: When an NCB.HANG.UP command is issued to the adapter, all pending (local) NCB.RECEIVE commands are terminated and returned to the issuer with "session closed" in the NCB_RETCODE field. The termination is valid whether or not any data had been transferred by the pending command. If a local NCB.SEND command is pending, the NCB.HANG.UP command waits for up to 20 seconds for a pending SEND to completed. This delay is true whether or not the command has begun to transfer data, or is waiting for the remote side to issue an NCB.RECEIVE command. The NCB.HANG.UP is performed when any of the following conditions occur:

- The NCB.SEND is completed.
- The NCB.SEND has aborted.
- The NCB.SEND fails because the session was terminated by the other side with an NCB.HANG.UP.
- The NCB.SEND fails because of the timeout specified when the session was opened.

If one of the above conditions does not occur within the system timeout period after the NCB.HANG.UP command is issued, the NCB.HANG.UP command is returned with a "command timed out" status and the session is aborted.

When a session closes, all NCB.SEND and NCB.RECEIVE commands pending on the closed session are returned to the user with a "session closed" status. If an NCB.RECEIVE.ANY command is pending on the local name used by the session, it is returned to the issuer with a "session closed" status.

Only a single NCB.RECEIVE.ANY command will be returned even though many NCB.RECEIVE.ANY commands are pending. Even though a single NCB.RECEIVE.ANY command is returned, many NCB.SENDs or NCB.RECEIVEs can be returned when pending.

When a session is abnormally terminated, all outstanding commands on that session will be returned to the issuer with a "session ended abnormally" status. When one side of a NETBIOS session issues an NCB.HANG.UP command, the remote partner does not free the session resources until at least one command completes, indicating the session is closed.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LSN
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface).

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.LAN.STATUS.ALERT

Hex F3 No-Wait

NCB.LAN.STATUS.ALERT

Command Description: This command is used by application programs that should be notified of temporary ring error conditions that last for more than one minute.

Command Specifics: There are two conditions where this command is completed:

- There is no temporary ring error or the current ring error has existed for less than one minute. In this case, any NCB.LAN.STATUS.ALERT commands completed will be queued by NETBIOS.
- The current ring error condition has existed for more than one minute. In this case all queued NCB.LAN.STATUS.ALERT commands are completed with a good return code. Any commands issued while this condition exists are completed with a good return code.

Note: The queued NCB.LAN.STATUS.ALERT commands can be canceled with the CANCEL command.

Supplied Fields:

NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_COMMAND
NCB_ADPTR_NUM.

Returned Fields:

NCB_RETCODE.

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.LISTEN

Hex 11 Wait 91 No-Wait

NCB.LISTEN

Command Description: This command enables a session to be opened with the name specified in the NCB_CALLNAME field, using the name specified by the NCB_NAME field.

Command Specifics: If the NCB_CALLNAME field has an asterisk (*), a session is established with any network node that issues an NCB.CALL to the local name.

NCB.LISTEN for a specific name has priority over an NCB.LISTEN for any name. Sessions can be established with either a local or remote name. Multiple sessions can be established with the same pair of names.

The timeout intervals are specified in 500-ms units. (A value of zero means that no timeout will occur.) An NCB.LISTEN command will not time out, but an NCB.LISTEN occupies a session entry and is considered a pending session in information returned in an NCB.STATUS command. Local session numbers (LSN) are assigned in a round-robin technique starting with the next available value within the range from 1 to 254. Also, if an asterisk (*) is used for the called name, the name that made the call is returned in the NCB_CALLNAME field.

The error "name conflict detected" is returned if, during the completion for an NCB.LISTEN command, a name exists in more than one table. All nodes with the name registered, except the one where the NCB.LISTEN command has returned successfully, will report the "name conflict detected" error.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)

NCB_NAME

NCB_CALLNAME (This can be specified in the first byte as an asterisk (*). The asterisk is used to listen for a call from any name to the local name. If a name is specified in this field, it takes priority over a name of an asterisk.)

NCB_POST@ (if the no-wait option is used)

NCB_DD_ID (if using the OS/2 DD interface)

NCB_RTO (500-ms increments. If the field is set at X'00', no receive timeout will occur.)

NCB_STO (500-ms increments. If the field is set at X'00', no send timeout will occur.).

Returned Fields:

NCB_RETCODE

NCB_LSN

NCB_CALLNAME (if NCB.LISTEN for any name is used, specified with an asterisk)

NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.RECEIVE

Hex 15 Wait 95 No-Wait

NCB.RECEIVE

Command Description: This command receives data from its session partner that has issued one of the following commands:

- NCB.CHAIN.SEND
- NCB.CHAIN.SEND.NO.ACK
- NCB.SEND
- NCB.SEND.NO.ACK.

Command Specifics: If more than one RECEIVE type of command is pending, the NCB.RECEIVE command has priority over the NCB.RECEIVE.ANY command. NCB.RECEIVE commands are processed in a first-in, first-out order. Within each priority, the commands are processed in a first-in, first-out order. Timeout values are specified during an NCB.CALL or NCB.LISTEN and cannot be specified with this command.

When a session is closed, either by a local session close command or by the remote side closing the session, all pending NCBs for that session are returned with a "session closed" status.

A return code of X'06' is posted in the NCB_RETCODE field if the receive buffer is not large enough for the message being received. Another receive can be issued to obtain the rest of the information before an NCB.SEND timeout occurs.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LSN
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_LENGTH.

Returned Fields:

NCB_RETCODE
NCB_LENGTH
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.RECEIVE.ANY

Hex 16 Wait 96 No-Wait

NCB.RECEIVE.ANY

Command Description: This command receives data for any session using the name defined in NCB_NUM. The data is sent from one of the following commands:

- NCB.CHAIN.SEND
- NCB.CHAIN.SEND.NO.ACK
- NCB.SEND
- NCB.SEND.NO.ACK.

You must use the name number instead of the name when issuing this command.

Command Specifics: If more than one RECEIVE type of command is pending, the NCB.RECEIVE command has priority over the NCB.RECEIVE.ANY command. Within each priority, the commands are processed in a first-in, first-out order.

If the NCB_NUM field is set to X'FF' by the application program, then the receive is for any remote name that you have a session with, for any of your names. NETBIOS sets the NCB_NUM field to the number of the name for which the data was received when the return code is presented.

When a session is closed by

1. a local session close command
2. a remote side closing the session, or
3. a session abort.

Only one NCB.RECEIVE.ANY or NCB.RECEIVE name will be posted with "session closed" or "session aborted," regardless of the number of session receives that are pending. An NCB.RECEIVE.ANY with no name specified will post only if there is no pending NCB.RECEIVE.ANY name for the session that closed.

A return code of X'06' is posted in the NCB_RETCODE field if the receive buffer is not large enough for the message being received. Another receive can be issued to obtain the rest of the information before a timeout occurs.

When using DOS application programs, use "NCB.RECEIVE.ANY to any name" with caution as this command can receive messages for other programs running in the Personal Computer.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_NUM.

Returned Fields:

NCB_RETCODE
NCB_LSN
NCB_LENGTH
NCB_RESERVE (if error X'4X' or X'FX' occurs)
NCB_NUM (if X'FF' was specified in this field when issued).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.RECEIVE.BROADCAST.DATAGRAM

Hex 23 Wait A3 No-Wait

NCB.RECEIVE.BROADCAST.DATAGRAM

Command Description: This command receives a datagram message from any name on the network that issues an NCB.SEND.BROADCAST.DATAGRAM command.

Command Specifics: A "message incomplete" status is returned if the receive buffer is not large enough for the data being received. The remaining data is lost at this point.

When a broadcast datagram is received, all broadcast datagrams are completed.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_NUM.

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).
NCB_LENGTH
NCB_CALLNAME.

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.RECEIVE.DATAGRAM

Hex 21 Wait A1 No-Wait

NCB.RECEIVE.DATAGRAM

Command Description: This command receives a datagram message from any name on the network that issues an NCB.SEND.DATAGRAM command.

Command Specifics: A "message incomplete" status is returned if the receive buffer is not large enough for the data being received. The remaining data is lost at this point.

This command will not receive a broadcast datagram.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)

NCB_NUM (if X'FF', then receive a datagram from any name on the network for any name in the local table).

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs)
NCB_LENGTH
NCB_CALLNAME
NCB_NUM.

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.RESET

There are four basic environments in NETBIOS:

- Local Area Network Support Program using "old" load NETBIOS parameters
- Local Area Network Support Program using "new" load NETBIOS parameters
- OS/2 EE application programs using the DLR interface
- OS/2 EE application programs using the DD interface.

The RESET command operates differently in each environment. The following sections describe the RESET operation.

Hex 32 Wait

NCB.RESET with the Local Area Network Support Program Using Old NETBIOS Parameters

Command Description: When RESET is issued, the following occurs:

- All current NETBIOS names are deleted, all current sessions are aborted, and all pending NCBs are purged.
- If indicated in the last DIR.OPEN.ADAPTER command, the adapter will be closed.
- If the adapter is closed, it will be re-opened.

Command Specifics: If NETBIOS opens the adapter, the adapter will be closed on each RESET. If the application program opens the adapter, it can choose to keep from closing the adapter. See "DIR.OPEN.ADAPTER" on page 3-17.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LSN (the number of sessions. If 0, the default is 6.)
NCB_NUM (the number of NCBs. If 0, the default is 12.).

Returned Fields:

NCB_RETCODE
NCB_RESERVE.

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

Note: It is preferable to use the NCB.HANGUP command rather than the NCB.RESET command when terminating existing NETBIOS sessions.

Hex 32 Wait

NCB.RESET with the Local Area Network Support Program Using New NETBIOS Parameters

Command Description: When RESET is issued, the following occurs:

- All current NETBIOS names are deleted, all current sessions are aborted, and all pending NCBs are purged.
- If the load parameter CLOSE.ON.RESET is coded as YES, the adapter is closed and then re-opened. (The default is NO.)

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LSN (the number of sessions)
- The maximum value is as defined by load parameters.
- If coded as 0, the default is one of the following:
  - 6, if the load parameter RESET.VALUES=NO (default)
  - As defined by the load parameter SESSIONS, if the load parameter RESET.VALUES=YES.

NCB_NUM (the number of NCBs)
- The maximum value is as defined by the load parameters.
- If coded as 0, the default is one of the following:
  - 12, if the load parameter RESET.VALUES=NO (default)
  - As defined by the load parameter COMMANDS, if the load parameter RESET.VALUES=YES.

Returned Fields:

NCB_RETCODE
NCB_RESERVE.

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

Note: It is preferable to use the NCB.HANGUP command rather than the NCB.RESET command when terminating existing NETBIOS sessions.

Hex 32 Wait

NCB.RESET—OS/2 Dynamic Link Routine Interface

Command Description: NCB.RESET can be issued by an application program even though the total number of NCBs allocated to the application program are in use. An application program executing in this environment must always issue a RESET under the following conditions:

- As the first NCB issued by the application program
- To recover from either a X'4X' or X'FX' return code.

The OS/2 environment differs from the DOS environment in that RESET applies only to the application program and not to the entire workstation. An application program issues RESET to allocate resources for itself from a pool of NETBIOS resources.

When RESET is issued, only resources applicable to that application program are given back to NETBIOS. For that application program, all current NETBIOS names are deleted, all current sessions are aborted, and all outstanding NCBs are purged.

After resources are freed, new resources are acquired for the application program, according to the RESET request.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_COMMAND
NCB_LSN (determines whether the RESET is requesting or freeing resources.)

- If NCB_LSN is X'00', it indicates a request for resources, according to the parameters in NCB_CALLNAME.
- If NCB_LSN is not X'00', it indicates that all resources associated with the environment are to be freed.
  - NCB_CALLNAME is ignored.
  - NCB_NAME has the normal return values set (see below).

NCB_CALLNAME is the resource request field. Any NCB_CALLNAME areas not defined here are reserved and should be zero.

If this is a request to free resources (NCB_LSN is not X'00'), NCB_CALLNAME is ignored.

- REQ_SESSIONS at NCB_CALLNAME+0 (1-byte field):
  - The number of sessions requested by the application program.
  - If zero, the default of 16 is used.
- REQ_COMMANDS at NCB_CALLNAME+1 (1-byte field):
  - The number of commands requested by the application program.
  - If zero, the default of 16 is used.
- REQ_NAMES at NCB_CALLNAME+2 (1-byte field):
  - The number of names requested by the application program. This does not include a reservation for NAME_NUMBER_1.
  - If zero, the default of 8 is used.
- REQ_NAME_ONE at NCB_CALLNAME+3 (1-byte field):
  - A request to reserve NAME_NUMBER_1 for this application program.
  - If 0, NAME_NUMBER_1 is not requested.
  - If not 0, NAME_NUMBER_1 is desired to be reserved for this application program.

Returned Fields: NCB_NAME is used to return resource information. Areas of NCB_NAME not defined here are reserved and set to zero.

The following returned values are resources actually obtained by the application program:

- ACT_SESSIONS at NCB_NAME+0 (1-byte field):

  The number of sessions obtained by the application program.
- ACT_COMMANDS at NCB_NAME+1 (1-byte field):

  The number of commands obtained by the application program.
- ACT_NAMES at NCB_NAME+2 (1-byte field):

  The number of names obtained by the application program does not include NAME_NUMBER_1.

- ACT_NAME_ONE at NCB_NAME+3 (1-byte field):
  - If zero, the application program does not have use of NAME_NUMBER_1.
  - If one, the application program has use of NAME_NUMBER_1.

Returned values are resources defined at NETBIOS load time. These values represent absolute maximums. The sum of all application program requests cannot exceed these values.

- LOAD_SESSIONS at NCB_NAME+8 (1-byte field):

  The number of sessions defined at NETBIOS load time.
- LOAD_COMMANDS at NCB_NAME+9 (1-byte field):

  The number of commands defined at NETBIOS load time.
- LOAD_NAMES at NCB_NAME+10 (1-byte field):

  The number of names defined at NETBIOS load time. This number includes NAME_NUMBER_1.
- LOAD_STATIONS at NCB_NAME+11 (1-byte field):

  The actual number of link stations obtained by NETBIOS.

  This cannot be the number requested at load time. NETBIOS will obtain as many link stations as possible of the number requested.
- LOAD_REMOTE_NAMES at NCB_NAME+14 (1-byte field):

  The number of remote names (RND) defined at NETBIOS load time.
- NCB_NAME+15 (1-byte field) is reserved.

NCB_RESERVE is used as in the Local Area Network Support Program.

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

Hex 32 Wait

NCB.RESET—OS/2 Device Driver Interface

Command Description: NCB.RESET can be issued by an application program even though the total number of NCBs allocated to the application program are in use. An application program executing in this environment must always issue a RESET under the following conditions:

- As the first NCB issued by the application program
- To recover from either an X'4X' or X'FX' return code.

The OS/2 environment differs from the DOS environment in that RESET applies only to the application program and not to the entire workstation. An application program issues RESET to allocate resources for itself from a pool of NETBIOS resources.

When RESET is issued, only resources applicable to that application program are given back to NETBIOS. For that application program, all current NETBIOS names are deleted, all current sessions are aborted, and all pending NCBs are purged. After resources are freed, new resources are acquired for the application program, according to the RESET request.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)

NCB_COMMAND: Same as OS/2 using the DLR interface.

NCB_LSN: Same as OS/2 using the DLR interface.

NCB_BUFFER@: This field must contain the address (selector: offset) of the application device driver name ASCIIZ string if this is the first RESET issued by the application program.

NCB_CALLNAME: Same as OS/2 using the DLR interface.

NCB_DD_ID:

- If this is the first RESET issued by the application program for this adapter, the value must be X'0000'.
- Subsequent NCB.RESET commands issued by an application program must supply this field, as returned on the first NCB.RESET for this adapter.

Returned Fields:

NCB_NAME: Same as OS/2 using the DLR interface.

NCB_DD_ID:

- This value must be placed in all subsequent NCBs issued by this application program

NCB_RESERVE: Same as OS/2 using the DLR interface.

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.SEND

Hex 14 Wait 94 No-Wait

NCB.SEND

Command Description: This command sends data to the session partner as defined by the session number in the NCB_LSN field. The data to send is in the buffer pointed to by the NCB_BUFFER@ field.

Command Specifics: If the NCB.SEND timeout expires, the session is aborted and a "command timed out" status is returned. Timeout values for the NCB.SEND are associated with the session when an NCB.CALL or NCB.LISTEN was issued and cannot be specified with this command.

Message size must be between 0 and 65,535 bytes long.

If more than one session SEND type of command is pending, the data is transmitted in a first-in, first-out order within a session.

If the NCB.SEND cannot be completed for any reason, the session ends abnormally and the session is dropped. This guarantees data integrity.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_LSN.

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.SEND.BROADCAST.DATAGRAM

Hex 22 Wait   A2 No-Wait

NCB.SEND.BROADCAST.DATAGRAM

Command Description: This command sends a datagram message to every station that has an NCB.RECEIVE.BROADCAST.DATAGRAM command pending.

Command Specifics: If the remote station does not have an NCB.RECEIVE.BROADCAST.DATAGRAM command pending, it will not get the message. If a station issues an NCB.SEND.BROADCAST.DATAGRAM and has an NCB.RECEIVE.BROADCAST.DATAGRAM pending, the station will receive its own message. If a station has several broadcast messages pending, the next send command issued will satisfy all NCB.RECEIVE.BROADCAST.DATAGRAM commands.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_NUM.

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.SEND.DATAGRAM

Hex 20 Wait A0 No-Wait

NCB.SEND.DATAGRAM

Command Description: This command sends a datagram message to any unique name or group name on the network.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_NUM
NCB_CALLNAME.

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.SEND.NO.ACK

Hex 71 Wait F1 No-Wait

NCB.SEND.NO.ACK

Command Description: The purpose of this command is to provide a send facility that does not require a data acknowledgment at the NETBIOS level. This reduces the completion time required of a send command.

This command sends data by the session number indicated in the local session number (LSN). The data is taken from the buffer indicated by the NCB_BUFFER@ for the indicated number of bytes.

When the remote side closes a session, all NCB.SEND.NO.ACK commands pending on the closed session are returned with a *session closed* status. If a local NCB.HANG.UP command is issued with any pending NCB.SEND.NO.ACK commands, the NCB.HANG.UP is delayed until the NCB.SEND.NO.ACK commands are completed.

If a session is aborted, a "session ended abnormally" status is returned. If the NCB.SEND.NO.ACK timeout expires, the session is aborted and a "command timed out" status is returned. Timeout values for the NCB.SEND.NO.ACK are associated with the session when an NCB.CALL or NCB.LISTEN was issued and cannot be specified here.

Message size must be between 0 and 65,535 bytes long.

If more than one NCB.SEND.NO.ACK or NCB.CHAIN.SEND is pending, the data is transmitted in a first-in, first-out order within a session.

Command Specifics: Lost data, (indicated by a X'07' return code), as well as the possible need for retransmission, are both the responsibility of the application program.

If the remote node cannot process the NCB.SEND.NO.ACK, it is handled as an NCB.SEND; a DATA_ACK must be received before the NCB.SEND.NO.ACK will be executed.

This command should be used for transactions and not for data transfers.

Notes:

1. Note the following on a X'07' return code:
   - The return code is generated for the node that issued NCB.SEND.NO.ACK.
   - The command terminating with the X'07' must be reissued.
   - The return code is issued if the receiver either had no NCB.RECEIVE or NCB.RECEIVE.ANY up, or the transmitted data exceeded the length of the receive buffer.
   - Unless the user is aware of the number of NCB.SEND.NO.ACKs issued, he has no indication of how many were not completed successfully.
   - All other NCB.SEND return codes apply.
2. For data transmitted with NCB.SEND.NO.ACK, only one receive command will be satisfied. The NCB.SEND.NO.ACK data buffer must be no longer than the remote receive buffer for successful completion.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_LSN.

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

NCB.SESSION.STATUS

Hex 34 Wait   B4 No-Wait

NCB.SESSION.STATUS

Command Description: This command obtains the status of either all sessions for a local name or all sessions for all local names.

Command Specifics: If the NCB_NAME field contains a name, the session status is returned for that name. If the NCB_NAME field contains an asterisk (*) in the first byte, the session status for all names is returned.

The minimum valid buffer length is 4 bytes. An "illegal buffer length" status is returned if the NCB_LENGTH field is less than 4.

A "message incomplete" status is returned if the NCB_LENGTH field is less than the status data being generated. To obtain all status data, the buffer length must be at least 36 times the number of sessions being reported plus 4.

Supplied Fields:

NCB_ADPTR_NUM (adapter number)
NCB_LENGTH
NCB_BUFFER@
NCB_POST@ (if the no-wait option is used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_NAME (Specify an asterisk (*) for all names).

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs)
NCB_LENGTH.

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

Table 4-6. Data Areas Returned for the NCB.SESSION.STATUS Command

Offset	Byte Length	8086 Type	Description
0	1	DB	Name number of sessions being reported; if 0, an add name is in process.
1	1	DB	Number of sessions with this name
2	1	DB	Number of NCB.RECEIVE.DATAGRAM and NCB.RECEIVE.BROADCAST.DATAGRAM commands outstanding
3	1	DB	Number of NCB.RECEIVE.ANY commands outstanding
4	1	DB	Local session number
5	1	DB	State of the session. This byte is represented as follows: **X'01'** LISTEN pending **X'02'** CALL pending **X'03'** Session established **X'04'** HANG UP pending **X'05'** HANG UP complete **X'06'** Session aborted
6	16	DB	Local name
22	16	DB	Remote name
38	1	DB	Number of NCB.RECEIVE commands pending
39	1	DB	Number of NCB.SEND and NCB.CHAIN.SEND commands pending

The contents of bytes 4 - 39 (36 bytes) are repeated for every session, if adequate buffer space is available.

NCB.STATUS

Hex 33 Wait B3 No-Wait

NCB.STATUS

Command Description: This command requests the status of either a local or remote NETBIOS.

Command Specifics: The NCB_CALLNAME field specifies which interface to get the status from. If the first byte of the field contains an asterisk (*), the local NETBIOS status is returned. If the NCB_CALLNAME field does not contain an asterisk, the status is returned from the node with that name. The status information is returned to the buffer defined by the NCB_BUFFER@ field. The minimum number of bytes is 60. The maximum buffer size needed to hold the status information is 18 times the maximum number of names plus 60.

Supplied Fields:

NCB_BUFFER@
NCB_LENGTH
NCB_CALLNAME (local or remote or an asterisk for local)
NCB_POST@ (if no-wait option used)
NCB_DD_ID (if using the OS/2 DD interface)
NCB_ADPTR_NUM (adapter number).

Returned Fields:

NCB_RETCODE
NCB_LENGTH
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

Table 4-7 (Page 1 of 4). Data Areas Returned for the NCB.STATUS Command

Offset	Byte Length	8086 Type	Description
0	6	DB	Adapter's encoded address.
6	1	DB	Software release level number: X'00' for NETBIOS 1.X, X'02' for NETBIOS 2.X, and X'03' for NETBIOS 3.X.
7	1	DB	Always X'00'.

Note: All counters roll over from X'F...F' to X'0...0' unless stated otherwise.

Table 4-7 (Page 2 of 4). Data Areas Returned for the NCB.STATUS Command

Offset	Byte Length	8086 Type	Description
8	2	DW	If offset 6 indicates NETBIOS 1.X, the first byte is X'FF' and the last byte indicates the software level. If offset 6 indicates NETBIOS 2.X, the first byte indicates the adapter type. For NETBIOS versions 2.0–2.33, the first byte indicates the adapter types as follows: X'FF' for a Token-Ring Network Adapter and X'FE' for a PC Network Adapter. For NETBIOS Version 2.5 or higher, the first byte indicates the adapter types as shown in Table 4-8 on page 4-41. For all versions of NETBIOS higher than 2.0, the first character of the second byte indicates the type of parameters used (X'1' represents old parameters and X'2' represents new parameters). The second character of the second byte is the second number of the software level. For example, X'1' is NETBIOS 2.1 and X'2' is NETBIOS 2.2.
10	2	DW	Duration of reporting period (in minutes). The Status Reporting Counter will not be incremented beyond X'FFFF' minutes if the reporting period (since the last NCB.RESET) exceeds 45.5 days.
12	2	DW	Number of FRMR frames received.
14	2	DW	Number of FRMR frames transmitted.
16	2	DW	Number of I frames received in error.
18	2	DW	Number of aborted transmissions.
20	4	DD	Number of successfully transmitted packets.
24	4	DD	Number of successfully received packets.
28	2	DW	Number of I frames transmitted in error.
30	2	DW	If offset 6 indicates NETBIOS 1.X, this field indicates lost data (out of SAP buffers). The counter does not increment beyond X'FFFF'. If offset 6 indicates NETBIOS 2.X, this field indicates the number of times that a buffer was not available to service a request from a remote computer.
32	2	DW	Number of times DLC T1 timer expired.
34	2	DW	Number of times DLC Ti timer expired.
36	4	DD	Address of extended status information (local status only; undefined for remote status). For OS/2 EE NETBIOS, this field is reserved. For the value of this field in LAPS (NETBIOS 4.0), see "NCB.STATUS Command Extension" on page 6-5.

Note: All counters roll over from X'F...F' to X'0...0' unless stated otherwise.

Table 4-7 (Page 3 of 4). Data Areas Returned for the NCB.STATUS Command

Offset	Byte Length	8086 Type	Description

Extended Status Information

1. This is the workstation system memory location of adapter-specified status. The data is in a fixed location and an NCB.STATUS need not be reissued. The data can be interrogated, but should not be disturbed.

   This data is available only for local status. The pointer is undefined for remote status.

   The format of the data is type DW, 2-byte fields:

Bytes 0–1	DIR.INITIALIZE bring-up error code
Bytes 2–3	DIR.OPEN.ADAPTER error code
Bytes 4–5	Latest network status
Bytes 6–7	Latest adapter check reason code
Bytes 8–9	Latest PC-detected error (contents of AX register)
Byte 10	Latest operational command code (4X or 5X). See note 2 below.
Byte 11	CCB return code of the latest implicit command. See note 2 below.

   Adapter counters:

Bytes 12–13	Line errors
Bytes 14–15	Internal errors
Bytes 16–17	Burst errors
Bytes 18–19	ARI/FCI delimiter
Bytes 20–21	Abort delimiter
Bytes 22–23	Reserved
Bytes 24–25	Lost frame
Bytes 26–27	Receive congestion
Bytes 28–29	Frame copied errors
Bytes 30–31	Frequency errors
Bytes 32–33	Token errors
Bytes 34–35	Reserved
Bytes 36–37	DMA bus errors
Bytes 38–39	DMA parity errors

   Locally administered address (for NETBIOS Version 2.5 or higher):

Bytes 40–45	Locally administered address

   The adapter counters are valid only if no network status appendage is defined. If an appendage is defined, it is the responsibility of the user to maintain these counts. When no appendage is defined, these counters are updated when network status bit 7 (counter overflow) is reported. (The counters wrap from X'FFFF' to X'0000'.)

   These counters are the same as obtained with a DIR.READ.LOG command. Any application program that issues a DIR.READ.LOG command will cause these counters to be incorrect as to the correct total since they are reset by the DIR.READ.LOG command.

2. When NETBIOS is initiated, DIR.INITIALIZE and DIR.OPEN.ADAPTER are sometimes executed and a DLC.OPEN.SAP is executed. Bytes 10 and 11 are useful for determining the reason for certain X'4X' return codes. Byte 10 provides the last of these CCB commands that was executed, and byte 11 contains the related CCB return code.

Offset	Byte Length	8086 Type	Description
40	2	DW	Current number of free NCBs.
42	2	DW	Configured maximum NCBs.

Note: All counters roll over from X'F...F' to X'0...0' unless stated otherwise.

Table 4-7 (Page 4 of 4). Data Areas Returned for the NCB.STATUS Command

Offset	Byte Length	8086 Type	Description
44	2	DW	If offset 6 indicates NETBIOS 1.X, this field indicates the maximum number of NCBs (always 255). If offset 6 indicates NETBIOS 2.X, this field indicates the maximum number of NCBs. With old parameters, the maximum number is 255. With new parameters, the maximum number is defined at load time.
46	2	DW	If offset 6 indicates NETBIOS 1.X, this field indicates the number of times a local station went busy. If offset 6 indicates NETBIOS 2.X, this field indicates the number of times that a buffer was not available to transmit information.
48	2	DW	Maximum datagram packet size.
50	2	DW	Number of pending sessions.
Note: The pending session is either an NCB_CALL pending, NCB_LISTEN pending, session established, session aborted, NCB_HANG_UP pending, or NCB_HANG_UP complete.			
52	2	DW	Configured maximum pending sessions.
54	2	DW	If offset 6 indicates NETBIOS 1.X, this field indicates the maximum number of pending sessions (always 254). If offset 6 indicates NETBIOS 2.X, this field indicates the maximum number of pending sessions. With old parameters, the maximum number is always 254. With new parameters, the maximum number is defined at load time.
56	2	DW	Maximum size of session data packet.
58	2	DW	Number of names in the local names table.
60	XXXX	DB	Local names table. Each name requires 18 bytes.

Note: The first 16 bytes of each entry represent the name and the last 2 bytes represent the name status. The first status byte is equal to the name number (NCB_NUM). The second status byte is the status when it is masked with a X'87'. The mask is used to get the last 3 bits of the byte. The other bits are reserved and can have non-zero values.

Nrrrr000 - Trying to register a name
Nrrrr100 - A registered name
Nrrrr101 - A deregistered name
Nrrrr110 - A detected duplicate name
Nrrrr111 - A detected duplicate name with deregister pending
Where: r = Reserved bit
Where: N = 0 if the name is a unique name
Where: N = 1 if the name is a group name.

DEREGISTERED NAME: If the name to be deleted still has an active session when a NCB.DELETE.NAME command is issued, the name is marked as "deregistered" and the name is removed from the local names table as soon as the session is closed.

Note: All counters roll over from X'F...F' to X'0...0' unless stated otherwise.

Table 4-8 on page 4-41 shows the values displayed in the first byte at offset 8 of the data area returned for the NCB.STATUS command (NETBIOS Version 2.5 and higher).

Table 4-8. Adapter Types

Adapter Type	Function Number
Token-Ring	X'FF'
PC Network	X'FE'
Ethernet	X'FD'
IBM FDDI	X'FC'
3174 Peer	X'FB'
Reserved	X'FA'
Reserved	X'F9'

NCB.TRACE

Hex 79 Wait F9 No-Wait

NCB.TRACE

Command Description: This command activates and deactivates a trace of all the NCBs issued to NETBIOS and some of the CCBs issued by NETBIOS, including transmits and receives.

For OS/2: The TRACE command is not available with OS/2. It is replaced with the OS/2 system trace. For a description of the OS/2 system trace for OS/2 EE, see Appendix E, "Operating System/2 Extended Edition Information"; for a description of the trace facility in LAPS, see "Operating System/2 Trace Facility" on page 6-10.

Receipt of NETBIOS protocol messages "remote trace off" is not supported.

Trace Entries:

The NETBIOS major trace code is X'A4'. There are two minor trace codes: X'20' indicates an OS/2 DD interface level trace entry and X'21' indicates an OS/2 DLR interface level trace entry.

Command Specifics: Both the immediate and the wait command are completed immediately and no post address is used.

To initiate the trace, the NCB_NUM field must be set to X'FF', the NCB_LENGTH field must be set to the length of the trace table (1024 or greater), and the NCB_BUFFER@ must be set to the address of the trace table location in workstation memory. The NCB_ADPTR_NUM field must be set to a valid adapter number, but the trace will be active for both adapters. The automatic adapter open function of NETBIOS does not occur when this command is issued. This command can be issued either when an adapter is open or when it is closed.

To terminate the trace, this command is issued with the NCB_NUM field set to X'00'. In this case the NCB_LENGTH and NCB_BUFFER@ fields are returned values.

To terminate the trace locally and at remote adapters that have NETBIOS running and a trace active, set the NCB_NUM field to X'01'. This terminates the NCB.TRACE and also issues a PDT.TRACE.OFF command to terminate the adapter support software trace. It also causes a request to be sent to NETBIOS at all remote adapters to issue NCB.TRACE (off) and PDT.TRACE.OFF commands. If the local trace is not active, the command can still be issued to terminate remote traces even though the command will be completed with a return code of X'0D'.

Supplied Fields:

NCB_BUFFER@ (trace table address)
NCB_LENGTH (trace table length (at least 1024))
NCB_NUM
    X'FF' = trace on,
    X'00' = local trace off
    X'01' = local and all remote trace off
NCB_ADPTR_NUM (adapter number).

Returned Fields:

NCB_RETCODE

NCB_RESERVE (if error X'4X' or X'FX' occurs)

NCB_BUFFER@ (address of the trace table, returned when the trace off option is requested in the NCB_NUM field)

NCB_LENGTH (length of the trace table, returned when the trace off option is requested in the NCB_NUM field).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

Each trace entry is 32 bytes long. The initial entry is the table header. The next entry is the first trace entry. When all the trace table entry locations have been used, the table is wrapped back to the first entry overwriting previous entries. The initial, or header, entry is not overwritten.

Trace Table Header
1st Trace Entry
2nd Trace Entry
•
•
•
Nth Trace Entry

Figure 4-1. Trace Entry Format Example

Table 4-9. The Trace Table Header Format

Byte	Usage
00-03	The address of the trace buffer (its own address)
04-05	The length of the trace buffer
06-07	The offset of the first trace entry
08-15	Code change level
16-19	Trace entry counter (low word, high word)
20-21	Number of times internal response buffers were exhausted
22-23	Number of times internal transmit buffers were exhausted
24-27	Reserved
28-29	Offset of the end of the trace buffer
30-31	Offset of the next trace entry to be used

Bytes 0 through 5 of all the trace entries are the same format. The format of the remainder of the entry is dependent upon the content of bytes 2 (type) and 3 (modifier).

Table 4-10. Trace Entry Format (Bytes 0 through 5)

Byte	Usage
0	Adapter X'AA' = Adapter 0 X'BB' = Adapter 1 X'FF' = See note below.
1	X'00' (non-3270 XMA code) or bank ID (if 3270 XMA code executing)
2	Type
3	Modifier of Type
4-5	Number of 100-ms ticks since last entry

Note: If this command is the first use of NETBIOS since it was loaded, the X'FF' entry will appear for the next NCB command issued. That should be the only occurrence of a X'FF' entry.

Table 4-11 (Page 1 of 3). Trace Entry Format (Bytes 6 through 31)

Byte 2 Type	Byte 3 Modify	Meaning	Bytes 6-31	
FF	00	NCB issued	**Byte 6**	The byte preceding the NCB
			Byte 7	The byte following the NCB
			Bytes 8-11	NCB post address
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-31	First 16 bytes of the NCB
	01	NCB immediate return	**Byte 6**	The byte preceding the NCB
			Byte 7	The byte following the NCB
			Bytes 8-11	NCB post address
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-31	First 16 bytes of the NCB
	02	NCB final return code	**Byte 6**	The byte preceding the NCB
			Byte 7	The byte following the NCB
			Bytes 8-11	NCB post address
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-31	First 16 bytes of the NCB

Table 4-11 (Page 2 of 3). Trace Entry Format (Bytes 6 through 31)

Byte 2 Type	Byte 3 Modify	Meaning	Bytes 6-31	
EE	00	Network status	**Bytes 6-7**	DS
			Bytes 8-9	SS
			Bytes 10-11	SP
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-17	Network status
	01	Protocol driver internal error	**Bytes 6-7**	DS
			Bytes 8-9	SS
			Bytes 10-11	SP
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-17	Outstanding CCBs passed by the PC-detected error appendage
	02	Adapter status	**Bytes 6-7**	DS
			Bytes 8-9	SS
			Bytes 10-11	SP
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-19	Pointer to the first in a queue of commands returned
			Bytes 20-21	Adapter check reason code
	03	DLC status	**Bytes 6-7**	DS
			Bytes 8-9	SS
			Bytes 10-11	SP
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-17	Station ID
			Bytes 18-19	DLC status
			Bytes 20-24	FRMR data
			Byte 25	Access priority
			Bytes 26-31	Remote node address
0B	The CCB return code	Transmit I frame	**Bytes 6-7**	DS
			Bytes 8-9	SS
			Bytes 10-11	SP
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-31	NETBIOS header transmitted *
0D	The CCB return code	Transmit UI frame	**Bytes 6-7**	DS
			Bytes 8-9	SS
			Bytes 10-11	SP
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-31	NETBIOS header transmitted
28	00	Data received and processed	**Bytes 6-7**	DS
			Bytes 8-9	SS
			Bytes 10-11	SP
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-31	NETBIOS header received *
	FF	Data received and not processed	**Bytes 6-7**	DS
			Bytes 8-9	SS
			Bytes 10-11	SP
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-31	NETBIOS header received *

Table 4-11 (Page 3 of 3). Trace Entry Format (Bytes 6 through 31)

Byte 2 Type	Byte 3 Modify	Meaning	Bytes 6-31	
Any other CCB cmd code	The CCB return code	Any other CCB command	**Bytes 6-7**	DS
			Bytes 8-9	SS
			Bytes 10-11	SP
			Bytes 12-15	Address of data in workstation memory
			Bytes 16-31	Image of the CCB

Note: *If byte 20 of the trace is either X'15' or X'16', then bytes 30-31 contain the length of the user data transmitted or received.

NCB.UNLINK

Hex 70 Wait

NCB.UNLINK

Command Description: This command is provided for NETBIOS compatibility. NETBIOS 3.0 treats this as a "no-operation."

Supplied Fields:

NCB_ADPTR_NUM (adapter number).

Returned Fields:

NCB_RETCODE
NCB_RESERVE (if error X'4X' or X'FX' occurs).

Valid Return Codes: See "NCB Return Codes Listed by Command" on page B-34.

Chapter 5. The NETBIOS Frames Protocol

- About This Chapter . . . 5-3
  - Assumptions . . . 5-3
  - Related Documents . . . 5-3
- Terms and Definitions . . . 5-3
    - The NETBIOS Service Access Point (SAP) . . . 5-3
    - Addressing . . . 5-3
    - Broadcast . . . 5-3
    - Data Structures . . . 5-4
    - Translation . . . 5-4
- NETBIOS Commands . . . 5-4
- General Information on Remote Name Directory . . . 5-4
  - New NCB Commands . . . 5-5
  - Header Format Overview . . . 5-5
  - NETBIOS Header . . . 5-6
    - NETBIOS Non-Session Frame Header (DLC UI-Frame) . . . 5-6
    - NETBIOS Session Frame Header (DLC I-Frame) . . . 5-6
    - NETBIOS Frame Header Fields . . . 5-6
  - NETBIOS Frame Summary . . . 5-7
    - NETBIOS Frames Listed by Function . . . 5-9
    - NETBIOS Frames Listed by Transmission Type . . . 5-10
- NETBIOS Frame Formats and Descriptions . . . 5-11
  - ADD_GROUP_NAME_QUERY . . . 5-11
  - ADD_NAME_QUERY . . . 5-12
  - NAME_IN_CONFLICT . . . 5-13
  - STATUS_QUERY . . . 5-14
  - TERMINATE_TRACE . . . 5-16
  - DATAGRAM . . . 5-16
  - DATAGRAM_BROADCAST . . . 5-17
  - NAME_QUERY . . . 5-18
  - ADD_NAME_RESPONSE . . . 5-20
  - NAME_RECOGNIZED . . . 5-21
  - STATUS_RESPONSE . . . 5-22
  - TERMINATE_TRACE . . . 5-24
  - DATA_ACK . . . 5-24
  - DATA_FIRST_MIDDLE . . . 5-25
  - DATA_ONLY_LAST . . . 5-28
  - SESSION_CONFIRM . . . 5-30
  - SESSION_END . . . 5-32
  - SESSION_INITIALIZE . . . 5-33
  - NO_RECEIVE . . . 5-34
  - RECEIVE_OUTSTANDING . . . 5-36
  - RECEIVE_CONTINUE . . . 5-37
  - SESSION_ALIVE . . . 5-38
- NETBIOS Protocol Examples Without RND . . . 5-39
  - Name Management Examples . . . 5-40
    - Add a Name to the Network . . . 5-40
    - Add a Name: Name Already On Network . . . 5-41
    - Add a Name: Receive Multiple Responses . . . 5-42
  - Remote Adapter Status Examples . . . 5-43
    - Remote Adapter Status for a Name That Is Not on the Network . . . 5-43

Remote Adapter Status for a Name That Is On the Network 5-44
Remote Adapter Status Data: Segmentation 5-45
Session Establishment Examples 5-46
Call a Name: Name Not on Network 5-46
Call a Name: Name on Network but No Listen 5-47
Call a Name: Name Found—Start Session 5-48
Session Data Transfer Examples 5-49
Send Session Data: One Send and One Receive 5-49
Send Session Data: One Send and Multiple Receives 5-50
Send Session Data: Segmentation and One Receive 5-51
Send Session Data: Segmentation and Multiple Receives 5-52
NETBIOS Protocol Examples with RND 5-53
Remote Adapter Status for a Name Not on Network or RND 5-54
Remote Adapter Status for a Name Not in RND but on the Network . . 5-55
Remote Adapter Status for a Remote Name That Is in RND 5-56
Session Establishment RND Examples 5-57
Call a Name: Name Not on Network or RND 5-57
Call a Name: Name Not in RND but Is on Network—Start Session . . . 5-58
NETBIOS Protocol SEND.NO.ACK Examples 5-59
Session Establishment Examples 5-60
Call a Name: Name Found—Start Session 5-60
Session Data Transfer Examples 5-61
Send Session Data: One SEND.NO.ACK and No RECEIVE 5-61
Send Session Data: One SEND.NO.ACK and One RECEIVE 5-62
Send Session Data: One SEND.NO.ACK and Multiple RECEIVEs . . . 5-63

About This Chapter

This chapter describes the frame formats and protocols used by NETBIOS to communicate across one of the IBM local area networks. Each NETBIOS command results in a local action in the station and may result in the transmission of NETBIOS frames on the network.

The NETBIOS interface is a communication interface consisting of commands and their related control block structures. Communication on the network is done by using names to identify each communicating device. Two types of communication services are provided with this interface. One is a session-level service in which NETBIOS makes use of the IEEE 802.2 LLC connection-oriented services (Type 2) and the other is a datagram service in which NETBIOS makes use of the IEEE 802.2 LLC connectionless services (Type 1).

Assumptions

You should be familiar with the information contained in Chapter 4, "NETBIOS." You should also be familiar with general network architecture.

In all cases, except where explicitly stated otherwise, a function that applies to some specific version also applies to later (higher) versions.

Related Documents

The following Local Area Network standards may be helpful:

- ISO 8802-2 Local Area Networks Logical Link Control
- ISO 8802-5 Local Area Networks Token Ring Access Method.

Terms and Definitions

Refer to the glossary in the back of this manual for a complete terminology list.

The NETBIOS Service Access Point (SAP)

NETBIOS opens and uses SAP X'F0'. This SAP value is reserved by IBM for use by the NETBIOS function.

Addressing

NETBIOS sets the functional address X'00000080' on the adapter. This functional address is used by NETBIOS when broadcasting frames. This functional address bit is reserved by IBM for use by the NETBIOS function.

Broadcast

NETBIOS makes use of the broadcast capability of the network. In all cases but one, frames are sent as single-route broadcast: the broadcast bit and the single-route broadcast bit in the routing control field are both set to 1. In one case a frame is sent as a general broadcast: the broadcast bit in the routing control field is set to 1 and the single-route broadcast bit is set to 0 (zero).

Single-Route Broadcast: Single-route broadcast means that only one frame is to be delivered to each ring. This can be accomplished by designating selected bridges to forward single-route broadcast frames. Frames sent using single-route broadcast will have the broadcast bit and the single-route broadcast bit in the routing control field both set to 1.

General Broadcast: General broadcast means that all bridges will forward such frames. A frame sent using general broadcast will have the broadcast bit in the routing control field set to 1 and the single-route broadcast bit set to 0 (zero).

Data Structures

The data definition of each NETBIOS frame is shown using the INTEL data types for the various fields. The DB (define byte) type is used to define single byte fields and byte string fields. The DW (define word) type is used primarily to define 2-byte numeric fields.

Note: Fields defined as DW reside in memory as byte reversed. When a frame is transmitted, these fields are transmitted as they appear in memory–byte reversed.

Translation

NETBIOS assumes no translation for any character strings used in the NETBIOS frames (for example, names). These strings are viewed simply as binary data.

NETBIOS Commands

Below is a list of the NETBIOS commands. Some of these command names are referred to in the description of some of the NETBIOS frames.

NCB.ADD.GROUP.NAME
NCB.ADD.NAME
NCB.CALL
NCB.CANCEL
NCB.CHAIN.SEND
NCB.CHAIN.SEND.NO.ACK
NCB.DELETE.NAME
NCB.FIND.NAME
NCB.HANGUP
NCB.LAN.STATUS.ALERT
NCB.LISTEN
NCB.RECEIVE
NCB.RECEIVE.ANY
NCB.RECEIVE.BROADCAST.DATAGRAM
NCB.RECEIVE.DATAGRAM
NCB.RESET
NCB.SEND
NCB.SEND.BROADCAST.DATAGRAM
NCB.SEND.DATAGRAM
NCB.SEND.NO.ACK
NCB.SESSION.STATUS
NCB.STATUS
NCB.TRACE
NCB.UNLINK

General Information on Remote Name Directory

Remote Name Directory (RND) functions are implemented in NETBIOS Version 2.1.

To reduce the unnecessary interrupts to NETBIOS nodes, which are caused by broadcast frames to the NETBIOS functional address, the RND function is used to send the frame to a specific node, whenever possible.

The local station locates a remote name (on-ring only if RND is used) by:

1. Issuing a NAME_QUERY on-ring once
2. Issuing a NAME_QUERY off-ring per transmit count
3. Transmitting timeout values that can be defined by configuration parameters.

The range for NCB.TRANSMIT.COUNT is 1 through 10, and the default is 6. The range for NCB.TRANSMIT.TIMEOUT is 1/2 second to 10 seconds, and the default is 1/2 second.

When RND is used, after the local station has located a remote name, the remote node address is saved and subsequent messages to that name will be to a specific node rather than a broadcast to all nodes. This is true for CALLS and STATUS QUERIES (and also for SEND DATAGRAMS if REMOTE.DATAGRAM.CONTROL is used).

Note: When the RND function is being used, a duplicate network name will not be detected by a CALL function unless it is the first time that the remote (CALL) name has been used.

New NCB Commands

Two new commands: NCB.SEND.NO.ACK and NCB.CHAIN.SEND.NO.ACK were introduced in NETBIOS Version 2.2. The purpose of the NCB.SEND.NO.ACK and NCB.CHAIN.SEND.NO.ACK commands is to provide a SEND facility that does not require NETBIOS to transmit a data acknowledgment. This is to reduce the completion time of a SEND type of command. There is also a new NCB.LAN.STATUS.ALERT command. This LAN status command allows application programs to be alerted of temporary ring error conditions that last more than one minute.

Header Format Overview

All NETBIOS frames contain a header that immediately follows the data link control (DLC) header:

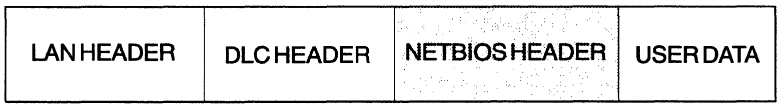


Figure 5-1. NETBIOS Frame Example

Notes:

1. With respect to the DLC frame, the NETBIOS header is data.
2. USER DATA is applicable only on certain messages.

NETBIOS Header

In this section the various NETBIOS frame headers are described. Refer to *IBM Token-Ring Network Architecture Reference* for a complete description of the LAN and DLC headers. Following are examples of NETBIOS frames.

NETBIOS Header Length	X'EFFF'	Command	Optional Data1	Optional Data2	Xmit/Resp Correlator	Dest ID	Source ID

Figure 5-2. General Format

NETBIOS Non-Session Frame Header (DLC UI-Frame)

These frames are transmitted using DLC connectionless services to either a functional address or a specific address:

0	2	4	5	6	8	12	28	44
NETBIOS Header Length	X'EFFF'	Command	Optional Data1	Optional Data2	Xmit/Resp Correlator	Dest Name	Source Name	

Figure 5-3. NETBIOS Non-Session Frame Header (DLC UI-Frame)

NETBIOS Session Frame Header (DLC I-Frame)

These frames are transmitted using DLC connection-oriented services to a specific address:

0	2	4	5	6	8	12	13	14
NETBIOS Header Length	X'EFFF'	Command	Optional Data1	Optional Data2	Xmit/Resp Correlator	Dest Num	Source Num	

Figure 5-4. NETBIOS Session Frame Header (DLC I-Frame)

NETBIOS Frame Header Fields

BYTES	MEANING
0-1	NETBIOS Header Length–The length of NETBIOS header (including the length field)
2-3	X'EFFF'–A delimiter that indicates that subsequent data is destined for the NETBIOS function.
4	Command–A specific protocol command that indicates the type or function of the frame. The command codes are partitioned as follows: **X'00' to X'13'** UI frames **X'14' to X'1F'** I frames
5	Data1–1 byte of optional data per a specific command
6-7	Data2–2 bytes of optional data per a specific command

8-11 Correlator–One or two numbers in the range X'0001' to X'FFFF'. Used to associate received responses with transmitted requests.

- Transmit Correlator: The value returned in a response to a given query (the value was received as the response correlator).
- Response Correlator: The value expected (in the transmit correlator field) when the response to that message is received.

Note: Commands X'00' to X'07' will be discarded by the sending NETBIOS upon return to the station. For example, when an ADD_NAME_QUERY is received by the station that transmitted it, the message is discarded.

Remaining bytes depend upon message type.

1. Non-Session frame (DLC UI frame):

Bytes	**Meaning**
12-27	Destination Name–A 16-character destination name.
28-43	Source Name–A 16-character source name.

2. Session frame (DLC I frame):

Bytes	**Meaning**
12	Destination Number–A 1-byte destination session number.
13	Source Number–A 1-byte source session number.

NETBIOS Frame Summary

Each NETBIOS frame has been given a name that suggests its use in implementing the various NETBIOS commands. The following tables summarize these commands.

Table 5-1 (Page 1 of 2). NETBIOS Frames Listed Alphabetically

Frame Name	Code	Function
ADD_GROUP_NAME_QUERY	X'00'	Check for duplicate group name on network
ADD_NAME_QUERY	X'01'	Check for duplicate name on network
ADD_NAME_RESPONSE	X'0D'	Negative response: add name is duplicate
DATA_ACK	X'14'	DATA_ONLY_LAST acknowledgment
DATA_FIRST_MIDDLE	X'15'	Session data message–first or middle frame
DATAGRAM	X'08'	Application program-generated datagram
DATAGRAM_BROADCAST	X'09'	Application program-generated broadcast datagram
DATA_ONLY_LAST	X'16'	Session data message–only or last frame
NAME_IN_CONFLICT	X'02'	Duplicate names detected
NAME_QUERY	X'0A'	Request to locate a name on the network
NAME_RECOGNIZED	X'0E'	Name Recognized: NAME_QUERY response
NO_RECEIVE	X'1A'	No receive command to hold received data
RECEIVE_CONTINUE	X'1C'	Indicates receive pending
RECEIVE_OUTSTANDING	X'1B'	Retransmit last data-receive command up
SESSION_ALIVE	X'1F'	Verify session is still active

Table 5-1 (Page 2 of 2). NETBIOS Frames Listed Alphabetically

Frame Name	Code	Function
SESSION_CONFIRM	X'17'	SESSION_INITIALIZE acknowledgment
SESSION_END	X'18'	Session termination
SESSION_INITIALIZE	X'19'	A session has been set up
STATUS_QUERY	X'03'	Request remote node status
STATUS_RESPONSE	X'0F'	Remote node status information, STATUS_QUERY response
TERMINATE_TRACE	X'07'	Terminate traces at remote nodes
TERMINATE_TRACE	X'13'	Terminate traces at local and remote nodes

Table 5-2. NETBIOS Frames Listed Numerically

Frame Name	Code	Function
ADD_GROUP_NAME_QUERY	X'00'	Check for duplicate group name on network
ADD_NAME_QUERY	X'01'	Check for duplicate name on network
NAME_IN_CONFLICT	X'02'	Duplicate names detected
STATUS_QUERY	X'03'	Request remote node status
TERMINATE_TRACE	X'07'	Terminate traces at remote nodes
DATAGRAM	X'08'	Application program-generated datagram
DATAGRAM_BROADCAST	X'09'	Application program-generated broadcast datagram
NAME_QUERY	X'0A'	Request to locate a name on the network
ADD_NAME_RESPONSE	X'0D'	Negative response: add name is duplicate
NAME_RECOGNIZED	X'0E'	Name Recognized: NAME_QUERY response
STATUS_RESPONSE	X'0F'	Remote node status information, STATUS_QUERY response
TERMINATE_TRACE	X'13'	Terminate traces at local and remote nodes
DATA_ACK	X'14'	DATA_ONLY_LAST acknowledgment
DATA_FIRST_MIDDLE	X'15'	Session data message–first or middle frame
DATA_ONLY_LAST	X'16'	Session data message–only or last frame
SESSION_CONFIRM	X'17'	SESSION_INITIALIZE acknowledgment
SESSION_END	X'18'	Session termination
SESSION_INITIALIZE	X'19'	A session has been set up
NO_RECEIVE	X'1A'	No receive command to hold received data
RECEIVE_OUTSTANDING	X'1B'	Retransmit last data-receive command up
RECEIVE_CONTINUE	X'1C'	Indicates receive pending
SESSION_ALIVE	X'1F'	Verify session is still active

NETBIOS Frames Listed by Function

The NETBIOS frames are grouped below by function.

Table 5-3. NETBIOS Name Management Frames

Frame Name	Code	Function
ADD_GROUP_NAME_QUERY	X'00'	Check for duplicate group name on network
ADD_NAME_QUERY	X'01'	Check for duplicate name on network
ADD_NAME_RESPONSE	X'0D'	Negative response: add name is duplicate
NAME_IN_CONFLICT	X'02'	Duplicate names detected

Table 5-4. NETBIOS Session Establishment and Termination Frames

Frame Name	Code	Function
NAME_QUERY	X'0A'	Request to locate a name on the network
NAME_RECOGNIZED	X'0E'	Name Recognized: NAME_QUERY response
SESSION_ALIVE	X'1F'	Verify session is still active
SESSION_CONFIRM	X'17'	SESSION_INITIALIZE acknowledgment
SESSION_END	X'18'	Session termination
SESSION_INITIALIZE	X'19'	A session has been set up

Table 5-5. NETBIOS Data Transfer Frames

Frame Name	Code	Function
DATA_ACK	X'14'	DATA_ONLY_LAST acknowledgment
DATA_FIRST_MIDDLE	X'15'	Session data message–first or middle frame
DATAGRAM	X'08'	Application program-generated datagram
DATAGRAM_BROADCAST	X'09'	Application program-generated broadcast datagram
DATA_ONLY_LAST	X'16'	Session data message–only or last frame
NO_RECEIVE	X'1A'	No receive command to hold received data
RECEIVE_CONTINUE	X'1C'	Indicates receive pending
RECEIVE_OUTSTANDING	X'1B'	Re-transmit last data–receive command up

Table 5-6. Additional NETBIOS Frames

Frame Name	Code	Function
STATUS_QUERY	X'03'	Request remote node status
STATUS_RESPONSE	X'0F'	Remote node status information, STATUS_QUERY response
TERMINATE_TRACE	X'07'	Terminate traces at remote nodes
TERMINATE_TRACE	X'13'	Terminate traces at local and remote nodes

NETBIOS Frames Listed by Transmission Type

The NETBIOS frames are grouped below according to the type of transmission used for each frame.

Table 5-7. NETBIOS UI Frames to Functional Address, Single-Route Broadcast

Frame Name	Code	Function
ADD_GROUP_NAME_QUERY	X'00'	Check for duplicate group name on network
ADD_NAME_QUERY	X'01'	Check for duplicate name on network
DATAGRAM *	X'08'	Application program-generated datagram
DATAGRAM_BROADCAST	X'09'	Application program-generated broadcast datagram
NAME_IN_CONFLICT	X'02'	Duplicate names detected
NAME_QUERY *	X'0A'	Request to locate a name on the network
STATUS_QUERY *	X'03'	Request remote node status
TERMINATE_TRACE	X'07'	Terminate traces at remote nodes
TERMINATE_TRACE	X'13'	Terminate traces at local and remote nodes

* If remote name directory function is used, UI frames are sent to the specific address with no broadcast.

Table 5-8. NETBIOS UI Frames to Specific Address, No Broadcast

Frame Name	Code	Function
ADD_NAME_RESPONSE	X'0D'	Negative response: add name is duplicate
STATUS_RESPONSE	X'0F'	Remote node status information, STATUS_QUERY response

Table 5-9. NETBIOS UI Frames to Specific Address, General Broadcast

Frame Name	Code	Function
NAME_RECOGNIZED	X'0E'	Name Recognized: NAME_QUERY response

Table 5-10. NETBIOS I-Frames to Specific Address, No Broadcast

Frame Name	Code	Function
DATA_ACK	X'14'	DATA_ONLY_LAST acknowledgment
DATA_FIRST_MIDDLE	X'15'	Session data message–first or middle frame
DATA_ONLY_LAST	X'16'	Session data message–only or last frame
NO_RECEIVE	X'1A'	No receive command to hold received data
RECEIVE_OUTSTANDING	X'1B'	Retransmit last data–receive command up
SESSION_CONFIRM	X'17'	SESSION_INITIALIZE acknowledgment
SESSION_END	X'18'	Session termination
SESSION_INITIALIZE	X'19'	A session has been set up
RECEIVE_CONTINUE	X'1C'	Indicates receive pending
SESSION_ALIVE	X'1F'	Verify session is still active

NETBIOS Frame Formats and Descriptions

This section describes each NETBIOS frame. The frame descriptions are organized by command value.

ADD_GROUP_NAME_QUERY

Hex 00

ADD_GROUP_NAME_QUERY

Initiation: This frame is initiated by an ADD.GROUP.NAME command.

Function: To verify that the group name to be added does not already exist as a unique name within the network.

Transmission: This frame is transmitted as a UI frame to the NETBIOS functional address using single-route broadcast.

Responses: No response within the system timeout period indicates that the name is not already in use as a unique name. An ADD_NAME_RESPONSE indicates that the name is already in use as a unique name and cannot be added as a group name.

Note: An ADD_NAME_RESPONSE from more than one remote adapter causes the local adapter to transmit a NAME_IN_CONFLICT to all remote adapters.

Retries: The number of times the frame is transmitted can be defined by the NCB.TRANSMIT.COUNT parameter. The range for this count is 1 to 10 and the default is 6.

Note: The frame will be sent this number of times whether a response is received or not.

Timeouts: The timeout value between retransmissions of this frame can be defined by the NCB.TRANSMIT.TIMEOUT parameter. The range is 1/2 second to 10 seconds, and the default is 1/2 second.

Table 5-11. ADD_GROUP_NAME_QUERY Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'00'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	X'nnnn'
12	DEST NAME	16	DB	Reserved
28	SOURCE NAME	16	DB	Group name to be added

Response Correlator

Explanation: This is a 2-byte correlator generated by the sender and used to correlate any responses to this query. It is returned in any ADD_NAME_RESPONSE in the transmit correlator field.

Source Name

Explanation: This is a 16-byte field that contains the group name to be added. This is the NCB_NAME field in the ADD.GROUP.NAME command.

ADD_NAME_QUERY

Hex 01

ADD_NAME_QUERY

Initiation: This frame is initiated by an ADD.NAME command.

Function: To verify that the name to be added is a unique name within the network.

Transmission: This frame is transmitted as a UI frame to the NETBIOS functional address using single-route broadcast.

Responses: No response within the system timeout period indicates that the name is a unique name. An ADD_NAME_RESPONSE indicates that the name is not unique and cannot be added.

Note: An ADD_NAME_RESPONSE from more than one remote adapter will cause the local adapter to transmit a NAME_IN_CONFLICT to all remote adapters.

Retries: The number of times the frame is transmitted can be defined by the NCB.TRANSMIT.COUNT parameter. The range for this count is 1 to 10 and the default is 6.

Note: The frame will be sent this number of times whether a response is received or not.

Timeouts: The timeout value between transmissions of this frame can be defined by the NCB.TRANSMIT.TIMEOUT parameter. The range is 1/2 second to 10 seconds and the default is 1/2 second.

Table 5-12. ADD_NAME_QUERY Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'01'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	X'nnnn'
12	DEST NAME	16	DB	Reserved
28	SOURCE NAME	16	DB	Name to be added

Response Correlator

Explanation: This is a 2-byte correlator generated by the sender and used to correlate any responses to this query. It is returned in any ADD_NAME_RESPONSE in the transmit correlator field.

Source Name

Explanation: This is a 16-byte field that contains the name to be added. This is the NCB_NAME field in the ADD.NAME command.

NAME_IN_CONFLICT

Hex 02

NAME_IN_CONFLICT

Initiation: This frame is sent if the following is detected:

- More than one adapter has the same unique name registered
- A name is registered as both a group name and a unique name in the network.

For example, this frame is sent if an ADD_NAME_RESPONSE is received from more than one node following transmission of an ADD_NAME_QUERY or ADD_GROUP_NAME_QUERY. This frame is also sent if a NAME_RECOGNIZED is received from more than one adapter following transmission of a NAME_QUERY.

Function: To indicate to all nodes that the specified name has been detected as being registered at more than one node.

Transmission: This frame is transmitted as a UI frame to the NETBIOS functional address using single-route broadcast.

Responses: None

Retries: The number of times the frame is transmitted can be defined by the NCB.TRANSMIT.COUNT parameter. The range for this count is 1 to 10, and the default is 6.

Timeouts: The timeout value between retransmissions of this frame can be defined by the NCB.TRANSMIT.TIMEOUT parameter. The range is 1/2 second to 10 seconds, and the default is 1/2 second.

Table 5-13 (Page 1 of 2). NAME_IN_CONFLICT Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'02'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NAME	16	DB	Name in conflict

Table 5-13 (Page 2 of 2). NAME_IN_CONFLICT Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
28	SOURCE NAME	16	DB	Sending node NAME_NUMBER_1

Destination Name

Explanation: This is a 16-byte field containing the name that was determined to be in conflict.

Source Name

Explanation: This is a 16-byte field that contains the NAME_NUMBER_1, consisting of 10 bytes of binary zeros followed by the 6-byte permanently encoded address for this adapter.

STATUS_QUERY

Hex 03

STATUS_QUERY

Initiation: This frame is sent because of STATUS command.

Function: To request remote adapter status.

Transmission: This frame is transmitted as a UI frame to the NETBIOS functional address using single-route broadcast.

Responses: No response within the system timeout period causes the command to time out. A STATUS_RESPONSE is used to return the remote adapter status.

If a status request is sent to a NETBIOS 2.1 station from a NETBIOS 2.1 station, and the remote adapter cannot transmit all the status data at once, upon receiving the STATUS_RESPONSE, an additional STATUS_QUERY frame is transmitted as a UI frame to the specific address of the node that sent the STATUS_RESPONSE. This process is repeated until all the data is received or the application program buffer is filled up.

Retries: The number of times the frame is transmitted can be defined by the NCB.TRANSMIT.COUNT parameter. The range for this count is 1 to 10 and the default is 6.

Timeouts: The timeout value between transmissions of this frame can be defined by the NCB.TRANSMIT.TIMEOUT parameter. The range is 1/2 second to 10 seconds, and the default is 1/2 second.

Table 5-14 (Page 1 of 2). STATUS_QUERY Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'03'
5	DATA1	1	DB	X'nn'

Table 5-14 (Page 2 of 2). STATUS_QUERY Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
6	DATA2	2	DW	Length of user's status buffer
8	XMIT CORRELATOR	2	DW	reserved
10	RSP CORRELATOR	2	DW	X'nnnn'
12	DEST NAME	16	DB	Name of receiver
28	SOURCE NAME	16	DB	Sending node NAME_NUMBER_1

Data1

Explanation: This is a 1-byte field generated by the sender.

X'nn' indicates the status request, where:

X'nn' = 0	Status request 1.x or 2.0
X'nn' = 1	Initial status request, NETBIOS 2.1
X'nn' > 1	NETBIOS 2.1 request for more names and an indication of the total number of names received so far (names received in the first response and all additional name responses). **Note:** The X'nn' > 1 status request is not transmitted unless the remote station is NETBIOS 2.1.

Data2

Explanation: This is a 2-byte field containing the length of the user's buffer area as defined by the NCB_LENGTH field in the STATUS command .

Response Correlator

Explanation: This is a 2-byte correlator generated by the sender and used to correlate any responses to this query. It is returned in any STATUS_RESPONSE in the transmit correlator field.

Destination Name

Explanation: This is a 16-byte field containing the name on the network for which the status is being requested. This is the name from the NCB_CALLNAME field in the STATUS command.

Source Name

Explanation: This is a 16-byte field that contains the NAME_NUMBER_1, consisting of 10 bytes of binary zeros followed by the 6-byte permanent adapter address for this adapter.

TERMINATE_TRACE

Hex 07

TERMINATE_TRACE

Initiation: This frame is sent as a result of a TRACE command with the "remote trace off" option set (NCB_NUM = X'01').

Function: To terminate the trace at any remote nodes that had activated a trace with the TRACE command.

Transmission: This frame is transmitted as a UI frame to the NETBIOS functional address using single-route broadcast.

Responses: None

Retries: None

Timeouts: None

Table 5-15. TERMINATE_TRACE Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'07'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NAME	16	DB	Reserved
28	SOURCE NAME	16	DB	Reserved

DATAGRAM

Hex 08

DATAGRAM

Initiation: This frame is sent as a result of a SEND.DATAGRAM command.

Function: To transmit a user-specified datagram to a name.

Transmission: This frame is transmitted as a UI frame to the NETBIOS functional address using single-route broadcast.

Responses: None

Retries: None

Timeouts: None

Table 5-16 (Page 1 of 2). DATAGRAM Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'

Table 5-16 (Page 2 of 2). DATAGRAM Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'08'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NAME	16	DB	Name of receiver
28	SOURCE NAME	16	DB	Name of sender
44	USER DATA	nn	DB	Datagram

Destination Name

Explanation: This is a 16-byte field containing the name on the network to which the datagram is being sent. This is the NCB_CALLNAME field in the SEND.DATAGRAM command.

Source Name

Explanation: This is a 16-byte field that contains the name of the originator of the datagram. This is the name associated with the NCB_NUM field in the SEND.DATAGRAM command.

User Data

Explanation: This is the datagram that the user provided with the SEND.DATAGRAM command.

Note: If the adapter is opened by an application program other than NETBIOS, and the size of the transmit buffer is not sufficient for the size of the datagram, the datagram is truncated with no indication given.

DATAGRAM_BROADCAST

Hex 09

DATAGRAM_BROADCAST

Initiation: This frame is sent as a result of a SEND.BROADCAST.DATAGRAM command.

Function: To transmit a user-specified datagram to all names on the network.

Transmission: This frame is transmitted as a UI frame to the NETBIOS functional address using single-route broadcast.

Responses: None

Retries: None

Timeouts: None

Table 5-17. DATAGRAM_BROADCAST Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'09'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NAME	16	DB	Reserved
28	SOURCE NAME	16	DB	Name of sender
44	USER DATA	nn	DB	Datagram

Source Name

Explanation: This is a 16-byte field that contains the name of the originator of the datagram. This is the name associated with the NCB_NUM field in the SEND.BROADCAST.DATAGRAM command.

User Data

Explanation: This is the datagram that the user provided with the SEND.BROADCAST.DATAGRAM command.

Note: If the adapter is opened by an application program other than NETBIOS, and the size of the transmit buffer is not sufficient for the size of the datagram, the datagram is truncated with no indication given.

NAME_QUERY

Hex 0A

NAME_QUERY

Initiation: This frame is sent as a result of a CALL command or a FIND.NAME command.

Function: To request the remote node to establish a session (CALL) or to locate a name on the network (FIND.NAME).

Transmission: This frame is transmitted as a UI frame to the NETBIOS functional address using single-route broadcast.

Responses: No response within the system timeout period indicates the name is not on the network and the command gets an appropriate return code (X'14' for CALL and X'05' for FIND.NAME). A NAME_RECOGNIZED response indicates if a session can be established in the case of a CALL command or indicates the location of a name in the case of the FIND.NAME command. A NAME_RECOGNIZED response for a CALL command terminates the response timer (with a valid LISTEN on the remote side) and the NAME_QUERY is not retried.

Retries: The number of times the frame can be transmitted can be defined by the NCB.TRANSMIT.COUNT parameter. The range is 1 to 10, and the default is 6.

Note: If a NAME_RECOGNIZED with no LISTEN response is received for a CALL command, after transmitting NAME_QUERY frame NCB.TRANSMIT.COUNT times, NAME_QUERY process is retried one more time.

Timeouts: The timeout value between transmissions of this frame can be defined by the NCB.TRANSMIT.TIMEOUT parameter. The range is 1/2 second to 10 seconds, and the default is 1/2 second.

The command timeout for the FIND.NAME case is the value of NCB.TRANSMIT.COUNT multiplied by the value of NCB.TRANSMIT.TIMEOUT. The default is 3 seconds.

Table 5-18. NAME_QUERY Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'0A'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	X'ttss'
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	X'nnnn'
12	DEST NAME	16	DB	Name of receiver
28	SOURCE NAME	16	DB	Name of sender

Data2

Explanation: 'tt' indicates the type of name that is issuing the CALL command, where:

X'00'	A unique name
X'01'	A group name.

'ss' indicates the local session number that is assigned to refer to this session if the CALL is completed. The number is assigned in round-robin fashion (1 to 254). A value of zero is not a valid session number and indicates a FIND.NAME request.

Response Correlator

Explanation: This is a 2-byte field containing a correlator that is generated by the sender and is used to correlate any responses with this query. The value is returned in the NAME_RECOGNIZED response in the transmit correlator field.

Destination Name

Explanation: This is a 16-byte field containing the name on the network that is being called or located. This is the name in the NCB_CALLNAME field of the command.

Source Name

Explanation: This is a 16-byte field containing the local name that is issuing the CALL command. This is the name in the NCB_NAME field of the CALL command. This field is not used if this frame is for a FIND.NAME command ('ss' in DATA2 is zero).

ADD_NAME_RESPONSE

Hex 0D

ADD_NAME_RESPONSE

Initiation: This frame is sent as a response to an ADD_NAME_QUERY or an ADD_GROUP_NAME_QUERY frame.

Function: To indicate that the identified name in the query frame is already in use. This response is generated to an ADD_NAME_QUERY if the name is already registered as either a group name or a unique name. This response is generated to an ADD_GROUP_NAME_QUERY only if the name is already registered as a unique name.

Transmission: This frame is transmitted as a UI frame to the specific address of the node that sent the query.

Responses: None

Retries: None

Timeouts: None

Table 5-19. ADD_NAME_RESPONSE Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'0D'
5	DATA1	1	DB	0 = add name not in process, 1 = add name in process
6	DATA2	2	DW	0 = unique name, 1 = group name
8	XMIT CORRELATOR	2	DW	X'nnnn'
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NAME	16	DB	Name to be added
28	SOURCE NAME	16	DB	Name to be added

Transmit Correlator

Explanation: This is a 2-byte field containing the correlator that was in the response correlator field of the ADD_NAME_QUERY or ADD_GROUP_NAME_QUERY. This is used to correlate the response with the original query at the originator.

Destination Name

Explanation: This is a 16-byte field containing the name that the originator wanted to add.

Source Name

Explanation: This is a 16-byte field containing the name that the originator wanted to add.

NAME_RECOGNIZED

Hex 0E

NAME_RECOGNIZED

Initiation: This frame is sent as a response to a NAME_QUERY frame.

Function: Indicates whether a session can be established with the queried name (CALL) or used to indicate the location of a name (FIND.NAME).

Transmission: This frame is transmitted as a UI frame to the specific address of the node that sent the NAME_QUERY using general broadcast.

Responses: If this NAME_RECOGNIZED frame indicates that no session is to be established, then no response is expected. If this NAME_RECOGNIZED frame indicates that a session is to be established, then a SESSION_INITIALIZE frame is expected.

Retries: If multiple NAME_QUERY frames are received from the same node, then a NAME_RECOGNIZED frame is sent for each NAME_QUERY received. This could happen if the originator of the NAME_QUERY timed out and retried the NAME_QUERY before the NAME_RECOGNIZED was received by the originator.

Timeouts: If the NAME_RECOGNIZED indicates that a session is to be established, then the SESSION_INITIALIZE frame is expected within a timeout period determined by configuration parameters (the value of NCB.TRANSMIT.COUNT multiplied by the value of NCB.TRANSMIT.TIMEOUT). The default is 3 seconds.

Table 5-20. NAME_RECOGNIZED Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'0E'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	X'ttss'
8	XMIT CORRELATOR	2	DW	X'nnnn'
10	RSP CORRELATOR	2	DW	X'nnnn'
12	DEST NAME	16	DB	Name of receiver
28	SOURCE NAME	16	DB	Name of sender

Data2

Explanation:

The 'tt' in DATA2 indicates the type of name as it is registered locally, where:

X'00'	A unique name
X'01'	A group name

The 'ss' in DATA2 indicates the state of the queried name, where:

X'00'	No LISTEN command is pending for this name or this is a FIND.NAME response
X'FF'	A LISTEN command is pending for this name, but there are insufficient resources to establish a session at this time
X'01' - X'FE'	A locally assigned session number that will identify this session when the call process is completed.

Transmit Correlator

Explanation: This is a 2-byte field containing the correlator that was in the response correlator field of the NAME_QUERY frames. This is used to correlate the response with the original query at the originator.

Response Correlator

Explanation: This is a 2-byte field containing a correlator that is generated by the sender. This is used to correlate the SESSION_INITIALIZE frame with this NAME_RECOGNIZED frame. It is returned in the transmit correlator field of the SESSION_INITIALIZE frame. If this NAME_RECOGNIZED frame indicates that no session is to be established, then this field is unused.

Destination Name

Explanation: This is a 16-byte field containing the name at the originator that issued the NAME_QUERY.

Source Name

Explanation: This is a 16-byte field containing the name that was being queried.

STATUS_RESPONSE

Hex 0F

STATUS_RESPONSE

Initiation: This frame is sent as a response to a STATUS_QUERY frame.

Function: Returns the status information of the adapter.

Transmission: This frame is transmitted as a UI frame to the specific address of the node that sent the STATUS_QUERY.

Responses: None

Retries: None

Timeouts: None

Table 5-21. STATUS_RESPONSE Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'0F'
5	DATA1	1	DB	X'nn'
6	DATA2	2	DW	X'xybbbbbbbbbbbbbb'
8	XMIT CORRELATOR	2	DW	X'nnnn'
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NAME	16	DB	Receiver's NAME_NUMBER_1
28	SOURCE NAME	16	DB	Name of sender

Data1

Explanation: This is a 1-byte field generated by the sender.

X'nn' indicates the status response, where:

X'nn' = 0	Status response 1.x or 2.0
X'nn' > 0	NETBIOS 2.1 status response and an indication of the total number of names sent so far (names sent in the first response and additional name responses, including this one).

Data2

Explanation:

'x'	1 indicates the length of the status data exceeds the maximum UI frame size (the data is truncated).
'y'	1 indicates the length of the status data exceeds the size of the user' buffer (sent in the STATUS_QUERY).
'bbb...'	indicates the length of the status data sent (interpreted as a 2-byte (DW) field with the 2 high/ order bits set to zero).

Transmit Correlator

Explanation: This is a 2-byte field containing the correlator that was in the response correlator field of the STATUS_QUERY frames. This is used to correlate the response with the original query at the originator.

Destination Name

Explanation: This is a 16-byte field containing the NAME_NUMBER_1 of the originator of the STATUS_QUERY. This was the source name field of the STATUS_QUERY frame and is 10 bytes of binary zeros followed by the 6-byte permanent adapter address.

Source Name

Explanation: This is a 16-byte field containing the name that was being queried. This was the destination name in the STATUS_QUERY.

TERMINATE_TRACE

Hex 13

TERMINATE_TRACE

Initiation: This frame must be generated by some application program other than NETBIOS. This frame is never sent by NETBIOS, but it will react to the receipt of a X'13'.

Function: To terminate the traces at any remote nodes that had activated a trace with the TRACE command.

Transmission: This frame is transmitted as a UI frame to the NETBIOS functional address using single-route broadcast.

Responses: None

Retries: None

Timeouts: None

Table 5-22. TERMINATE_TRACE Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'002C'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'13'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NAME	16	DB	Reserved
28	SOURCE NAME	16	DB	Reserved

DATA_ACK

Hex 14

DATA_ACK

Initiation: This frame is a response to a DATA_ONLY_LAST frame.

Function: Used to indicate a positive acknowledgment to a received DATA_ONLY_LAST frame.

Transmission: This frame is transmitted as an I frame to the specific address of the node that sent the DATA_ONLY_LAST frame.

Responses: None

Retries: All retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer.

Table 5-23. DATA_ACK Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'14'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	X'nnnn'
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NUMBER	1	DB	Remote session number
13	SOURCE NUMBER	1	DB	Local session number

Transmit Correlator

Explanation: This is a 2-byte field containing the correlator that was in the response correlator field of the DATA_ONLY_LAST frame. This is used to correlate this DATA_ACK frame with the data frame at the originator.

Destination Number

Explanation: This is a 1-byte field containing the session number that was assigned at the remote node during session initialization to identify this session. This value was in the source number field of the DATA_ONLY_LAST frame.

Source Number

Explanation: This is a 1-byte field containing the local session number that was assigned during session initialization to identify this session.

DATA_FIRST_MIDDLE

Hex 15

DATA_FIRST_MIDDLE

Initiation: This frame is transmitted as a result of one of the session SEND commands.

Function: To transfer a user message across a session, where the message has to be segmented because the size of the message exceeds the transmit buffer size or the maximum frame size that was established for this session during session initialization. This is one frame of a message, but not the last one.

Transmission: This frame is transmitted as an I frame to the specific address of the remote session partner.

Responses: If a RECEIVE_CONTINUE frame is requested (on the first DATA_FIRST_MIDDLE), a RECEIVE_CONTINUE frame is returned if all the data was accepted and only partially filled the RECEIVE buffer. If the data completely fills or exceeds the RECEIVE buffer, then a NO_RECEIVE frame is returned.

Note: In NETBIOS Version 2.X or higher, if the data completely fills or exceeds the RECEIVE buffer, and if another RECEIVE is pending for the same session, instead of returning a NO_RECEIVE/RECEIVE_OUTSTANDING sequence, a RECEIVE_OUTSTANDING frame is returned to acknowledge the last byte received, and also to indicate the RECEIVE is available to receive more data.

In NETBIOS Version 2.2 or later, a NO_RECEIVE frame is returned, if the NCB.SEND.NO.ACK data or NCB.CHAIN.SEND.NO.ACK data was not received at all or was only partially received by the remote application program.

Certain NETBIOS application programs (especially those characterized by quick exchanges of short data records) can benefit from the data acknowledgment option that allows the sending of a positive data acknowledgment with the next data transmission. This feature reduces the number of frames to be processed and transmitted. The session partners negotiate the use of this option automatically. It is transparent to the application, and is available in Local Area Network Support Program Version 1.3 or higher and in OS/2 EE and OS/2 LAN Adapter and Protocol Support. The acknowledgment bit in the DATA1 field of the DATA_FIRST_MIDDLE frame is used instead of the DATA_ACK frame if this option is used.

Retries: Since this is an I frame, all retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer. At the NETBIOS level, there may be a SEND timeout in effect for the entire message transfer that was specified in the NCB_STO field of the CALL command .

Table 5-24. DATA_FIRST_MIDDLE Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'15'
5	DATA1	1	DB	B'rrrrxryz' where: x = Acknowledge_included indicator y = NO.ACK indicator (NETBIOS 2.2) z = RECEIVE_CONTINUE request
6	DATA2	2	DW	Re-synch indicator
8	XMIT CORRELATOR	2	DW	X'nnnn'
10	RSP CORRELATOR	2	DW	X'nnnn'
12	DEST NUMBER	1	DB	Remote session number
13	SOURCE NUMBER	1	DB	Local session number
14	USER DATA	nn	DB	Message from SEND

Data1

Explanation:

B'r'	Reserved bit, always zero.
B'x'	Acknowledge_included indicator. The acknowledge_included indicator plus a non-zero XMIT CORRELATOR is equivalent to a DATA_ACK.
B'y'	The NO.ACK indicator (NETBIOS 2.2). The NO.ACK indicator means no acknowledgment is expected. If y = 0, versions earlier than NETBIOS 2.20 are being used. If y = 1, the frame was sent via NCB.SEND.NO.ACK or NCB.CHAIN.SEND.NO.ACK (NETBIOS 2.2).
B'z'	Indicates a RECEIVE_CONTINUE was requested. If z = 0, RECEIVE_CONTINUE was not requested. If z = 1, a RECEIVE_CONTINUE was requested.

The RECEIVE_CONTINUE is requested on the first DATA_FIRST_MIDDLE frame if this is the first message on this session or if any frames in the previous message encountered a NO_RECEIVE response. If all segments of a message are sent without receiving a NO_RECEIVE frame, then the first segment of the next message will not have this request bit set. The RECEIVE_CONTINUE frame indicates that all of the data in this DATA_FIRST_MIDDLE frame was accepted. When the first DATA_FIRST_MIDDLE is sent (with the request indicator on), subsequent segments are not sent until a response is received (RECEIVE_CONTINUE or RECEIVE_OUTSTANDING). This is to avoid sending all message segments only to have them discarded at the remote node if there is no RECEIVE pending or if the RECEIVE buffer cannot accept the entire message.

Data2

Explanation: This is a 2-byte field where a value of X'0001' indicates that this is the first DATA_FIRST_MIDDLE following receipt of a RECEIVE_OUTSTANDING from the remote node. A RECEIVE_OUTSTANDING is sent to indicate the ability to receive more data following a NO_RECEIVE. The indicator is set so the remote node can re-synch on the first valid DATA_FIRST_MIDDLE following a NO_RECEIVE/RECEIVE_OUTSTANDING sequence. After the first DATA_FIRST_MIDDLE is responded to, all remaining segments are transmitted in succession.

Transmit Correlator

Explanation: X'nnnn' is zero when no acknowledgment is sent with the data or X'nnnn' is the previously received response correlator of a DATA_ONLY_LAST frame in which the ACKNOWLEDGE_WITH_DATA_ALLOWED indicator was set on. When an acknowledgment is sent, the ACKNOWLEDGE_INCLUDED bit must be set on.

Response Correlator

Explanation: This is a 2-byte field containing a correlator that is generated by the sender. This is used to correlate the response frame with this frame. The correlator is returned in the transmit correlator field of the RECEIVE_CONTINUE frame.

Destination Number

Explanation: This is a 1-byte field containing the session number that was assigned at the remote node during session initialization to identify this session.

Source Number

Explanation: This is a 1-byte field containing the local session number that was assigned during session initialization to identify this session.

User Data

Explanation: This is a segmented portion of the message that the user provided.

DATA_ONLY_LAST

Hex 16

DATA_ONLY_LAST

Initiation: This frame is transmitted as a result of one of the session send commands.

Function: To transfer a user message across a session. This is either the only frame in this message or this is the last segment of a segmented message.

Transmission: This frame is transmitted as an I frame to the specific address of the remote session partner.

Responses: Responses could be one of the following to acknowledge receipt of this frame or portion of this frame:

DATA_ACK

NO_RECEIVE (returned if the data exceeds the receive buffer)

RECEIVE_OUTSTANDING

DATA_ONLY_LAST or DATA_FIRST_MIDDLE (if the ACKNOWLEDGE_WITH_DATA_ALLOWED indicator was set on and the session partner supports this option).

Certain NETBIOS application programs (especially those characterized by quick exchanges of short data records) can benefit from the data acknowledgment option that allows the sending of a positive data acknowledgment with the next data transmission. This feature reduces the number of frames to be processed and transmitted. The session partners negotiate the use of this option automatically. It is transparent to the application, and is available in Local Area Network Support Program Version 1.3 or higher and in OS/2 EE and OS/2 LAN Adapter and Protocol Support.

Note: In NETBIOS Version 2.2 or later, if the data exceeds the RECEIVE buffer, and if another RECEIVE is pending for the same session, instead of returning the

NO_RECEIVE/RECEIVE_OUTSTANDING sequence, a RECEIVE_OUTSTANDING frame is returned to acknowledge the last byte received and also to indicate that the RECEIVE is available to receive more data.

In NETBIOS Version 2.2 or later, a NO_RECEIVE frame is returned if the NCB.SEND.NO.ACK data or NCB.CHAIN.SEND.NO.ACK data was not received at all or was only partially received by the remote application program.

Retries: Since this is an I frame, all retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer. At the NETBIOS level there may be a SEND timeout in effect for the entire message transfer that was specified in the NCB_STO field of the CALL command.

Table 5-25. DATA_ONLY_LAST Frame Format

Offset	**Field Name**	**Byte Length**	**8086 Type**	**Description**
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'16'
5	DATA1	1	DB	B'rrrrxyzr' where: x = ACKNOWLEDGE_INCLUDED indicator y = ACKNOWLEDGE_WITH_ DATA_ALLOWED indicator z = NO.ACK indicator
6	DATA2	2	DW	Re-synch indicator
8	XMIT CORRELATOR	2	DW	X'nnnn'
10	RSP CORRELATOR	2	DW	X'nnnn'
12	DEST NUMBER	1	DB	Remote session number
13	SOURCE NUMBER	1	DB	Local session number
14	USER DATA	nn	DB	Message from SEND

Data1

Explanation:

B'r'=0 Reserved bit, always zero.

B'x' ACKNOWLEDGE_INCLUDED indicator.

B'y' ACKNOWLEDGE_WITH_DATA_ALLOWED indicator.

The session partner can acknowledge either with a DATA_ACK frame, or with a frame in which the acknowledge_included indicator is set, and the XMIT CORRELATOR is the same as the RSP CORRELATOR in this DATA_ONLY_LAST frame.

B'z' NO.ACK indicator.

Data2

Explanation: This is a 2-byte field where a value of X'0001' indicates that this is the first DATA_ONLY_LAST following receipt of a RECEIVE_OUTSTANDING (otherwise this field is X'0000'). The RECEIVE_OUTSTANDING is sent to indicate the ability to receive more data following a NO_RECEIVE.

Transmit Correlator

Explanation: X'nnnn' is zero either when no acknowledgment is sent with the data or when X'nnnn' is the previously received RSP CORRELATOR of a DATA_ONLY_LAST frame in which the ACKNOWLEDGE_WITH_DATA_ALLOWED indicator was set on. When an acknowledgment is sent, the ACKNOWLEDGE_INCLUDED bit must be set on.

Destination Number

Explanation: This is a 1-byte field containing the session number that was assigned at the remote node during session initialization to identify this session.

Source Number

Explanation: This is a 1-byte field containing the local session number that was assigned during session initialization to identify this session.

User Data

Explanation: This is the last or only segment of the message that the user provided.

SESSION_CONFIRM

Hex 17

SESSION_CONFIRM

Initiation: This frame is transmitted in response to a SESSION_INITIALIZE frame.

Function: To indicate a positive acknowledgment to a SESSION_INITIALIZE to complete the session establishment process.

Transmission: This frame is transmitted as an I frame to the specific address of the remote session partner.

Responses: None

Retries: Since this is an I frame, all retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer.

Table 5-26 (Page 1 of 2). SESSION_CONFIRM Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'

Table 5-26 (Page 2 of 2). SESSION_CONFIRM Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
4	COMMAND	1	DB	X'17'
5	DATA1	1	DB	B'yrrrrrrx'
6	DATA2	2	DW	Max data receive size
8	XMIT CORRELATOR	2	DW	X'nnnn'
10	RSP CORRELATOR	2	DW	X'nnnn' = Session init xmit correlator
12	DEST NUMBER	1	DB	Remote session number
13	SOURCE NUMBER	1	DB	Local session number

Data1

Explanation: This is a 1-byte field generated by the sender.

B'rrrrrr'	Reserved bits, always zero
B'x' = 0	NETBIOS 1.xx
B'x' = 1	NETBIOS Version 2.00 or higher
B'y' = 0	Versions earlier than NETBIOS 2.20
B'y' = 1	Flag to indicate the ability to handle SEND.NO.ACK or CHAIN.SEND.NO.ACK.

Data2

Explanation: This is a 2-byte field containing the maximum size user data in any frame that this node wants to receive on this session. NETBIOS determines this value based on the size and number of receive buffers available in the adapter such that this value is approximately one-half of the receive buffer space. The remote partner will limit the size of the user data in frames transmitted over this session to this size or the size available in its transmit buffer (DHB), whichever is smaller. This is to avoid having the receipt of one session frame consume all of the available receive buffers in the adapter. NETBIOS takes into account the maximum size message that bridges in the path will forward. It will never send a frame larger than the bridge will forward.

Transmit Correlator

Explanation: This is a 2-byte field containing the correlator that was in the response correlator field of the SESSION_INITIALIZE frame. This is used to correlate this response with the SESSION_INITIALIZE frame at the originator.

Destination Number

Explanation: This is a 1-byte field containing the session number that was assigned at the remote node during session initialization to identify this session.

Source Number

Explanation: This is a 1-byte field containing the local session number that was assigned during session initialization to identify this session.

SESSION_END

Hex 18

SESSION_END

Initiation: This frame is transmitted as a result of a HANG.UP command, a SEND command that timed out, or some abnormal condition.

Function: To indicate the termination of a session to the remote partner.

Transmission: This frame is transmitted as an I frame to the specific address of the remote session partner.

Responses: None

Retries: Since this is an I frame, all retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer.

Table 5-27. SESSION_END Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'18'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Termination indicator
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NUMBER	1	DB	Remote session number
13	SOURCE NUMBER	1	DB	Local session number

Data2

Explanation: This is a 2-byte field used to indicate the type of termination, where X'0000' indicates a normal session end (as a result of a HANG.UP command) and X'0001' indicates an abnormal session end (typically a SEND command has timed out).

Destination Number

Explanation: This is a 1-byte field containing the session number that was assigned at the remote node during session initialization to identify this session.

Source Number

Explanation: This is a 1-byte field containing the local session number that was assigned during session initialization to identify this session.

SESSION_INITIALIZE

Hex 19

SESSION_INITIALIZE

Initiation: This frame is transmitted in response to a NAME_RECOGNIZED frame that indicates that a session is to be established.

Function: To indicate that a session has been established.

Transmission: This frame is transmitted as an I frame to the specific address of the remote session partner.

Responses: A SESSION_CONFIRM is expected to indicate acknowledgment of the session establishment.

Retries: Since this is an I frame, all retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer. At the NETBIOS level, the SESSION_CONFIRM is expected within a timeout period equal to the value of NCB.TRANSMIT.COUNT multiplied by the value of NCB.TRANSMIT.TIMEOUT plus 1 second per bridge. The default is 3 seconds.

Table 5-28. SESSION_INITIALIZE Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'19'
5	DATA1	1	DB	B'zrrrxxxy'
6	DATA2	2	DW	Maximum data receive size
8	XMIT CORRELATOR	2	DW	X'nnnn'
10	RSP CORRELATOR	2	DW	X'nnnn'
12	DEST NUMBER	1	DB	Remote session number
13	SOURCE NUMBER	1	DB	Local session number

Data1

Explanation: This is a 1-byte field generated by the sender.

B'rrrr'	Reserved bits, always zero
B'xxx'	Indicates the largest frame value as seen by the MAC layer (LF field in the routing information)
B'y' = 0	NETBIOS 1.x

B'y' = **1**	NETBIOS Version 2.0 or higher
B'z' = **0**	Versions earlier than NETBIOS 2.20
B'z' = **1**	Flag to indicate the ability to handle SEND.NO.ACK or CHAIN.SEND.NO.ACK

Data2

Explanation: This is a 2-byte field containing the maximum size user data in any frame that this node wants to receive on this session. NETBIOS determines this value based on the size and number of receive buffers available in the adapter such that this value is approximately one-half of the receive buffer space. The remote partner will limit the size of the user data in frames transmitted over this session to this size or the size available in its transmit buffer (DHB), whichever is smaller. This is to avoid having the receipt of one session frame consume all of the available receive buffers in the adapter. NETBIOS takes into account the maximum size message that bridges in the path will forward. It will never send a frame larger than the bridge will forward.

Transmit Correlator

Explanation: This is a 2-byte field containing the correlator that was in the response correlator field of the NAME_RECOGNIZED frame. This is used to correlate this frame with the NAME_RECOGNIZED frame at the remote node.

Response Correlator

Explanation: This is a 2-byte field containing a correlator that is generated by the sender. This is used to correlate the SESSION_CONFIRM frame with this SESSION_INITIALIZE frame. It is returned in the transmit correlator field of the SESSION_CONFIRM frame.

Destination Number

Explanation: This is a 1-byte field containing the session number that was assigned at the remote node during session initialization to identify this session.

Source Number

Explanation: This is a 1-byte field containing the local session number that was assigned during session initialization to identify this session.

NO_RECEIVE

Hex 1A

NO_RECEIVE

Initiation: This frame is transmitted as a result of receiving a DATA_ONLY_LAST or a DATA_FIRST_MIDDLE frame.

Function: To acknowledge receipt of session data, and to indicate how much of the data was accepted. This frame is sent when the RECEIVE buffer is filled. Upon receipt of this frame at the remote node, the remote node discontinues sending session data on this session until the local node transmits a RECEIVE_OUTSTANDING frame. Any session data that continues to arrive at the local node is ignored until a DATA_FIRST_MIDDLE or DATA_ONLY_LAST frame arrives with the re-synch indicator set (which means the local node has sent a RECEIVE_OUTSTANDING frame).

Notes:

1. When a data frame is received, if there is no RECEIVE command pending then nothing is transmitted to the sender. When a RECEIVE command is issued, then a RECEIVE_OUTSTANDING frame is transmitted.
2. In NETBIOS Version 2.X or higher, if the data completely fills or exceeds the RECEIVE buffer, and if another RECEIVE is pending for the same session, instead of returning a NO_RECEIVE/RECEIVE_OUTSTANDING sequence, a RECEIVE_OUTSTANDING frame is returned to acknowledge the last byte received and also to indicate the RECEIVE is available to receive more data.

Transmission: This frame is transmitted as an I frame to the specific address of the remote session partner.

Responses: None

Retries: Since this is an I frame, all retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer.

Table 5-29. NO_RECEIVE Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'1A'
5	DATA1	1	DB	B'rrrrrxr'
6	DATA2	2	DW	Number of data bytes accepted
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NUMBER	1	DB	Remote session number
13	SOURCE NUMBER	1	DB	Local session number

Data1

Explanation:

B'rrrr'	Reserved bit, always zero
B'x' = 0	Versions earlier than NETBIOS 2.20
B'x' = 1	Flag to indicate that NCB.SEND.NO.ACK data or NCB.CHAIN.SEND.NO.ACK data was either not received or was only partially received (NETBIOS 2.2).

Data2

Explanation: This is a 2-byte field containing the number of bytes of user data that were accepted. When the remote node is instructed to resume transmitting session data (through a RECEIVE_OUTSTANDING frame), it resumes with the next byte following the last acknowledged byte.

Destination Number

Explanation: This is a 1-byte field containing the session number that was assigned at the remote node during session initialization to identify this session.

Source Number

Explanation: This is a 1-byte field containing the local session number that was assigned during session initialization to identify this session.

RECEIVE_OUTSTANDING

Hex 1B

RECEIVE_OUTSTANDING

Initiation: This frame is transmitted following transmission of a NO_RECEIVE if a RECEIVE command is or becomes available.

Function: To inform the remote session partner that there is a RECEIVE pending and that the remote partner should resume sending session data. The remote node resumes with the first data byte following the last acknowledged byte indicated in the NO_RECEIVE frame. The remote node indicates the first session data frame following the RECEIVE_OUTSTANDING frame by setting the re-synch indicator.

Transmission: This frame is transmitted as an I frame to the specific address of the remote session partner.

Responses: After transmitting this frame, the local node is expecting a DATA_ONLY_LAST or DATA_FIRST_MIDDLE frame with the re-synch indicator set.

Retries: Since this is an I frame, all retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer.

Table 5-30 (Page 1 of 2). RECEIVE_OUTSTANDING Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'1B'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Number of data bytes accepted
8	XMIT CORRELATOR	2	DW	Reserved

Table 5-30 (Page 2 of 2). RECEIVE_OUTSTANDING Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NUMBER	1	DB	Remote session number
13	SOURCE NUMBER	1	DB	Local session number

DATA2

Explanation: This field is defined for NETBIOS 2.1. For earlier versions, this is a reserved field.

This is a 2-byte field containing the number of bytes of user data that were accepted. When the remote node is instructed to resume transmitting session data (through a RECEIVE_OUTSTANDING frame), it resumes resume with the next byte following the last acknowledged byte.

Destination Number

Explanation: This is a 1-byte field containing the session number that was assigned at the remote node during session initialization to identify this session.

Source Number

Explanation: This is a 1-byte field containing the local session number that was assigned during session initialization to identify this session.

RECEIVE_CONTINUE

Hex 1C

RECEIVE_CONTINUE

Initiation: This frame is transmitted in response to a DATA_FIRST_MIDDLE that had the RECEIVE_CONTINUE request bit set.

Function: To indicate that there is a RECEIVE command pending on this session. It also acknowledges receipt of all the session data in the DATA_FIRST_MIDDLE (a NO_RECEIVE would be transmitted if the data filled or exceeded the RECEIVE buffer).

Note: In NETBIOS Version 2.X or higher, if the data completely fills or exceeds the RECEIVE buffer, and if another RECEIVE is pending for the same session, instead of returning a NO_RECEIVE/RECEIVE_OUTSTANDING sequence, a RECEIVE_OUTSTANDING frame is returned to acknowledge the last byte received and also to indicate the RECEIVE is available to receive more data.

Transmission: This frame is transmitted as an I frame to the specific address of the remote session partner.

Responses: None

Retries: Since this is an I frame, all retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer.

Table 5-31. RECEIVE_CONTINUE Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'1C'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	X'nnnn'
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NUMBER	1	DB	Remote session number
13	SOURCE NUMBER	1	DB	Local session number

Transmit Correlator

Explanation: This is a 2-byte field containing the correlator that was in the response correlator field of the DATA_FIRST_MIDDLE frame. This is used to correlate this frame with the DATA_FIRST_MIDDLE frame at the originator.

Destination Number

Explanation: This is a 1-byte field containing the session number that was assigned at the remote node during session initialization to identify this session.

Source Number

Explanation: This is a 1-byte field containing the local session number that was assigned during session initialization to identify this session.

SESSION_ALIVE

Hex 1F

SESSION_ALIVE

Initiation: This frame is transmitted as a result of the periodic expiration of a link inactivity timer.

Function: To periodically determine, in the absence of session data, if the links to any remote nodes for which sessions are established are still available. This is to prevent keeping resources allocated when a link may have become unavailable without indication. A bad return code on the transmit of this frame will cause the user to be informed of abnormal session termination, and the resources for every session on this link will be released. Receipt of this frame is ignored.

Transmission: This frame is transmitted as an I frame to the specific address of the remote session partner.

Responses: None

Retries: Since this is an I frame, all retries are handled by the IEEE 802.2 LLC layer.

Timeouts: Retry timeouts are handled by the IEEE 802.2 LLC layer. At the NETBIOS level, the periodic link inactivity timer is 30 seconds.

Table 5-32. SESSION_ALIVE Frame Format

Offset	Field Name	Byte Length	8086 Type	Description
0	LENGTH	2	DW	X'000E'
2	DELIMITER	2	DW	X'EFFF'
4	COMMAND	1	DB	X'1F'
5	DATA1	1	DB	Reserved
6	DATA2	2	DW	Reserved
8	XMIT CORRELATOR	2	DW	Reserved
10	RSP CORRELATOR	2	DW	Reserved
12	DEST NUMBER	1	DB	Reserved
13	SOURCE NUMBER	1	DB	Reserved

NETBIOS Protocol Examples Without RND

Following are NETBIOS 2.1 examples of typical protocol scenarios including name management, session establishment, and session data transfer. The direction of a frame on the network is indicated by an arrow. The frame type is above the arrow and the contents of the frame are below the arrow. If a frame is repeated, the data appears only on the first instance of the frame.

Notes:

1. The numeric values shown in the various examples indicate the true number; in a trace, the high-order and low-order bytes may be reversed because of 8086 architecture.
2. The use of 'rrrr' within certain fields of the frame indicates a reserved field.

Name Management Examples

Following are scenarios that show the adding of a name to the network.

Add a Name to the Network

The application program attempts to add a unique name to the network. The ADD_NAME_QUERY is sent at 1/2 second intervals for 6 times (by default). If no response is received in that period of time, the name is assumed to be unique and is added to the name table. The application program NCB gets a return code of X'00'. See an example of the command sequence in Figure 5-5.

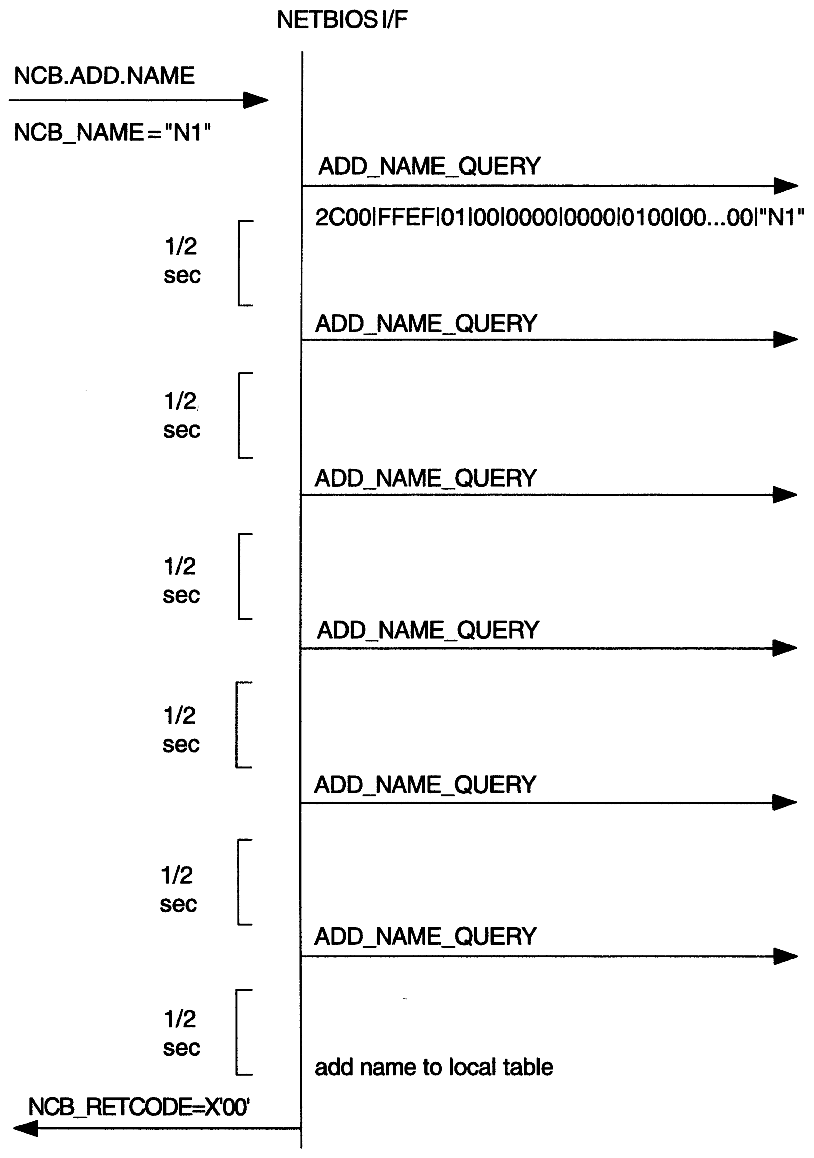


Figure 5-5. Add a Name to the Network Command Sequence

Add a Name: Name Already On Network

The application program attempts to add a unique name to the network, and the name is already registered at some remote node. The remote node responds to the ADD_NAME_QUERY with an ADD_NAME_RESPONSE. The originating node stops transmitting ADD_NAME_QUERY when a response is received and waits for the total timeout period checking for responses from other nodes. If no other nodes respond, the NCB gets a return code of X'16' (name in use on remote NETBIOS). See an example of the command sequence in Figure 5-6.

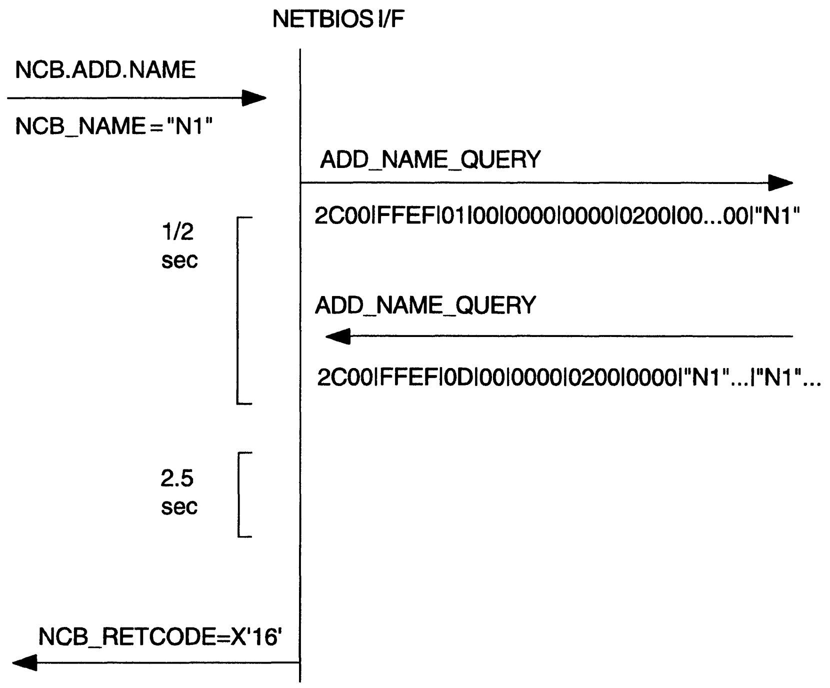
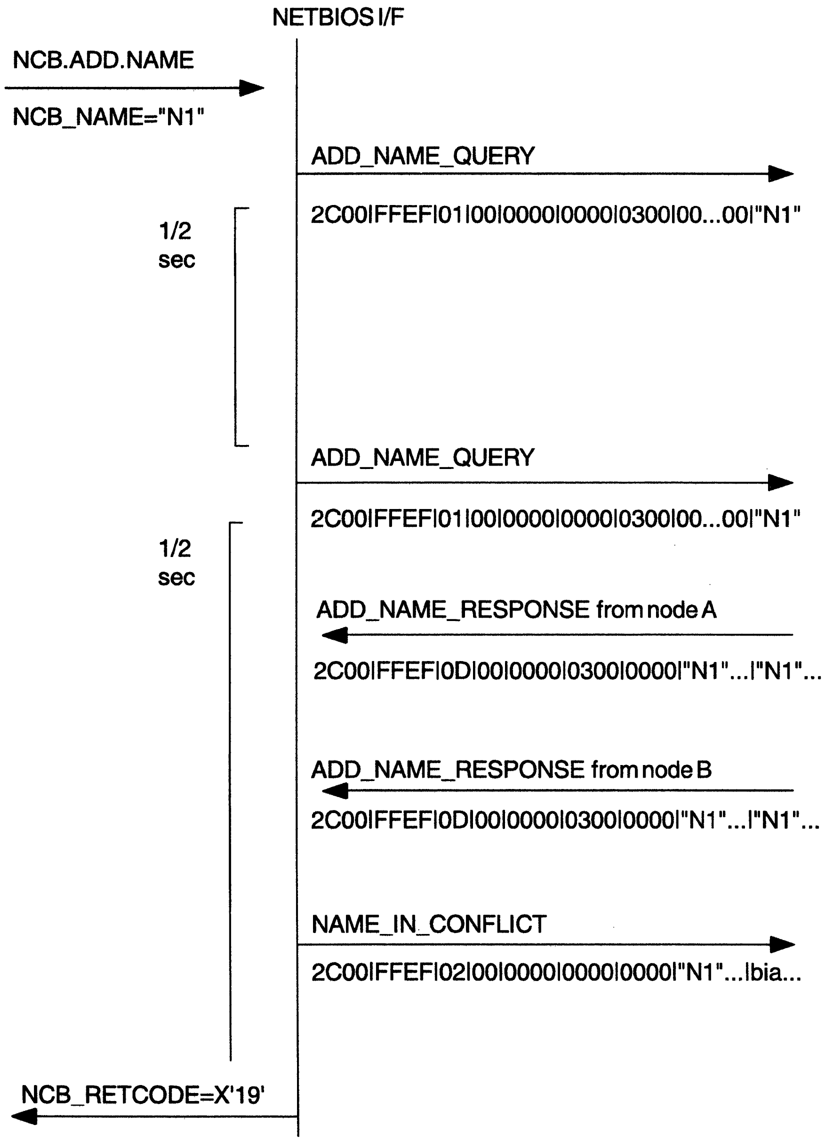


Figure 5-6. Name Already On Network Command Sequence

Add a Name: Receive Multiple Responses

The application program attempts to add a unique name to the network and the name is registered at more than one remote node (this is an error situation). The originating node stops transmitting ADD_NAME_QUERY when a response is received and waits for the total timeout period, checking for responses from other nodes. When more than one response is received from a different node, the NAME_IN_CONFLICT is detected by the originating node and sends a NAME_IN_CONFLICT to all remote nodes and no longer waits for the timeout period. The NCB gets a return code of X'19' (name conflict detected). See an example of the command sequence in Figure 5-7.



NETBIOS I/F
NCB.ADD.NAME
NCB_NAME="N1"
ADD_NAME_QUERY
2C00IFFEFI01I00I0000I0000I0300I00...00I"N1"
1/2 sec
ADD_NAME_QUERY
2C00IFFEFI01I00I0000I0000I0300I00...00I"N1"
1/2 sec
ADD_NAME_RESPONSE from node A
2C00IFFEFI0DI00I0000I0300I0000I"N1"...I"N1"...
ADD_NAME_RESPONSE from node B
2C00IFFEFI0DI00I0000I0300I0000I"N1"...I"N1"...
NAME_IN_CONFLICT
2C00IFFEFI02I00I0000I0000I0000I"N1"...Ibia...
NCB_RETCODE=X'19'

Figure 5-7. Receive Multiple Responses Command Sequence

Remote Adapter Status Examples

Following are examples of remote status requests.

Remote Adapter Status for a Name That Is Not on the Network

The application program attempts to request the remote status for a name that is not on the network. STATUS_QUERY is sent at 1/2 second intervals for 6 times (by default). If a STATUS_RESPONSE is not received the application program NCB gets a return code of X'05' (command timed out). See an example of the command sequence in Figure 5-8.

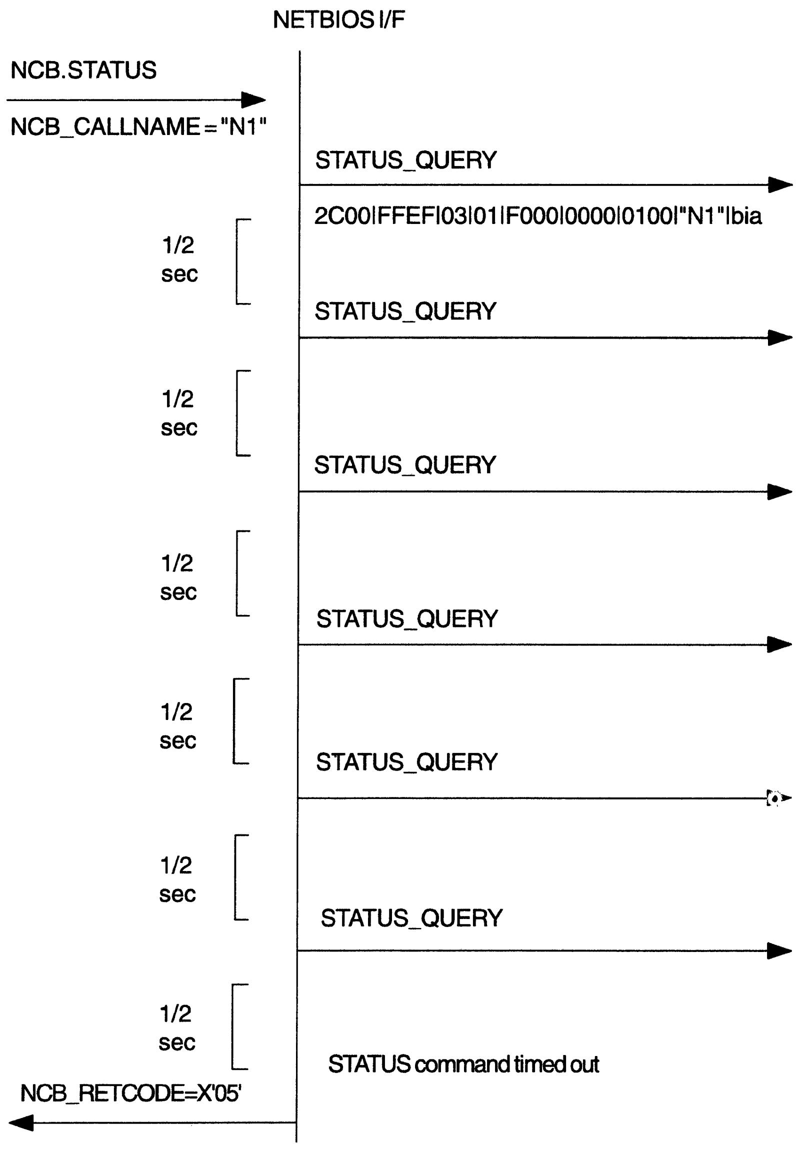


Figure 5-8. Remote Adapter Status for a Name That Is Not on the Network Command Sequence

Remote Adapter Status for a Name That Is On the Network

The application program attempts to request the remote status for a name that is on the network. STATUS_QUERY is sent at 1/2 second intervals for 6 times (by default). If a STATUS_RESPONSE is received the application program NCB gets a return code of X'00'. It is assumed that the application program buffer is big enough to hold the status data. See an example of the command sequence in Figure 5-9.

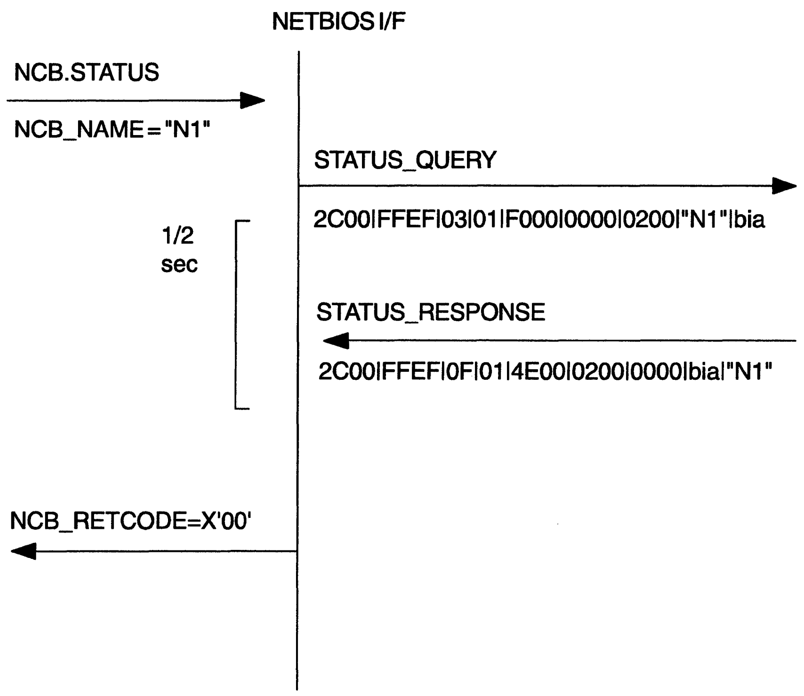
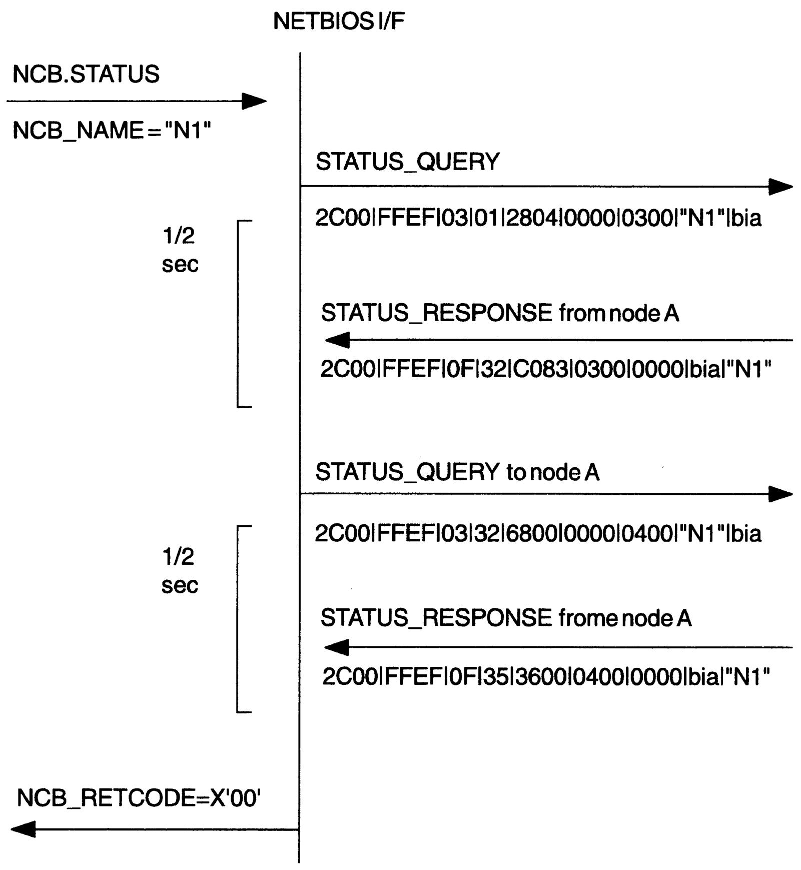


Figure 5-9. Remote Adapter Status for a Name that Is on the Network Command Sequence

Remote Adapter Status Data: Segmentation

The application program attempts to request the remote status for a name that is on the network. STATUS_QUERY is sent at 1/2 second intervals for 6 times (by default). The example shown in Figure 5-10 describes how the status data are segmented. It is assumed that the remote station has 63 names and the maximum UI frame that the remote station can transmit is 962 bytes. The application program puts up a buffer for 1064 bytes. The application program NCB gets a return code of X'00'. The command sequence is shown in Figure 5-10.



NETBIOS I/F
NCB.STATUS
NCB_NAME="N1"
STATUS_QUERY
2C00|FFEF|03|01|2804|0000|0300|"N1"|bia
1/2 sec
STATUS_RESPONSE from node A
2C00|FFEF|0F|32|C083|0300|0000|bia|"N1"
STATUS_QUERY to node A
2C00|FFEF|03|32|6800|0000|0400|"N1"|bia
1/2 sec
STATUS_RESPONSE frome node A
2C00|FFEF|0F|35|3600|0400|0000|bia|"N1"
NCB_RETCODE=X'00'

Figure 5-10. Remote Adapter Status Data: Segmentation Command Sequence

Session Establishment Examples

Following are scenarios that show session establishment.

Call a Name: Name Not on Network

The application program attempts to call a name that does not exist on the network. The NAME_QUERY is sent at 1/2 second intervals for 6 times (by default). If a NAME_RECOGNIZED is not received, the application program NCB gets a return code of X'14' (cannot find name or no answer). See an example of the command sequence in Figure 5-11.

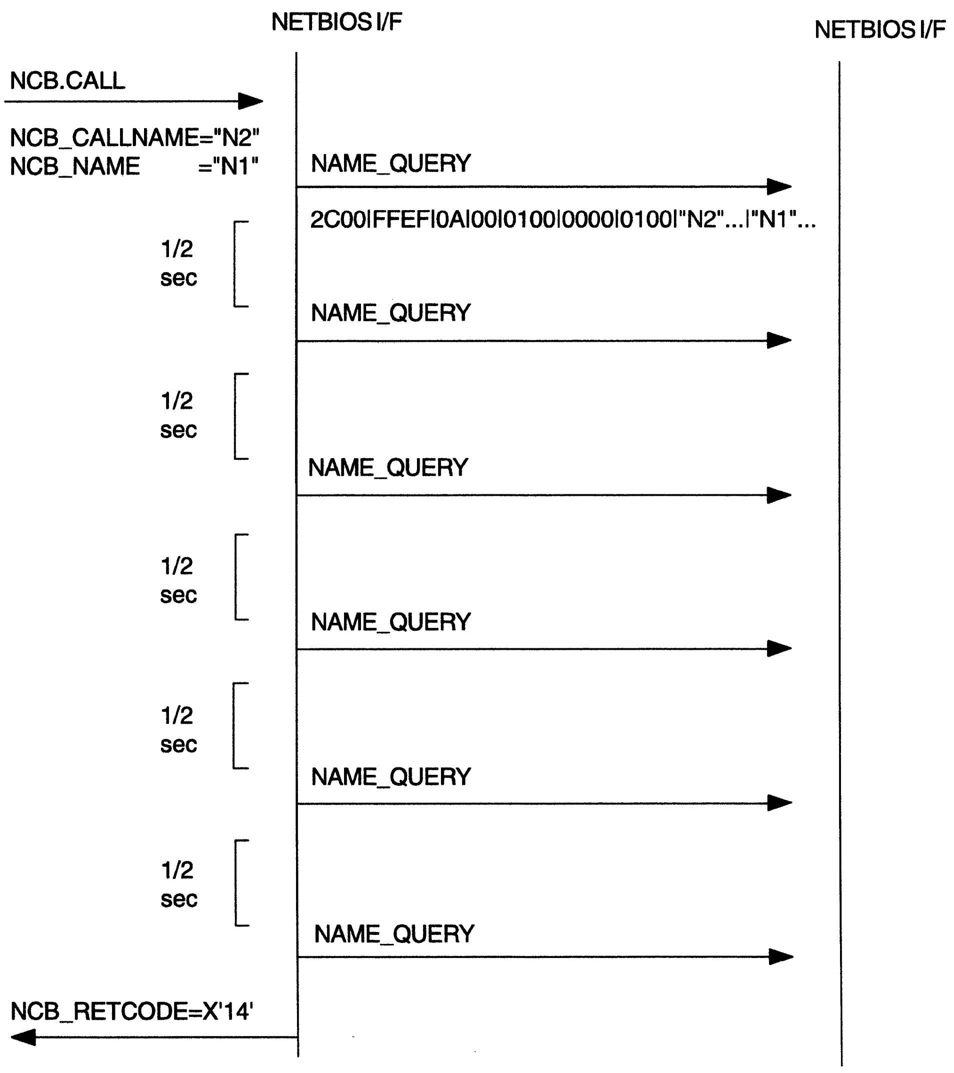


Figure 5-11. Call a Name: Name Not on Network Command Sequence

Call a Name: Name on Network but No Listen

The application program attempts to call a name on the network that is registered but there is no LISTEN pending for that name. The NAME_QUERY is sent at 1/2 second intervals for 6 times (by default). If a NAME_RECOGNIZED is received that indicates no LISTEN, the query cycle starts over. If a good NAME_RECOGNIZED is still not received, the application program NCB gets a return code of X'12' (session open rejected). See an example of the command sequence in Figure 5-12.

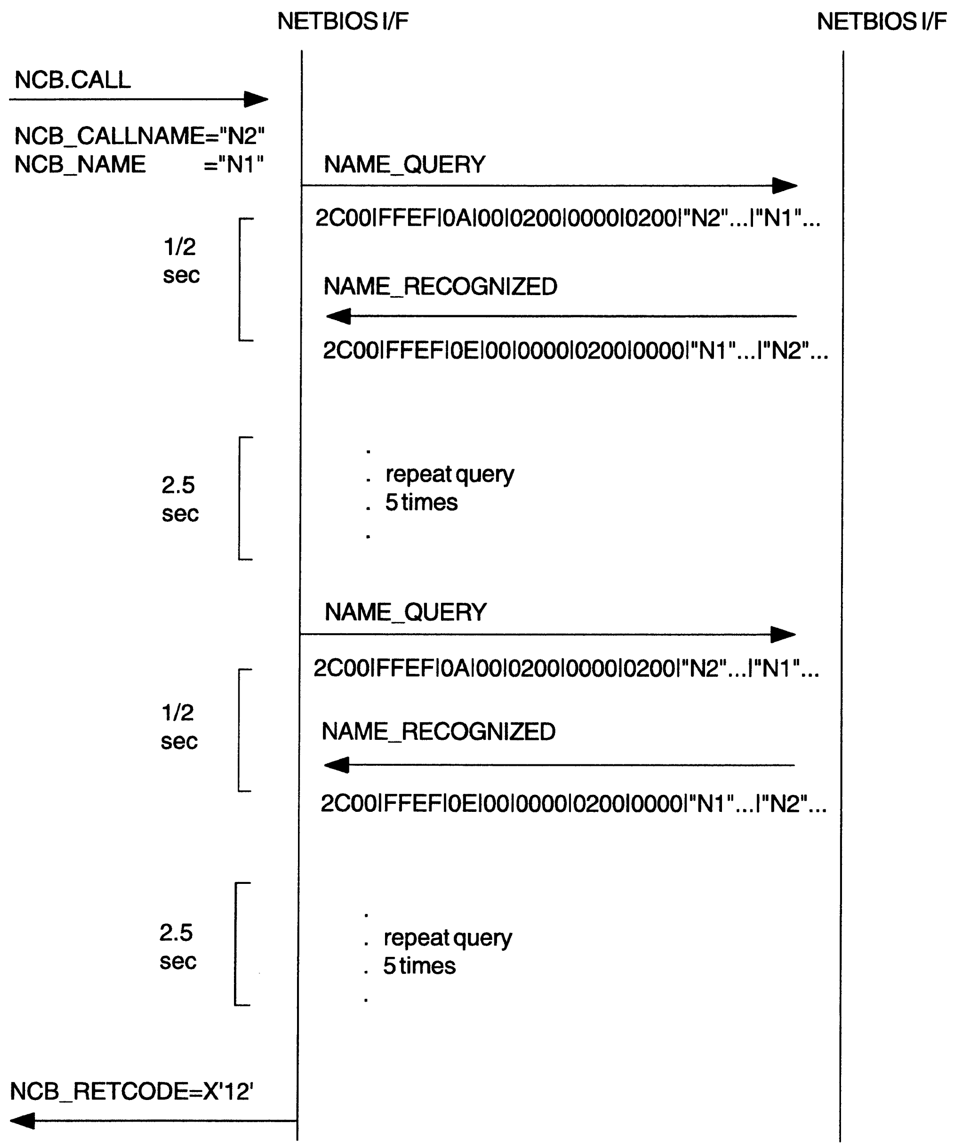


Figure 5-12. Call a Name: Name on Network but No Listen Command Sequence

Call a Name: Name Found—Start Session

The application program uses name N1 and issues a call to name N2 that is registered at a remote node that has a LISTEN pending for that name. The NAME_QUERY and NAME_RECOGNIZED contain the respective session numbers. Receipt of the NAME_RECOGNIZED causes the originating node to send a SESSION_INITIALIZE (as an I-frame) and run a timer waiting for a SESSION_CONFIRM (sent as an I-frame). The SESSION_INITIALIZE and SESSION_CONFIRM each indicate the maximum amount of user data that they are prepared to receive on this session. The remote partner then limits the size of the user data in frames transmitted over this session to this size or the size available in its transmit buffer (DHB), whichever is smaller. Receipt of the SESSION_CONFIRM completes the call and sets the NCB_RETCODE to X'00' and NCB_LSN to the locally assigned session number. See an example of the command sequence in Figure 5-13.

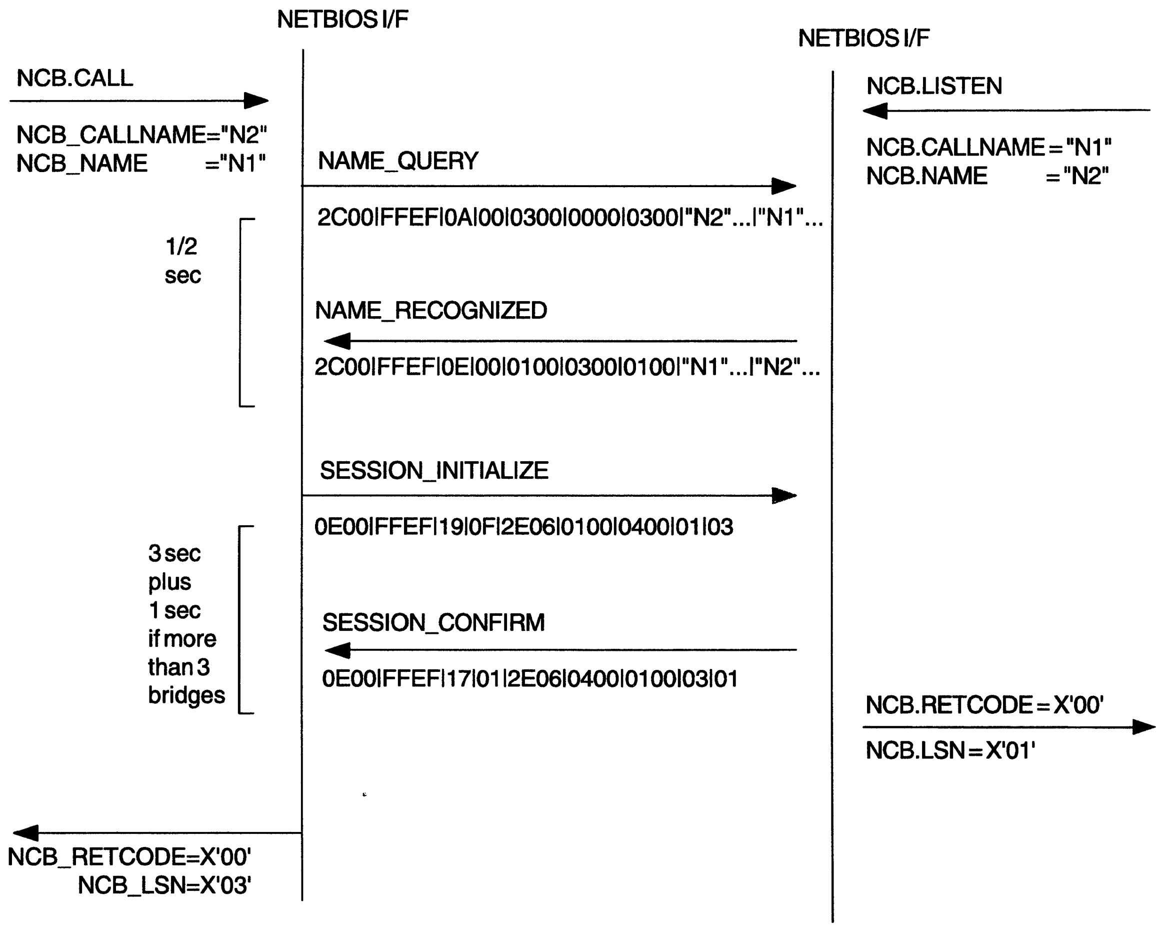


Figure 5-13. Call a Name: Name Found—Start Session Command Sequence. If the call is to a group name, then the first group name to respond with a name recognized will be the station with which the session is established.

Session Data Transfer Examples

The following scenarios show data transfer across a session including segmentation at the sender and multiple receives at the receiver.

Send Session Data: One Send and One Receive

The session has been established as above. The local node issues a SEND command for 100 bytes and the remote node has a corresponding 100-byte RECEIVE pending. The local node sends a DATA_ONLY_LAST and the remote node responds with a DATA_ACK. See an example of the command sequence in Figure 5-14.

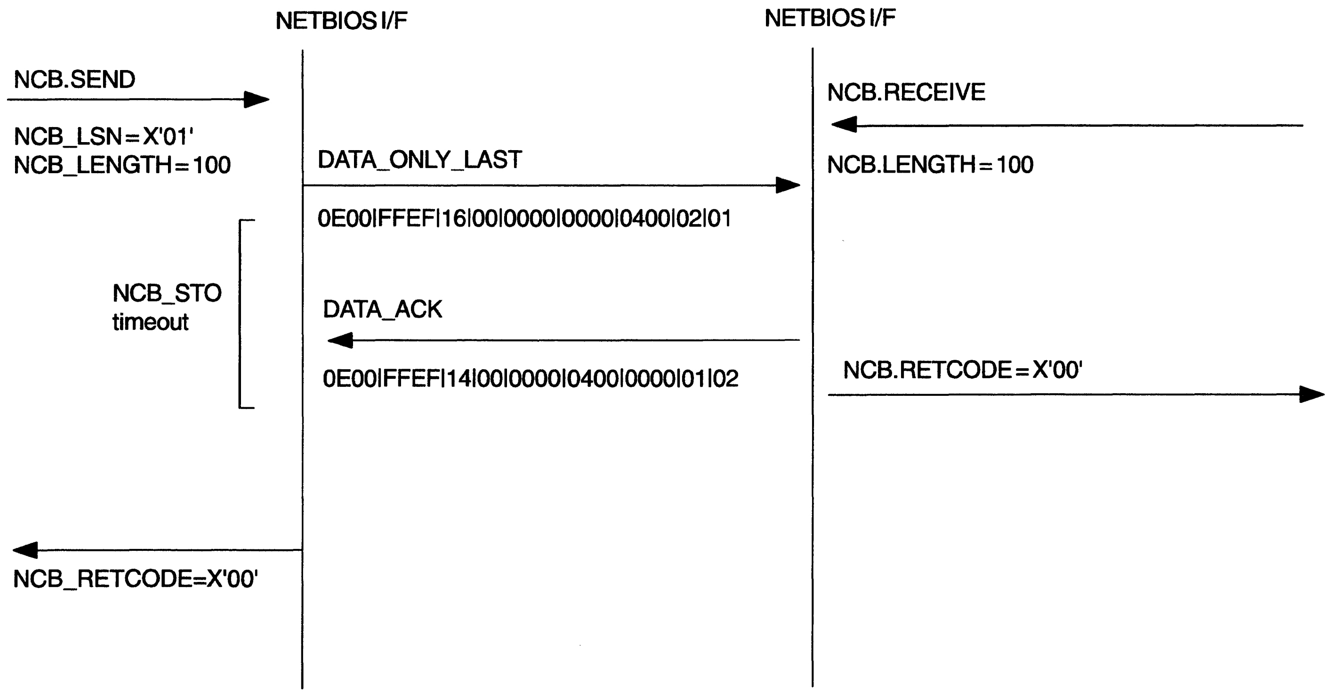


Figure 5-14. Send Session Data: One Send and One Receive Command Sequence

Send Session Data: One Send and Multiple Receives

The local node sends a 200-byte message over the session to the remote node. The remote node has two 100-byte RECEIVEs outstanding. The remote node sends a RECEIVE_OUTSTANDING, acknowledging 100 bytes and indicating another RECEIVE is pending. The local node resends until the remote has acknowledged all the data. See an example of the command sequence in Figure 5-15.

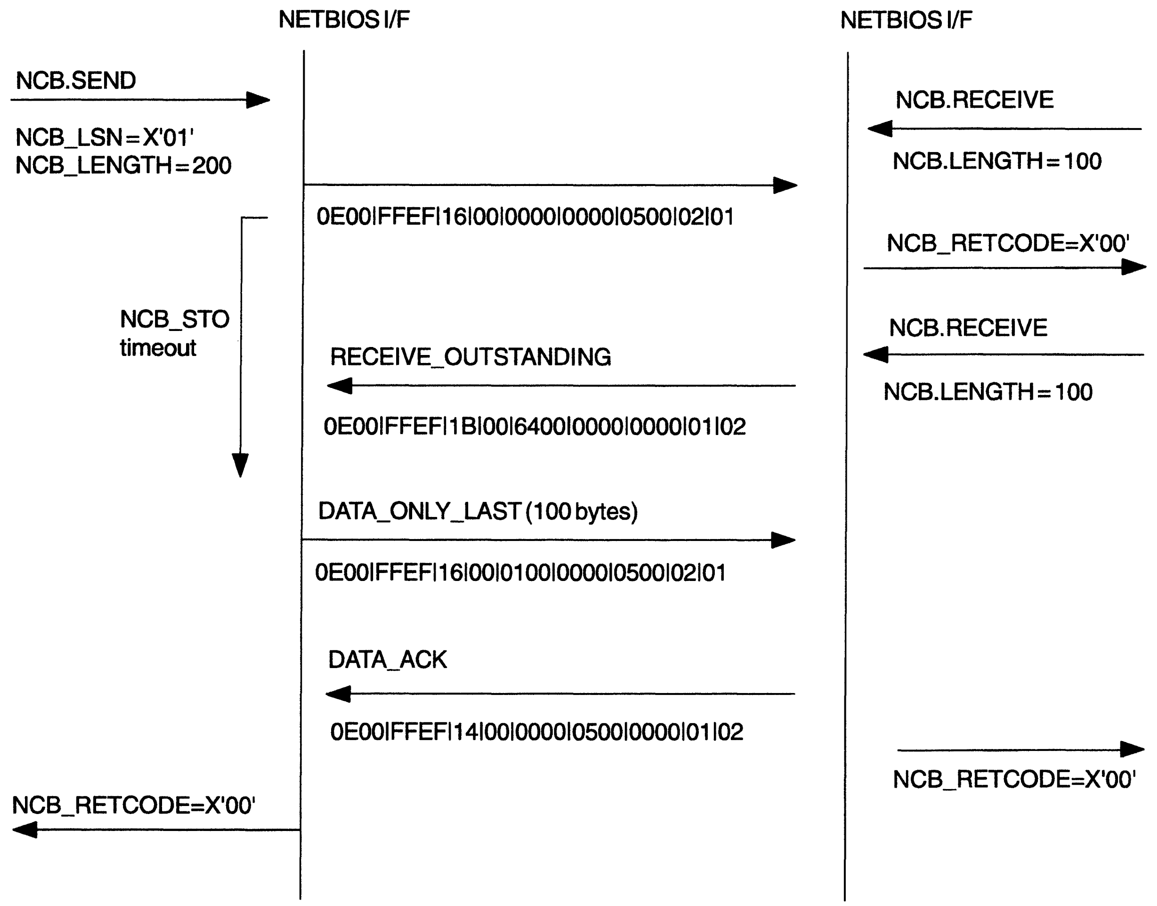
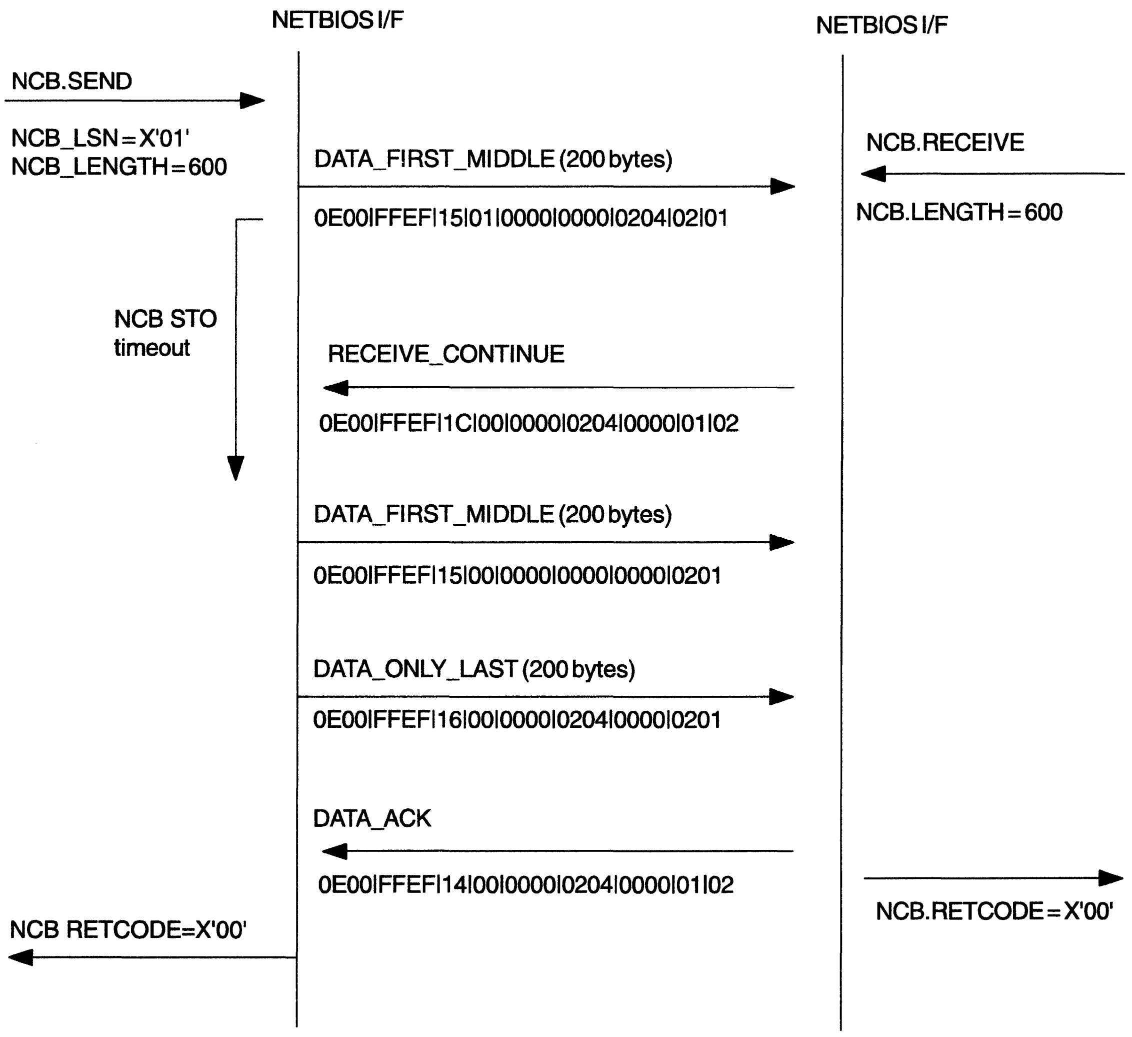


Figure 5-15. Send Session Data: One Send and Multiple Receives Command Sequence

Send Session Data: Segmentation and One Receive

The local node sends a 600-byte message over the session to the remote node that has a corresponding 600-byte RECEIVE pending. The local DHB size is such that the message must be segmented into 200-byte segments. The local node sends the first 200-byte segment with a bit requesting a RECEIVE_CONTINUE from the remote node indicating it has a RECEIVE pending. The remote node responds with the RECEIVE_CONTINUE that indicates acceptance of the 200-byte segment and indicates it can receive more data. The local node then sends the remainder. See an example of the command sequence in Figure 5-16.



NETBIOS I/F
NETBIOS I/F
NCB.SEND
NCB_LSN=X'01'
NCB_LENGTH=600
NCB STO
timeout
DATA_FIRST_MIDDLE (200 bytes)
0E00|FFEF|15|01|0000|0000|0204|02|01
NCB.RECEIVE
NCB.LENGTH=600
RECEIVE_CONTINUE
0E00|FFEF|1C|00|0000|0204|0000|01|02
DATA_FIRST_MIDDLE (200 bytes)
0E00|FFEF|15|00|0000|0000|0000|0201
DATA_ONLY_LAST (200 bytes)
0E00|FFEF|16|00|0000|0204|0000|0201
DATA_ACK
0E00|FFEF|14|00|0000|0204|0000|01|02
NCB RETCODE=X'00'
NCB.RETCODE=X'00'

Figure 5-16. Send Session Data: Segmentation and One Receive Command Sequence

Send Session Data: Segmentation and Multiple Receives

The local node sends a 400-byte message over the session to the remote node. The remote node has 4 receives of 100 bytes each. The local DHB size is such that the message must be segmented into 200-byte segments. The local node sends the first 200-byte segment with a bit requesting a RECEIVE_CONTINUE from the remote node indicating it has a RECEIVE pending. The remote node can only accept part of the first segment and responds with a RECEIVE_OUTSTANDING, acknowledging 100 bytes and indicating more RECEIVEs are pending. This continues until the entire message has been acknowledged. See an example of the command sequence in Figure 5-17.

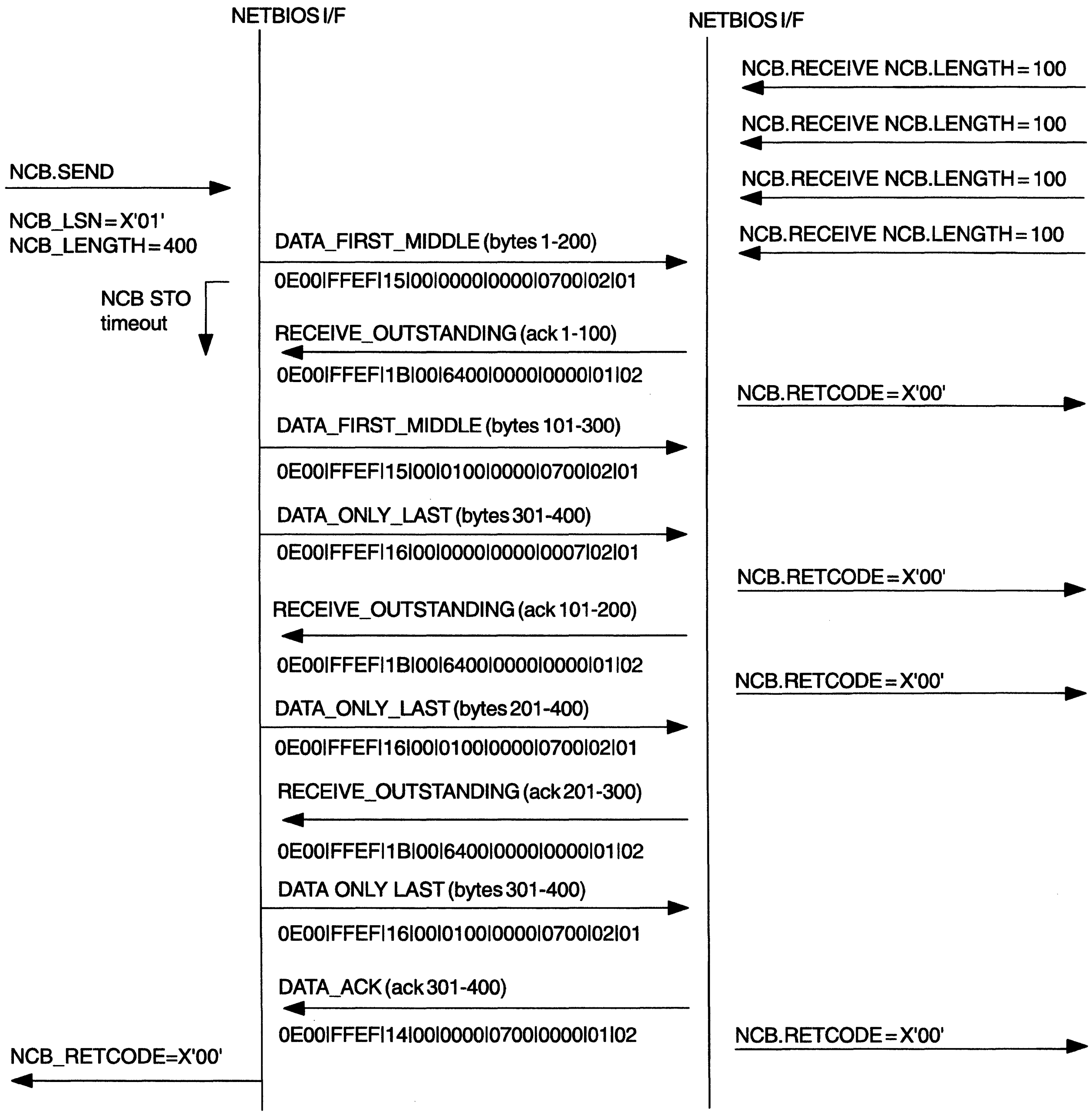

NETBIOS I/F
NETBIOS I/F
NCB.RECEIVE NCB.LENGTH=100
NCB.RECEIVE NCB.LENGTH=100
NCB.SEND
NCB.RECEIVE NCB.LENGTH=100
NCB_LSN=X'01'
NCB_LENGTH=400
DATA_FIRST_MIDDLE (bytes 1-200)
NCB.RECEIVE NCB.LENGTH=100
0E00|FFEF|15|00|0000|0000|0700|02|01
NCB STO
timeout
RECEIVE_OUTSTANDING (ack 1-100)
0E00|FFEF|1B|00|6400|0000|0000|01|02
NCB.RETCODE=X'00'
DATA_FIRST_MIDDLE (bytes 101-300)
0E00|FFEF|15|00|0100|0000|0700|02|01
DATA_ONLY_LAST (bytes 301-400)
0E00|FFEF|16|00|0000|0000|0007|02|01
NCB.RETCODE=X'00'
RECEIVE_OUTSTANDING (ack 101-200)
0E00|FFEF|1B|00|6400|0000|0000|01|02
NCB.RETCODE=X'00'
DATA_ONLY_LAST (bytes 201-400)
0E00|FFEF|16|00|0100|0000|0700|02|01
RECEIVE_OUTSTANDING (ack 201-300)
0E00|FFEF|1B|00|6400|0000|0000|01|02
DATA ONLY LAST (bytes 301-400)
0E00|FFEF|16|00|0100|0000|0700|02|01
DATA_ACK (ack 301-400)
0E00|FFEF|14|00|0000|0700|0000|01|02
NCB.RETCODE=X'00'
NCB_RETCODE=X'00'

Figure 5-17. Send Session Data: Segmentation and Multiple Receives Command Sequence

NETBIOS Protocol Examples with RND

Following are examples of NETBIOS 2.1 typical protocol scenarios that use the Remote Name Directory (RND) function to communicate with a remote station.

The local station locates a remote name by:

1. Issuing a NAME_QUERY on-ring once
2. Issuing a NAME_QUERY off-ring per transmit count
3. Transmitting timeout values that can be defined by NCB_TRANSMIT_COUNT and NCB_TRANSMIT_TIMEOUT parameters in the DIR.OPEN.ADAPTER command.

The range for NCB_TRANSMIT_COUNT is 1 to 10 and the default is 6. The range for NCB_TRANSMIT_TIMEOUT is 1/2 second to 10 seconds, and the default is 1/2 second.

When RND is used, after the local station has located a remote name, the remote node address is saved and subsequent messages to that name are to a specific node rather than a broadcast to all nodes.

The direction of a frame on the network is indicated by an arrow. The frame type is above the arrow and the contents of the frame are below the arrow. If a frame is repeated, the data appears only on the first instance of the frame.

Note: The numeric values shown in the various examples indicate the values as they appear on the line when the frame is being transmitted.

Remote Adapter Status for a Name Not on Network or RND

The application program attempts to request the remote status for a name that is not on the network. NAME_QUERY is sent to locate the name on the network on-ring once and off-ring at 1/2 second intervals for 6 times (by default). If a NAME_RECOGNIZED is not received, the application program NCB gets a return code of X'05' (command timed out).

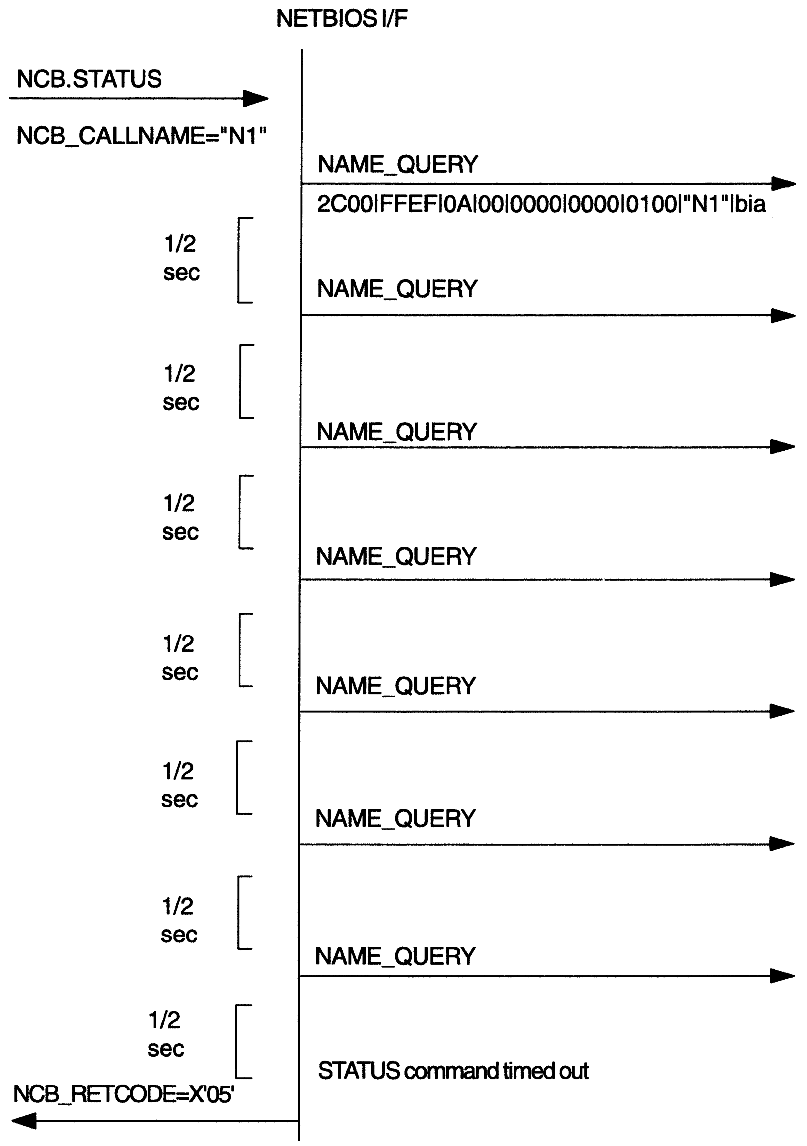


Figure 5-18. Remote Adapter Status for a Name Not on Network or RND Command Sequence

Remote Adapter Status for a Name Not in RND but on the Network

The application program attempts to request the remote status for a name that is on the network. The local RND is checked to see if the remote name exists in the table. If the remote name does not exist in the RND, the NAME_QUERY is sent to locate the name on the network on-ring once and off-ring at 1/2 second intervals for 6 times (by default). If a NAME_RECOGNIZED is received, the local NETBIOS station adds the remote name and node address to the RND and sends a STATUS_QUERY to that node. If a STATUS_RESPONSE is received the application program NCB gets a return code of X'00'. It is assumed that the application program buffer is big enough to hold the status data.

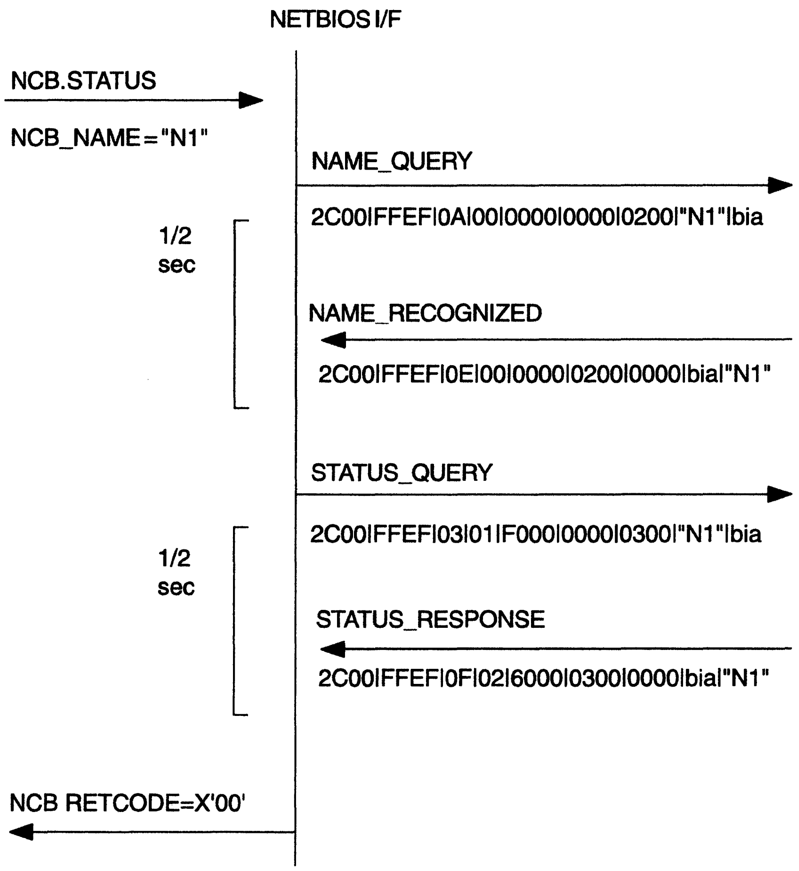

NETBIOS I/F
NCB.STATUS
NCB_NAME="N1"
NAME_QUERY
2C00|FFEF|0A|00|0000|0000|0200|"N1"|bia
1/2 sec
NAME_RECOGNIZED
2C00|FFEF|0E|00|0000|0200|0000|bia|"N1"
STATUS_QUERY
2C00|FFEF|03|01|F000|0000|0300|"N1"|bia
1/2 sec
STATUS_RESPONSE
2C00|FFEF|0F|02|6000|0300|0000|bia|"N1"
NCB RETCODE=X'00'

Figure 5-19. Remote Adapter Status for a Name Not in RND but on the Network Command Sequence

Remote Adapter Status for a Remote Name That Is in RND

The application program attempts to request the remote status for a name that is on the network. The local RND is checked to see if the remote name exists in the table. If the remote name exists in the RND, STATUS_QUERY is sent to the specific node address that is saved in the RND. If a STATUS_RESPONSE is received, the application program NCB gets a return code of X'00'. It is assumed that the application program buffer is big enough to hold the status data.

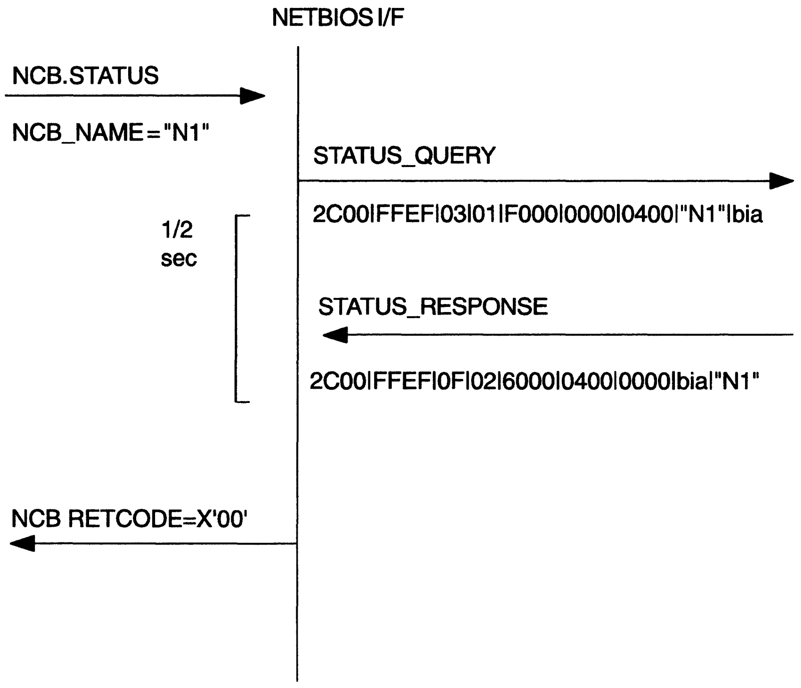


Figure 5-20. Remote Adapter Status for a Remote Name That Is in RND Command Sequence

Session Establishment RND Examples

The following scenarios show session establishment.

Call a Name: Name Not on Network or RND

The application program attempts to locate a name that does not exist on the network. The NAME_QUERY is sent on-ring once and at 1/2 second intervals for 6 times (by default). The NAME_QUERY frame is sent to a NETBIOS functional address (single-route broadcast). If a NAME_RECOGNIZED is not received, the application program NCB gets a return code of X'14' (cannot find name or no answer).

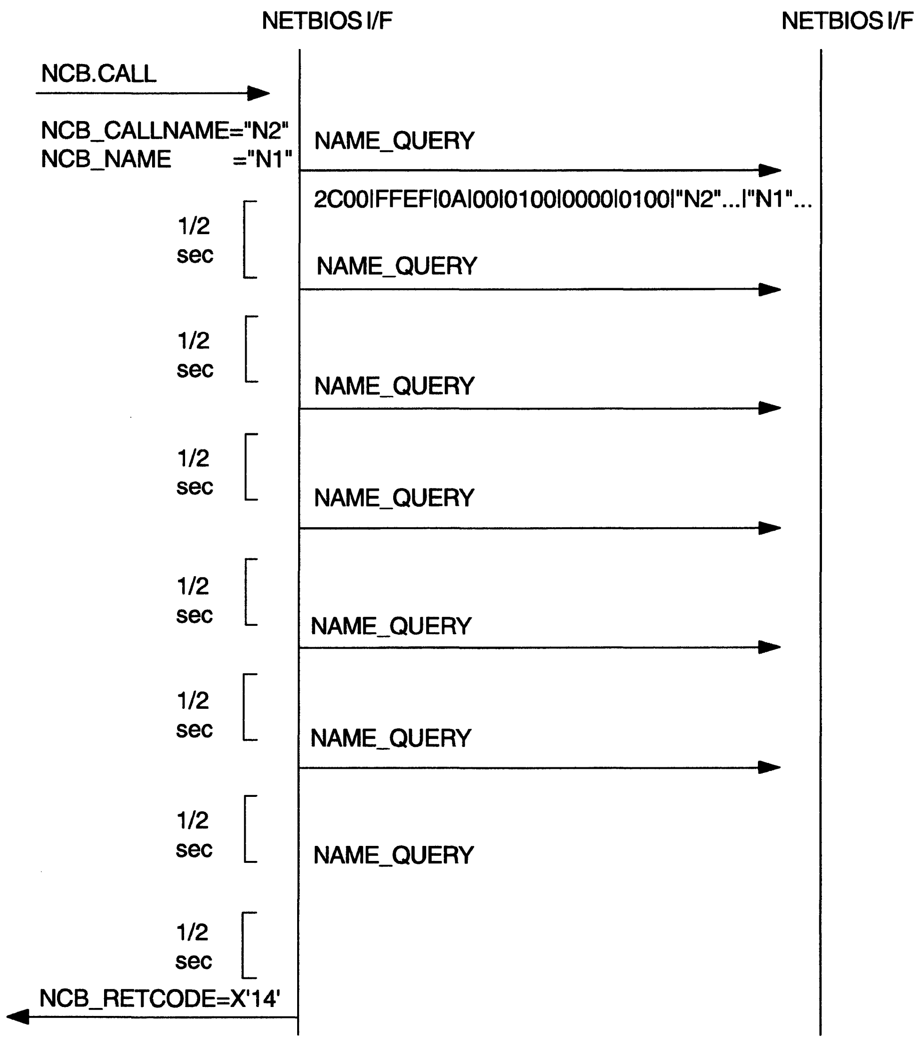


Figure 5-21. Call a Name: Name Not on Network or RND Command Sequence

Call a Name: Name Not in RND but Is on Network—Start Session

The application program uses name N1. and issues a call to name N2 that is registered at a remote node that has a LISTEN pending for that name. NAME_QUERY and NAME_RECOGNIZED contain the respective session numbers. Receipt of the NAME_RECOGNIZED causes the originating node to add the remote name N2 and the remote node address to the RND. This receipt also causes the originating node to send a SESSION_INITIALIZE (as an I frame) and run a timer waiting for a SESSION_CONFIRM (sent as an I-frame). The SESSION_INITIALIZE and SESSION_CONFIRM each indicate the maximum amount of user data that they are prepared to receive on this session. The remote partner then limits the size of the user data in frames transmitted over this session to this size or the size available in its transmit buffer (DHB), whichever is smaller. Receipt of the SESSION_CONFIRM completes the call and sets the NCB_RETCODE to X'00' and NCB_LSN to the locally assigned session number.

Note: If the remote name is already in RND, the NAME_QUERY is sent to a specific node address rather than to a NETBIOS functional address.

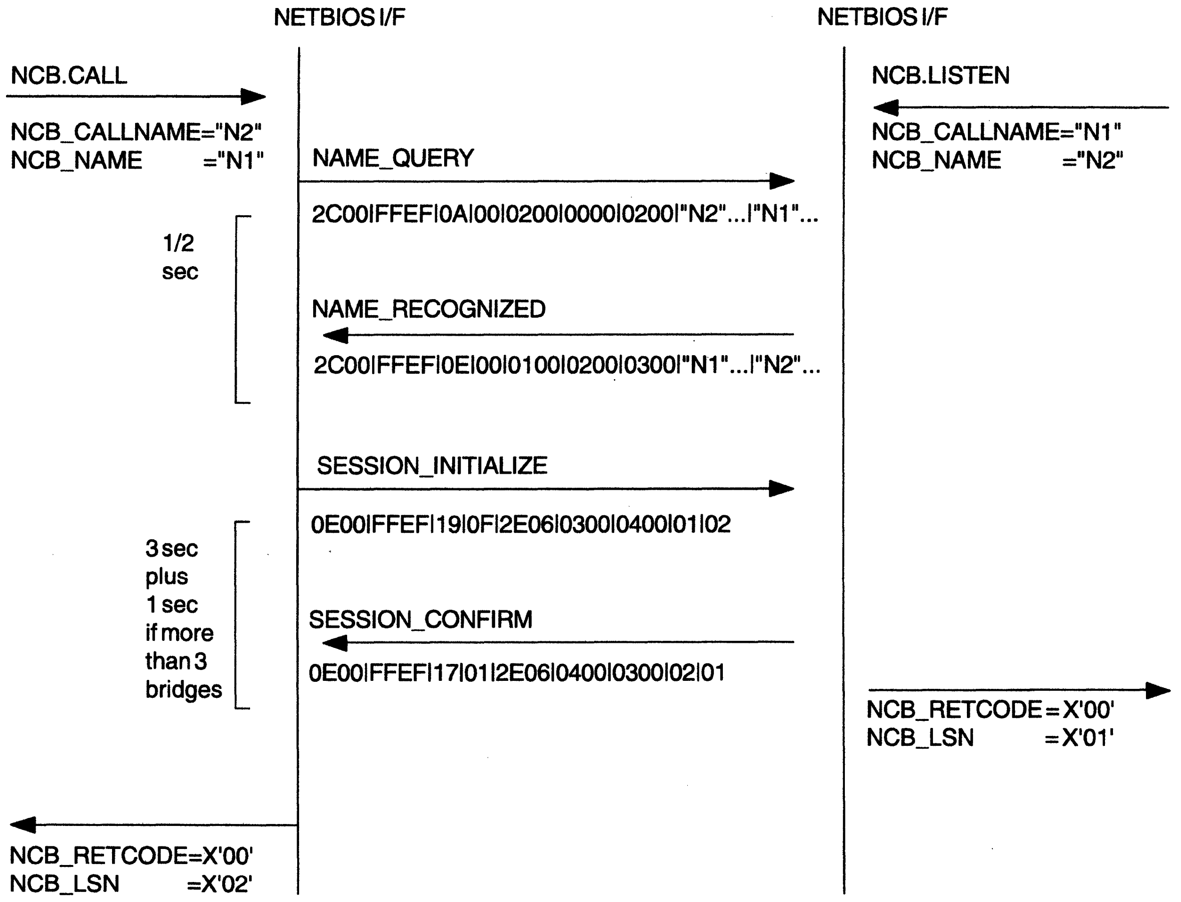


Figure 5-22. Call a Name: Name Not in RND but Is on Network—Start Session Command Sequence

NETBIOS Protocol SEND.NO.ACK Examples

The following NETBIOS 2.2 examples of typical protocol scenarios use the SEND.NO.ACK/CHAIN.SEND.NO.ACK function to communicate with a remote station.

The direction of a frame on the network is indicated by an arrow. The frame type is above the arrow and the contents of the frame are below the arrow. If a frame is repeated, the data appears only on the first instance of the frame.

Note: The numeric values shown in the various examples indicate the values as they appear on the line when the frame is being transmitted.

Session Establishment Examples

The following scenarios show session establishment.

Call a Name: Name Found—Start Session

The application program uses name "N1" and issues a call to name "N2" that is registered at a remote node that has a LISTEN pending for that name. NAME_QUERY and NAME_RECOGNIZED contain the respective session numbers. Receipt of the NAME_RECOGNIZED causes the originating node to send a SESSION_INITIALIZE (as an I frame) and run a timer waiting for a SESSION_CONFIRM (sent as an I frame). The SESSION_INITIALIZE and SESSION_CONFIRM each indicate the maximum amount of user data that they are prepared to receive on this session. The remote partner then limits the size of the user data in frames transmitted over this session to this size or the size available in its transmit buffer (DHB), whichever is smaller.

A NO_ACK indicator flag is set in the DATA1 field of SESSION_INITIALIZE and SESSION_CONFIRM frames to indicate that the sending node has the ability to handle SEND.NO.ACK or CHAIN.SEND.NO.ACK commands. Receipt of the SESSION_CONFIRM completes the call and sets the NCB_RETCODE to X'00' and NCB_LSN to the locally assigned session number.

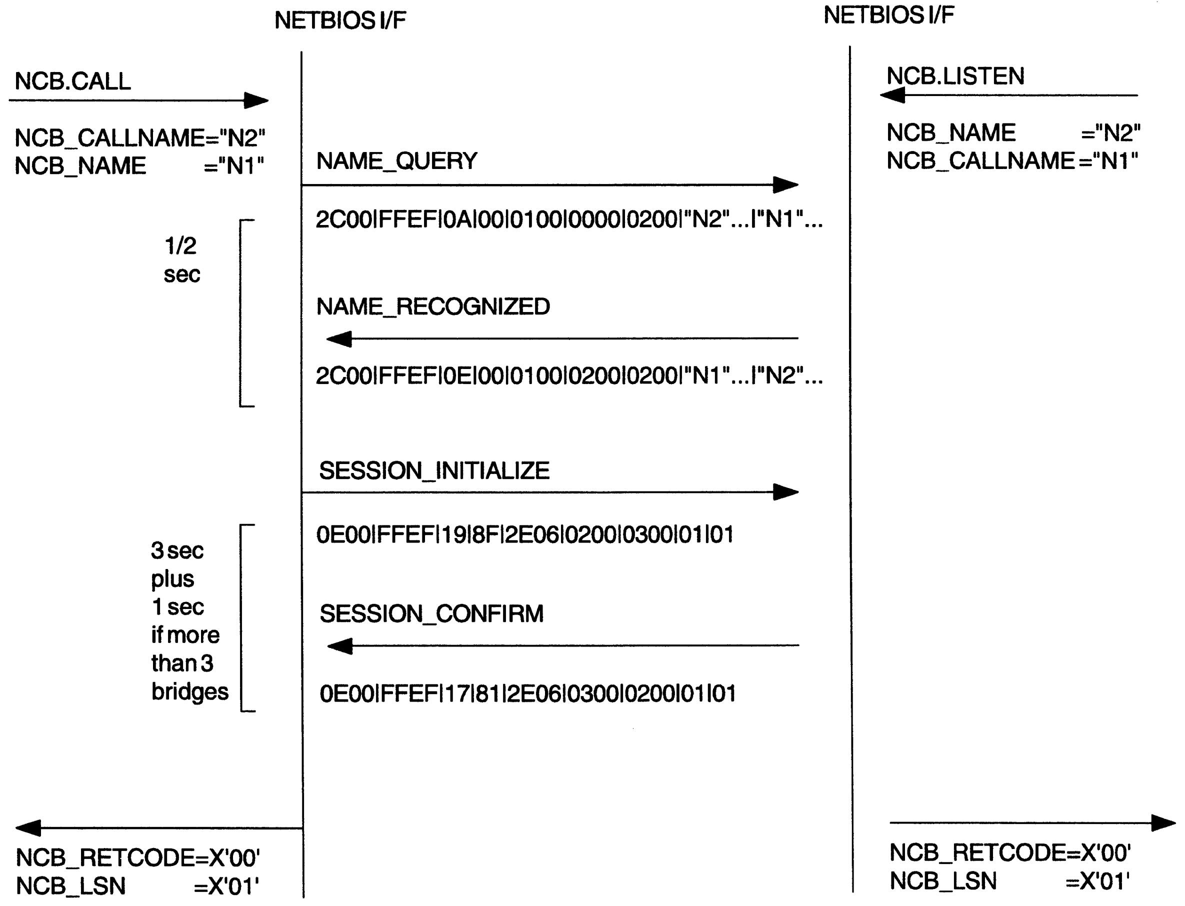


Figure 5-23. Call a Name: Name Found—Start Session Command Sequence

Session Data Transfer Examples

The following scenarios show data transfer across a session.

Send Session Data: One SEND.NO.ACK and No RECEIVE

The session has been established as above. The local side issues a RECEIVE_ANY no-wait NCB for 10 bytes and a SEND.NO.ACK command for 100 bytes and the remote node has no RECEIVE pending. The local side sends a DATA_ONLY_LAST frame. Soon after the frame is transmitted, the SEND.NO.ACK command is executed with a X'00' return code.

The remote side recognizes the DATA_ONLY_LAST frame and that the data is of SEND.NO.ACK or CHAIN.SEND.NO.ACK type. Since the remote side does not have any RECEIVE pending for the session, a NO_RECEIVE frame is sent in response to indicate that the SEND.NO.ACK or CHAIN.SEND.NO.ACK data is either not received or only partially received. When this frame is received, the local station terminates a pending session type command (in this example, RECEIVE_ANY) with a X'07' return code.

Note: If there is no pending RECEIVE command for the session at the receipt of a NO_RECEIVE frame, the next RECEIVE command (or other applicable command) issued will get a X'07' return code.

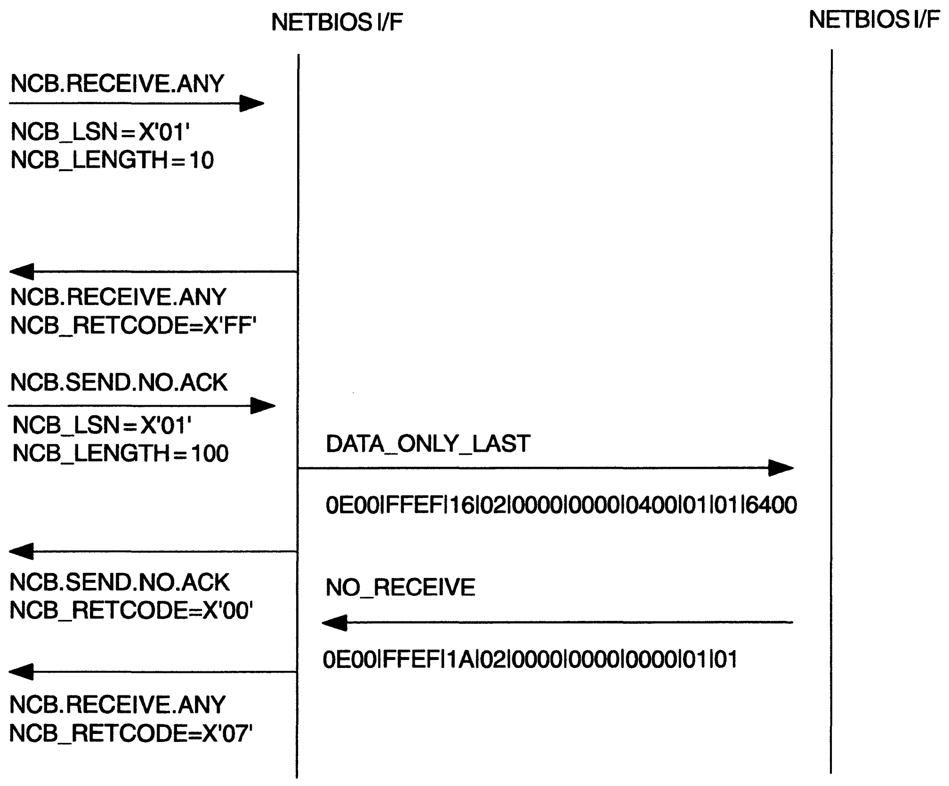


Figure 5-24. Send Session Data: One SEND.NO.ACK and No RECEIVE Command Sequence

Send Session Data: One SEND.NO.ACK and One RECEIVE

The session has been established as above. The local node issues a SEND.NO.ACK command for 100 bytes and the remote node has a corresponding 100-byte RECEIVE pending. The local node sends a DATA_ONLY_LAST and the SEND.NO.ACK is completed with a X'00' return code.

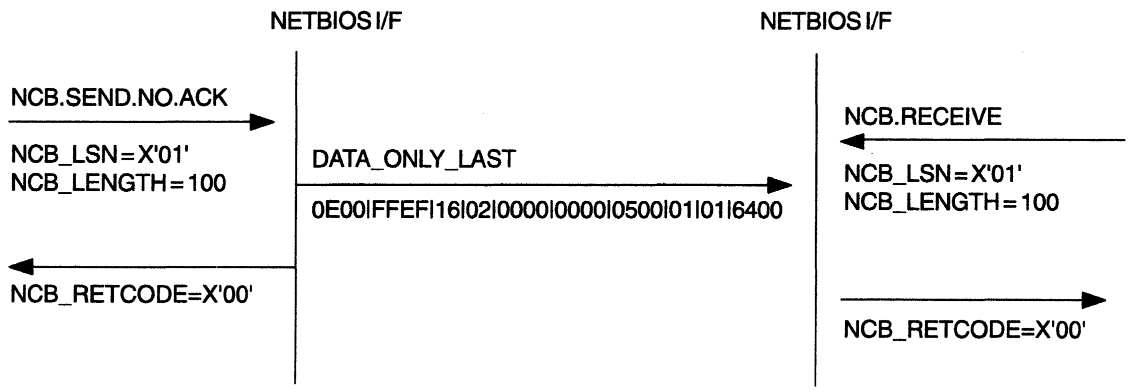


Figure 5-25. Send Session Data: One SEND.NO.ACK and One RECEIVE Command Sequence

Send Session Data: One SEND.NO.ACK and Multiple RECEIVEs

The local node sends a 200-byte message over the session to the remote node via the SEND.NO.ACK command. The remote node has two 100-byte RECEIVEs outstanding. The remote node accepts a 100-bytes RECEIVE and transmits a NO_RECEIVE frame indicating that only part of the SEND.NO.ACK or CHAIN.SEND.NO.ACK data was received by the remote application program.

If there is no pending RECEIVE command for the session at the receipt of a NO_RECEIVE frame, the next RECEIVE (or other applicable command) command issued will get a X'07' return code.

Note: For data transmitted via SEND.NO.ACK or CHAIN.SEND.NO.ACK, only one RECEIVE command will be satisfied. The SEND.NO.ACK or CHAIN.SEND.NO.ACK data buffer must be no longer than the remote receive buffer for successful completion.

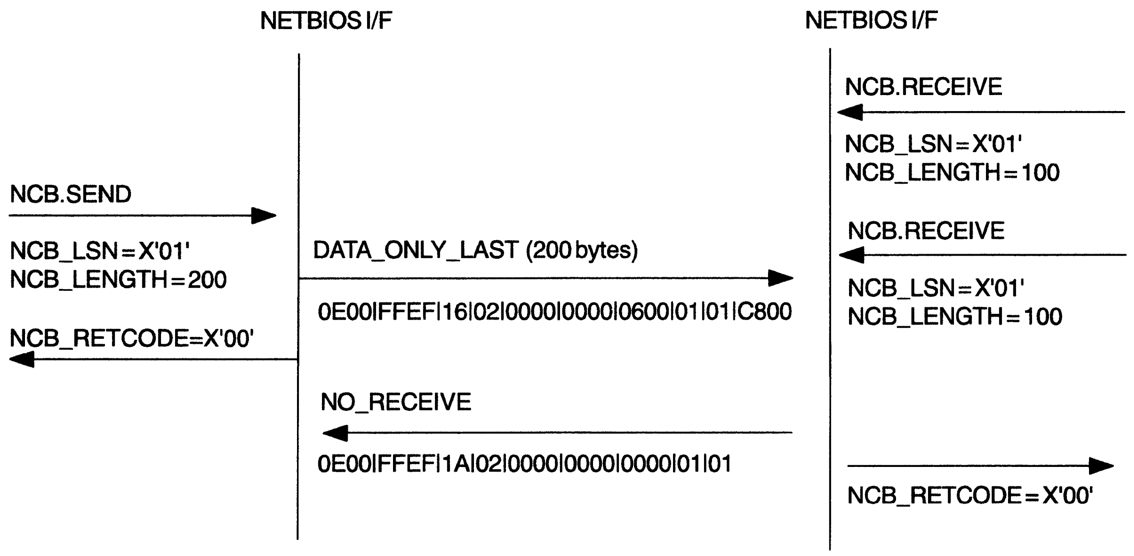


Figure 5-26. Send Session Data: One SEND.NO.ACK and Multiple RECEIVEs Command Sequence

Chapter 6. Support of NDIS Adapters Using IBM OS/2 LAN Adapter and Protocol Support

LAN Adapter and Protocol Support Overview 6-2
LAN Adapters Supported 6-3
Programming Information—IEEE 802.2 API 6-3
  Dynamic Link Interface 6-3
  Device Driver Interface 6-3
  Memory Restriction 6-3
Programming Information—NETBIOS API 6-4
  No-Wait Command and Post Routines 6-4
  Dynamic Link Interface 6-4
  Device Driver Interface 6-5
  Memory Restriction 6-5
  Resource Information 6-5
  NCB Reserve Field Change 6-5
  NCB.STATUS Command Extension 6-5
  Piggybacked Acknowledgment Behavior 6-6
Differences and Restrictions 6-6
Message Logging Facility 6-9
Operating System/2 Trace Facility 6-10
  New Tracing Parameters for the NETBIOS and IEEE 802.2 APIs 6-11
    TRACEOFF 6-11
    TRACENAMES 6-12
  Minor Codes for IEEE 802.2 Traces 6-13
  PROTOCOL.INI TRACE Parameter—IEEE 802.2 6-15
  Minor Codes for NETBIOS Traces 6-17
  PROTOCOL.INI OS2TRACEMASK Parameter—NETBIOS 6-19
  Tracing for OS/2 2.0 Virtual DOS LAN Support 6-20
    Minor Codes for OS/2 2.0 Virtual DOS LAN Support 6-21

LAN Adapter and Protocol Support Overview

LAN Adapter and Protocol Support, network communication software, is a combination of protocol drivers and network adapter drivers[1] that comply to the Network Driver Interface Specification (NDIS),[2] API support software, and configuration and installation software for these drivers. LAN Adapter and Protocol Support is included in Extended Services for OS/2, IBM OS/2 LAN Server Versions 2.0 and 3.0, and NTS/2. It consists of the network communication software necessary to support the LAN connectivity needed by both IBM Extended Services for OS/2 and IBM OS/2 LAN Server.

IEEE 802.2 and NETBIOS are the protocol drivers available with LAN Adapter and Protocol Support. A device driver interface API and a dynamic link interface API exist for both protocols. The IEEE 802.2 and NETBIOS APIs and the IEEE 802.2 and NETBIOS protocols are described in detail in Chapters 1 through 5 of this technical reference.

NDIS allows multiple protocols to bind to the same network adapter driver. Therefore, both the NETBIOS and 802.2 protocol drivers can bind to the same network adapter driver. Figure 6-1 shows the relationship between LAN adapters, network adapter drivers, protocol drivers, and applications.

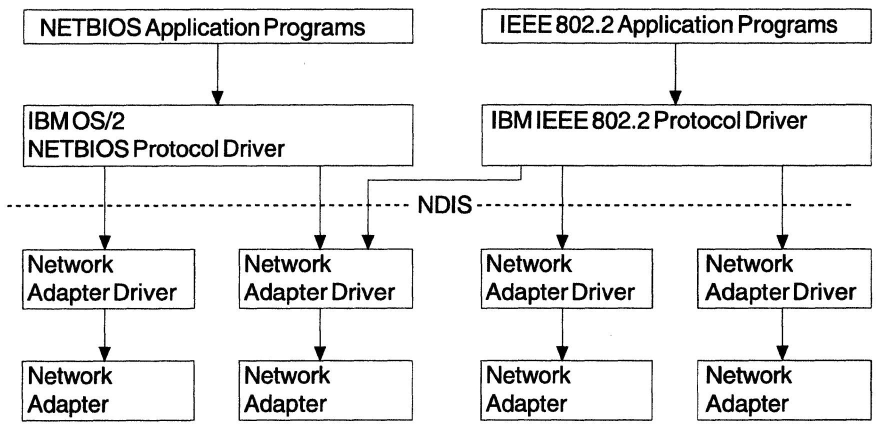


Figure 6-1. LAN Adapter and Protocol Support Diagram

In addition to the OS/2 API support, LAN Adapter and Protocol Support provides OS/2 2.0 Virtual LAN Support. The LAPS provided with ES 1.0 and LAN Server 2.0 offers virtual NETBIOS support. The LAPS provided with LAN Server 3.0 and NTS/2 offers virtual support for both NETBIOS and IEEE 802.2 applications. This virtual LAN support enables existing DOS NETBIOS applications and DOS 802.2 applications running in the OS/2 2.0 Virtual DOS Machine environment to share an adapter with other DOS and OS/2 NETBIOS applications running on the same machine.

Configuration parameters for LAN Adapter and Protocol Support network adapter drivers and protocol drivers exist in the file PROTOCOL.INI. PROTOCOL.INI also contains the binding information for each driver. Refer to the *Extended Services*

1 A *network adapter driver* is also known as an *NDIS MAC driver.*

2 See Appendix F, "NDIS Overview" for a brief description of NDIS.

Communications Manager Configuration Guide, the *Extended Services LAN Adapter and Protocol Support Configuration Guide*, the *OS/2 LAN Server Network Administrator Reference Volume 2: Performance Tuning*, or the *NTS/2 LAN Adapter and Protocol Support Configuration Guide* for a detailed explanation of configuration parameters and PROTOCOL.INI.

LAN Adapters Supported

LAPS supports most IBM adapters and certain NDIS adapters produced by manufacturers other than IBM. IBM has put in place enablement support for development and testing of network interface drivers by network interface adapter vendors. A list of the IBM adapters supported and the certified NDIS adapters produced by other manufacturers is available on CompuServe** and is maintained in the ES/LS Questions section of the OS/2 Support Forum.

Programming Information—IEEE 802.2 API

The IEEE 802.2 protocol stack included in LAN Adapter and Protocol Support is NDIS-based. This section documents the changes to the API as a result of the new release. This section also contains answers to user questions about the OS/2 IEEE 802.2 API.

Dynamic Link Interface

The ACS.LIB library shipped with IBM OS/2 Extended Services contains the dynamic link definition for the IEEE 802.2 DLL user. ACS.LIB is not included in LAN Server Version 2.0 or 3.0. However, the user can create a .DEF file to create the same definition without linking to ACS.LIB. The .DEF file should contain the following statements:

```
IMPORTS

    ACSLAN.ACSLAN
```

For the dynamic link interface IEEE 802.2 locks the data buffers associated with the CCB. The overhead associated with operating system locks and unlocks may be minimized if the application uses the same segment or segments for all of its data buffers.

Device Driver Interface

In an application program using the IEEE 802.2 OS/2 device driver interface, the register pair ES : BX contains a pointer to the CCB issued. This pointer must be a GDT when using the device driver interface.

Memory Restriction

OS/2 2.0 introduces a paged memory model. However, the IEEE 802.2 protocol stack is a 16-bit device driver. Any memory passed to the IEEE 802.2 protocol stack in the form of CCBs or buffers must be contiguous in physical memory. The IBM C Set/2 compiler provides the /Gt compiler option, which defines 16-bit versions of the `malloc` family of functions. Use of this option guarantees that memory allocated and freed will not cross a 64 KB boundary. Refer to the *IBM C Set/2 User's Guide* for more information on compiler options. OS/2 2.0 ensures that the memory of a 16-bit application program is physically contiguous.

Programming Information—NETBIOS API

The release of NETBIOS included in LAN Adapter and Protocol Support is NETBIOS 4.0. NETBIOS 4.0 is NDIS-based. This section documents the changes to the API as a result of the new release.

No-Wait Command and Post Routines

When using a no-wait command at either the dynamic link or device driver interface, the user commonly has a post routine that gets control upon command completion. When the post routine is called, NETBIOS is running on interrupt. Therefore, the post routine should be as short in duration as possible. It is valid to issue NCB commands inside the post routine as long as the command is a no-wait command.

For the dynamic link interface, the NETBIOS stack used by the thread that calls the user's post routine is 4KB. Before calling the post routine, NETBIOS pushes the NCB pointer ES : BX on the stack. Upon entry to the user's post routine, ES and BX are at the following location in the stack frame.

ES	stack pointer + 8
BX	stack pointer + 6
Undefined	stack pointer + 4
Return address	stack pointer + 2
Return address	stack pointer + 0

Use a C calling convention, not Pascal, to access the NCB pointer on the stack.

Note: The priority of the NETBIOS dynamic link interface thread that calls the user's post routine is set to priority class fixed high and priority delta zero. Some dynamic link interface applications have experienced performance degradation when setting threads to a priority higher than the NETBIOS post routine thread.

Dynamic Link Interface

The ACS.LIB library shipped with IBM OS/2 Extended Services contains the dynamic link definition for the NETBIOS DLL user. ACS.LIB is not included in IBM OS/2 LAN Server Version 2.0. However, the user can create a .DEF file to create the same definition without linking to ACS.LIB. The .DEF file should contain the following statements:

```
IMPORTS

    ACSNETB.NETBIOS
```

For the dynamic link interface NETBIOS locks the data buffers (NCB_BUFFER) associated with the NCB. The overhead associated with operating system locks and unlocks may be minimized if the application uses the same segment or segments for all of its data buffers.

Device Driver Interface

In an application program using the NETBIOS OS/2 device driver interface, the register pair ES : BX contains a pointer to the NCB issued. This pointer must be a GDT when using the device driver interface.

Memory Restriction

OS/2 2.0 introduces a paged memory model. However, NETBIOS 4.0 is a 16-bit device driver. Any memory passed to NETBIOS in the form of NCBs or buffers must be contiguous in physical memory. The IBM C Set/2 compiler provides the /Gt compiler option, which defines 16-bit versions of the `malloc` family of functions. Use of this option guarantees that memory allocated and freed will not cross a 64KB boundary. Refer to the *IBM C Set/2 User's Guide* for more information on compiler options. OS/2 2.0 ensures that the memory of a 16-bit application program is physically contiguous.

Resource Information

NETBIOS uses the message logging facility of LAN Adapter and Protocol Support at IPL time. Following is the NETBIOS information extracted from a log:

```
IBM OS/2 NETBIOS 4.0

Adapter 0 has 32 NCBs, 32 sessions, and 32 names available to NETBIOS
applications.

NETBIOS 4.0 is loaded and operational.
```

NETBIOS will log the sessions, commands, and names available to the NETBIOS API user for each adapter configured. In this example, only adapter 0 is configured. NETBIOS 4.0 will support a maximum of four adapters. A maximum of 251 application programs (processes) can use NETBIOS 4.0 simultaneously.

NCB Reserve Field Change

The only returned field defined for the NCB_RESERVE area of the NCB is offset 6. It will contain the last Adapter Check reason code. See Appendix B, "Return Codes" for information on Adapter Check reason codes.

NCB.STATUS Command Extension

The NCB.STATUS command has been extended to return the locally administered address on local status. Offset 36 of the status buffer is now defined for OS/2 for local status only. If the NCB is passed to the dynamic link interface, offset 36 is an LDT; if the NCB is passed to the device driver interface, offset 36 is a GDT. The LDT or GDT is the address of the extended status information. The locally administered address is returned in the extended status information at offset 40 through 45. If a locally administered address is not configured, the encoded address of the adapter is returned. Other areas defined in the extended status information are:

- Bytes 2 and 3 contain the latest NDIS Open Adapter return code
- Bytes 4 and 5 contain the latest NDIS Ring Status status code
- Bytes 6 and 7 contain the latest NDIS Adapter Check reason code.

Piggybacked Acknowledgment Behavior

NETBIOS 4.0 will not attempt to piggyback acknowledgments if the data is received on an NCB.RECEIVE; a DATA_ACK will be sent upon receipt of the data. However, if an NCB.RECEIVE.ANY is used to receive data, the protocol will attempt to piggyback the acknowledgment. Piggybacked acknowledgment is only applicable for the NETBIOS device driver interface. Piggybacked acknowledgment is configurable. However, the configuration parameter sets piggybacking for the adapter, not the application. Therefore, use caution when changing this parameter. It will affect the performance of other NETBIOS applications. Refer to the *Extended Services Communications Manager Configuration Guide*, the *Extended Services LAN Adapter and Protocol Support Configuration Guide*, the *OS/2 LAN Server Network Administrator Reference Volume 2: Performance Tuning*, or the *NTS/2 LAN Adapter and Protocol Support Configuration Guide* for more information about the NETBIOS configuration parameters.

Differences and Restrictions

The following is a list of differences between the version of LAN Adapter and Protocol Support that is included in IBM Extended Services for OS/2 and IBM OS/2 LAN Server Versions 2.0 and 3.0, and the network communication software that is included in IBM OS/2 LAN Extended Edition 1.3. Many of the changes are a result of moving to NDIS-compliant network adapter drivers and protocol drivers. Some, such as support of four network adapters, are enhancements. The restrictions resulting from these changes are related to NDIS requirements and OS/2 2.0.

- A protocol driver using the NDIS interface now implements all the LLC function. The LLC function is handled in the protocol driver, not on the adapter. For the Token-Ring Network adapters, the LLC function is now handled in the protocol driver.
- The LAN protocol drivers will now bind to a maximum of four network adapter drivers. Therefore, adapter numbers 0 through 3 are valid parameters for the NETBIOS API and the IEEE 802.2 CCB2 and CCB3 APIs.
- The NDIS interface now restricts the number of chained transmit buffers to 8. Therefore, if an attempt is made to transmit a frame that is contained in more than 8 buffers, the CCB2 and CCB3 TRANSMIT command will be returned with a return code of X'23' This return code will also be reported if the internal supply of NDIS frame descriptors is exhausted.
- The NDIS interface now restricts the number of chained receive buffers to 8. Therefore, if an attempt is made to receive a frame that is larger than 8 SAP buffers, the CCB2 and CCB3 RECEIVE command will be returned with a return code of X'18'.
- IBM Token-Ring and PC Network network adapter drivers now implement NDIS 2.02 to allow the protocol stack to access adapter-specific information. For network adapter drivers that do not support NDIS 2.02, the following CCB2 and CCB3 return fields are undefined:
  - DIR.INITIALIZE
    - BRING_UPS
    - SRAM_ADDRESS
    - VIRTUAL_SRAM_ADDRESS
    - VIRTUAL_MMIO_ADDRESS

- DIR.OPEN.ADAPTER
  - OPEN_ERROR_CODE
  - BRING_UPS
- DIR.STATUS
  - MICROCODE_LEVEL
  - ADAPTER_PARMS_ADDR
  - ADAPTER_MAC_ADDR

- DIR.OPEN.ADAPTER command for CCB2 and CCB3. The value returned in the parameter DATA_HOLD_BUFFERS, offset 24 of the Adapter Parms Open Parameters, is the value present in the characteristic table for the network adapter driver to which the IEEE 802.2 protocol is bound. The value returned in the parameter DHB_BUFFER_LENGTH, offset 22 of the Adapter Parms Open Parameters, minus 6 is the length of the maximum guaranteed I field.
- The term data hold buffer (DHB) is only applicable when the network adapter driver is Token-Ring.
- DIR.READ.LOG adapter log change. The adapter log data returned by DIR.READ.LOG is the data returned by a Read Error Log request to the network adapter driver. For IBM Token-Ring and PC Network adapters, the format is unchanged from the previous version of the IEEE 802.2 interface. If the network adapter driver does not support the Read Error Log function, the adapter log contents will be undefined.
- DIR.SET.FUNCTIONAL.ADDRESS command for CCB2 and CCB3. The following applies only to network adapter drivers that do not support functional addresses, as indicated by the Service Flags parameter in the MAC Service Specific Characteristics table. For each bit to be set in the functional address, an NDIS Add Multicast Address command is issued to add a multicast address with that bit on. For each bit to be reset in the functional address, an NDIS Delete Multicast Address command is issued to delete the corresponding multicast address. If the number of bits to be in the functional address exceeds the number of available multicast addresses, the DIR.SET.FUNCTIONAL.ADDRESS command is not executed and a return code of X'1E' is posted to the CCB.
- The algorithms for calculating RAM, shared RAM, and work area are new. Refer to the *Extended Services Communications Manager Configuration Guide*, the *Extended Services LAN Adapter and Protocol Support Configuration Guide*, the *OS/2 LAN Server Network Administrator Reference Volume 2: Performance Tuning* , or the *NTS/2 LAN Adapter and Protocol Support Configuration Guide* for the calculation tables.
- NETBIOS provided in LAN Adapter and Protocol Support is NETBIOS Version 4.0.
- A maximum of 251 application programs (processes) can use NETBIOS 4.0 simultaneously.
- The only returned field defined for the NCB_RESERVE area of the NCB is offset 6. This field will contain the last Adapter Check reason code.
- NCB.STATUS command. This command has been extended to return the locally administered address on local status. Offset 36 of the status buffer is now defined for local status only. If the NCB is passed to the DLL interface, offset 36 is an LDT; if the NCB is passed to the DD interface, offset 36 is a

GDT. The LDT or GDT is the address of the extended status information. The locally administered address is returned in the extended status information at offset 40 through 45. If a locally administered address is not configured, the encoded address of the adapter is returned. other areas defined in the extended status information are:

  - Bytes 2 to 3 contain the latest NDIS Open Adapter return code
  - Bytes 4 to 5 contain the latest NDIS Ring Status status code
  - Bytes 6 to 7 contain the latest NDIS Adapter Check reason code.

- NETBIOS 4.0 will not attempt to piggyback acknowledgments if the data is received on an NCB.RECEIVE; a DATA_ACK will be sent upon receipt of the data. However, if an NCB.RECEIVE.ANY is used to receive data, the protocol will attempt to piggyback the acknowledgment. Piggybacked acknowledgment is only applicable for the NETBIOS device driver interface. Piggybacked acknowledgment is configurable. However, the configuration parameter sets piggybacking for the adapter, not the application. Therefore, use caution when changing this parameter. It will affect the performance of other NETBIOS applications. Refer to the *Extended Services Communications Manager Configuration Guide*, the *Extended Services LAN Adapter and Protocol Support Configuration Guide*, the *OS/2 LAN Server Network Administrator Reference Volume 2: Performance Tuning* or the *NTS/2 LAN Adapter and Protocol Support Configuration Guide* for more information on NETBIOS configuration parameters.
- The Adapter Status parameter table returned by the CCB2 and CCB3 DIR.STATUS command may contain all zeroes if the network adapter driver to which IEEE 802.2 is bound is not NDIS Version 2.02.
- OS/2 2.0 Virtual DOS LAN Support operates only on OS/2 2.0 or greater.
- OS/2 2.0 Virtual DOS LAN Support will share the adapter with other OS/2 and DOS NETBIOS or 802.2 applications. Therefore, a DOS-based NETBIOS application can experience problems if it assumes that it is the only user of the adapter. For example, NCB.STATUS will return the names in the name table of the adapter, which includes names in use for all NETBIOS applications using the adapter. The LTSVCFG configuration utility is provided to handle some inconsistencies between the DOS and OS/2 NETBIOS environment. Refer to the *OS/2 LAN Server Network Administrator Reference Supplement for OS/2 2.0* or the *NTS/2 LAN Adapter and Protocol Support Configuration Guide* for more information about OS/2 2.0 Virtual DOS LAN Support configuration.
- The Token-Ring Network 16/4 Busmaster Server Adapter/A is a DMA adapter. Due to a hardware limitation, the adapter does not access memory above a boundary of 16 MB in physical memory. Therefore, any 32-bit device driver written to the IEEE 802.2 device driver interface API or the NETBIOS device driver interface API that wants to successfully run on the Token-Ring Network 16/4 Busmaster Server Adapter/A network adapter must ensure that all memory passed to the API exists below the 16MB boundary in physical memory. The 16-bit device driver interface applications, 16-bit dynamic link interface applications, and 32-bit dynamic link interface applications are not affected.
- Memory passed to all LAN Adapter and Protocol Support interfaces must be contiguous in physical memory. OS/2 2.0 introduces a paged memory model. However, the IEEE 802.2 protocol stack and the NETBIOS 4.0 protocol stack are both 16-bit device drivers. Any memory passed to either protocol stack in the form of control blocks or buffers must be contiguous in physical memory.

OS/2 2.0 ensures that the memory of a 16-bit application program is physically contiguous. See also "Programming Information—NETBIOS API" on page 6-4, "Programming Information—IEEE 802.2 API" on page 6-3, and the *NTS/2 LAN Adapter and Protocol Support Configuration Guide.*

- The *3COM*/Microsoft* LAN Manager Network Driver Interface Specification Version 2.01 Final* defines syntax guidelines for the PROTOCOL.INI files. The specification states that all PROTOCOL.INI keywords that contain multiple values on the right-hand side of the equals sign may be separated by commas, semicolons, or spaces. However, the LAN Adapter and Protocol Support Configuration utility included with IBM Extended Services for OS/2 and IBM OS/2 LAN Server Version 2.0 supports commas and semicolons as separators but does not support spaces as separators.

Message Logging Facility

LAN Adapter and Protocol Support includes a message logging facility. This facility serializes logging of messages to a common log file, or optionally, to the screen. Service is provided for both ring 0 and ring 3 users. The LAN Adapter and Protocol Support protocol and network adapter drivers log messages at IPL time to the common log file LANTRAN.LOG. Some of the network adapter drivers will also log error messages at run time. The contents of the file help with debugging information as well as with resource and configuration information.

Figure 6-2 is a screen display of LANTRAN.LOG. In this example, the NETBIOS protocol driver and the IEEE 802.2 protocol driver are bound to the Token-Ring network adapter driver. The Token-Ring adapter is configured as adapter 0 for both protocols.

```
IBM OS/2 LANMSGDD [12/03/91] 2.01 is loaded and operational.

IBM OS/2 LAN Protocol Manager

IBM OS/2 NETBEUI 2.01
IBM - IBM Token-Ring Network Driver, Version V.2.02
IBM OS/2 LANDD [11/12/91] 2.01.2
IBM OS/2 LANDLLDD 2.01
IBM OS/2 LANDLLDD is loaded and operational.
IBM OS/2 NETBIOS 4.0
Adapter 0 has 32 NCBs, 32 sessions, and 32 names available to NETBIOS
applications.
NETBIOS 4.0 is loaded and operational.
IBM OS/2 LAN Netbind
IBM Token-Ring adapter data rate is 4 mbps.
IBM LANDD is accessing IBM 802.5 LAN Interface.
Adapter 0 was initialized and opened successfully.
Adapter 0 is using node address 10005A8D4426.
IBM LANDD was successfully bound to MAC: IBMTOK->VECTOR.
```

Figure 6-2. Sample LANTRAN.LOG

To learn more about the message logging facility, refer to the *IBM NDIS Implementation Information for OS/2 Messaging & National Language Support* contained in the *IBM OS/2 NDIS Driver Implementation Package.*

Operating System/2 Trace Facility

The Operating System/2 System Trace functions are redefined for the NDIS-based version of the IEEE 802.2 interface. The major code for IEEE 802.2 system traces will remain 164 (0xA4). Minor codes 64 (0x40) through 81 (0x51) will be used by the NDIS-based version of the IEEE 802.2 interface. The format of the trace data for each of the minor codes is explained in "Minor Codes for IEEE 802.2 Traces" on page 6-13.

The Operating System/2 System Trace functions are redefined for NETBIOS 4.0. The major trace code for NETBIOS 4.0 is still 164 (0xA4). Minor codes defined for NETBIOS 4.0 are 96 (0x60) through 106 (0x 6A) and 120 (0x78) through 123 (0x7B). The format of the trace data for each of the minor codes is explained in "Minor Codes for NETBIOS Traces" on page 6-17.

To enable the OS/2 Trace Facility, the user must specify a trace buffer in CONFIG.SYS. Add the following statement to CONFIG.SYS to specify the trace buffer:

```
TRACEBUF=63
```

The information traced is controlled by the OS/2 major code trace selection and the protocol trace bit mask. The major code for LAN Adapter and Protocol Support is 164 (0xA4). Both NETBIOS and IEEE 802.2 have defined trace bit masks to select minor code information traced.

The IEEE 802.2 tracing is controlled by the TRACE parameter in the LANDD section of the PROTOCOL.INI file. This parameter is a bitmap of flags for enabling or disabling subsets of the available trace points. The TRACE parameter consists of 10 bit flags. To enable a particular subset of traces, the bit controlling those traces is set on in the TRACE parameter. For example, to trace CCBs issued and their parameter tables and status events, set TRACE = 0x000E. The subset of traces controlled by each bit of the TRACE parameter is explained in "PROTOCOL.INI TRACE Parameter—IEEE 802.2" on page 6-15 and summarized in Table 6-1 on page 6-14.

The OS2TRACEMASK parameter in the NETBEUI section of PROTOCOL.INI controls the tracing of NETBIOS minor code events. The default OS2TRACEMASK is 0x0000, which is OFF. You can use the Configuration/Installation Program to change the value of the OS2TRACEMASK parameter, or you can use an editor with caution to edit the NETBEUI section of PROTOCOL.INI. Add OS2TRACEMASK to the NETBEUI section of PROTOCOL.INI, if it does not already exist. The format is OS2TRACEMASK = 0x*nnnn* where *nnnn* is the bitmap. Enabling and disabling tracing of minor codes is controlled by bit selection in OS2TRACEMASK. The bits associated with each minor code are defined in "PROTOCOL.INI OS2TRACEMASK Parameter—NETBIOS" on page 6-19 and summarized in Table 6-5 on page 6-20.

After making changes to CONFIG.SYS and PROTOCOL.INI for NETBIOS or IEEE 802.2, restart the computer. To begin tracing, type the following at the command prompt:

```
TRACE ON 164
```

New Tracing Parameters for the NETBIOS and IEEE 802.2 APIs

Two new tracing parameters, in addition to TRACE and OS2TRACEMASK, have been added to LAPS. They are TRACEOFF and TRACENAMES.

TRACEOFF

This parameter is used to turn off 164 trace events automatically when a specified return code is received. Up to four return code values may be specified. This parameter eliminates the need to turn off tracing manually to prevent the trace buffer from overflowing. Appropriate trace flags must be activated in OS2TRACEMASK for NETBIOS or TRACE for 802.2. Once the TRACEOFF parameter is added to PROTOCOL.INI, the system must be restarted for the parameter to take effect. If your level of NETBEUI or LANDD.OS2 does not support the TRACEOFF parameter, contact IBM support for an updated version.

TRACEOFF can be placed either in the NETBEUI_nif or LANDD_nif sections of the PROTOCOL.INI file. The following lines show the format of the parameter:

```
TRACEOFF = 0xwwxxyyzz   ; where ww, xx, yy, and zz are
                        ; hexadecimal return code values
```

The following example shows the placement and format of TRACEOFF for NETBIOS in the PROTOCOL.INI file:

```
[NETBEUI_nif]

   DriverName = netbeui$
   Bindings = IBMTOK_nif
   ...
   ...
   DLCRETRIES = 5
   OS2TRACEMASK = 0x07FF
   TRACEOFF = 0x18
```

Other valid formats are:

```
TRACEOFF = 0x1805         ; return codes 18 or 05 stop traces
TRACEOFF = 0x0D181911     ; return codes 0D, 18, 19, or 11 stop traces
```

The following example shows the placement and format of TRACEOFF for IEEE 802.2 in the PROTOCOL.INI file:

```
[LANDD_nif]

   DriverName = LANDD$
   ...
   ...
   GDTS = 0030
   TRACE = 0x003F
   TRACEOFF = 0x1C
```

Other valid formats are shown in the following lines:

```
TRACEOFF = 0x1C57         ; return codes 1C or 57 stop traces
TRACEOFF = 0x59212732     ; return codes 59, 21, 27, or 32 stop traces
```

TRACENAMES

This parameter is used to limit tracing activity to the interaction between the local node and specified remote nodes. Before this parameter became available, tracing at the transport level picked up all frame activity on the line and, as a result, the trace buffer was quickly filled with irrelevant frame records. TRACENAMES prevents the buffer from filling up too quickly. You can use this parameter to help isolate a problem between a server and a requester on a large network, and you can use the parameter either on the server or on the requester.

TRACENAMES limits tracing by enabling you to select the NETBIOS names of the remote workstations with which you need to trace the activity. You may specify the names either as 16-byte ASCII or 32-byte hex values. Up to four names may be selected in either format. You can use the wild card character * in the ASCII format to simplify the selection of the names or to increase the number of remote names traced if the workstations have similar names. All the names must be placed on the same line as the TRACENAMES parameter. If the length of the entries is incorrect or if the hex values used are not acceptable, an error occurs and NETBEUI will not load.

TRACENAMES works in conjunction with the OS2TRACEMASK parameter and the TRACEOFF parameter already available for NETBEUI in NTS/2 Version 1.0 and LAN Server Version 3.0. OS2TRACEMASK must be set to the necessary non-zero values to activate any NETBIOS traces. The recommended setting for OS2TRACEMASK when using TRACENAMES is 0x07FF. NDIS frames are not filtered from the trace data if the NDIS trace flags are on.

The TRACENAMES parameter must be inserted in the PROTOCOL.INI file using an editor because it has not yet been included in the LAPS configuration and installation tool. TRACENAMES must be placed only in the NETBEUI_nif section. The following example shows the general location and format of the parameter:

```
PROTOCOL.INI (of workstation server1)

   [NETBEUI_nif]

      DriverName = netbeui$
      Bindings = IBMTOK_nif
      ...
      ...
      DLCRETRIES = 5
      OS2TRACEMASK = 0x07FF
      TRACENAMES= "req1            "  "req2            " "req3            "
```

Some valid TRACENAMES formats are shown in the following lines:

```
TRACENAMES = "72617131202020202020202020202000" "req2           *"
TRACENAMES = "req*************" "72617131202020202020202020202020"
```

Tracing Specifics for TRACENAMES: Activity relevant to the remote NETBIOS names will be traced. Items traced include NCBs issued with one of the specified remote names and all session activity related to that remote name. All statuses from the lower layers are reported, whether related to the remote station or not. Frame activity between the remote and local workstations is tracked by local session number (LSN) after the session with the remote workstation is established. A maximum of 12 active sessions can be tracked.

The NCBs found in the following list are not associated with a remote name or LSN and are not traced:

ADD.NAME
ADD.GROUP.NAME
DELETE.NAME
CANCEL
LAN.STATUS
SESSION.STATUS
RECEIVE.BROADCAST.DATAGRAM
SEND.BROADCAST.DATAGRAM

NCB.RESET is traced even though it is not related to sessions or to specific names. Datagrams and other UI frames are traced if they contain a remote name specified in TRACENAMES. RECEIVE.ANY commands are traced on completion if they report a session event related to a specified remote workstation.

Minor Codes for IEEE 802.2 Traces

The following minor codes are defined for IEEE 802.2:

0x40—CCB RECEIVED ENTRY

This trace entry indicates a CCB received at the CCB3 entry point. CCB2s will also be traced since they are converted to CCB3s and then sent to the CCB3 entry point. The trace data is the contents of the CCB3, except for the CCB_POINTER field. Instead of its actual contents, the CCB_POINTER field will contain the address of the CCB3.

0x41—CCB RECEIVED EXIT

This trace entry indicates the immediate return of a CCB. The trace data is the first 8 bytes of the CCB3, except that the CCB_POINTER field will be set to the address of the CCB3 and the CCB_RETURN_CODE field will be set to 0.

0x42, 0x43—CCB FINAL COMPLETION

This trace entry indicates the final completion of a CCB3. The trace data is the contents of the CCB3, except for the CCB_POINTER field. Instead of its actual contents, the CCB_POINTER field will contain the address of the CCB3. Minor code 43 is used for CCBs that are returning an information table, and code 42 is used for CCBs that are not.

0x44—CCB PARAMETER TABLE

This trace entry will follow a trace entry for minor codes 0x40, 0x41, 0x42, 0x43, and 0x50, if the CCB being traced has a parameter table. The trace data is the CCB parameter table.

0x45—DLC STATUS EVENT

Table 6-1 describes the DLC status event trace entry.

Table 6-1. DLC Status Event Trace Entry

Offset	Byte Length	Field
0	20	DLC status table
20	1	Adapter number
21	1	Application ID
22	2	Reserved

This trace entry indicates a DLC status event. The trace data is the DLC status table, followed by 1 byte for the adapter number, 1 byte for the application ID, and 2 bytes of reserved data.

0x46—EXCEPTION EVENT

Table 6-2 describes the Exception event trace entry:

Table 6-2. Exception Event Trace Entry

Offset	Byte Length	Field
0	20	Exception information table
20	1	Adapter number
21	1	Application ID
22	1	Exception event type
23	1	Reserved

This trace entry indicates one of the exception events controlled by the DIR.SET.EXCEPTION.FLAGS CCB. These events are Adapter Check, Network Status, PC Error, and System Action. The trace data is the CCB3 exception event information table (padded with 0 to 20 bytes in the case of Adapter Check and System Action), followed by 1 byte for the adapter number, 1 byte for the application ID, 1 byte for the exception event type, and 1 byte of reserved data. The exception event type values are:

0	Adapter Check
2	Network Status
4	PC Error
6	System Action.

0x47—CCB INFORMATION TABLE

This trace entry will follow a trace entry for minor code 0x43. The trace data is the CCB information table.

0x48, 0x49—NDIS DATA, RECEIVE

This trace entry indicates data received at the NDIS interface, through either a ReceiveLookahead or a ReceiveChain indication. The trace data is the received frame. The length of the trace is either the first 32 or 128 bytes of the frame, depending on the state of the Data Trace Size flag. If the indication is ReceiveLookahead, then there will be only one trace entry per frame, and it will be minor code 49. If the indication is ReceiveChain, then there will be one trace entry for every buffer in the chain, up to the number of buffers required to reach the 32- or 128-byte trace limit. The last trace entry per frame will be minor code 49 and all others, if any, will be minor code 48.

0x4A, 0x4B—NDIS DATA, TRANSMIT

This trace entry indicates data transmitted at the NDIS interface through the TransmitChain interface. The trace data is the transmitted frame. The length of the trace is either the first 32 or 128 bytes of the frame, depending on the state of the Data Trace Size flag. There will be one trace entry for every buffer in the chain, up to the number of buffers required to reach the 32- or 128-byte trace limit. The last trace entry per frame will be minor code 4B and all others, if any, will be minor code 4A.

0x4C, 0x4D—CCB DATA, RECEIVE

This trace entry indicates data received at the CCB interface, either through a Receive or a Receive_Modify CCB. The trace data is the SAP buffer contents, including buffer headers and frame data. The length of the trace is either the first 64 or 128 bytes of the SAP buffer, depending on the state of the Data Trace Size flag. There will be one trace entry for every SAP buffer in the chain, up to the number of buffers required to reach the 64- or 128-byte trace limit. The last trace entry per frame will be minor code 4D and all others, if any, will be minor code 4C.

0x4E, 0x4F—CCB DATA, TRANSMIT

This trace entry indicates data transmitted at the CCB interface through any of the Transmit CCBs. The trace data is the user-supplied transmit-frame data. The length of the trace is either the first 32 or 128 bytes of the frame, depending on the state of the Data Trace Size flag. There will be one trace entry for every buffer in the frame, up to the number of buffers required to reach the 32- or 128-byte trace limit. The last trace entry per frame will be minor code 4F and all others, if any, will be minor code 4E.

0x50—CCB RECEIVED FRAME EVENT

This trace entry indicates data received at the CCB interface, either through a Receive or a Receive_Modify CCB. The trace data is the Receive This CCB3. entry is only used for Receive CCBs that do not complete when a frame is received. If the CCB completes when the frame is received, whether due to an error or not, only the CCB completion trace entry will be logged.

0x51—DLR CCB RECEIVED ENTRY

This trace entry indicates a CCB received at the CCB2 entry point. The trace data is the CCB2.

PROTOCOL.INI TRACE Parameter—IEEE 802.2

The TRACE parameter in the LANDD section of the PROTOCOL.INI file consists of 10 bit flags. The format of this parameter for each trace event is as follows:

Bit 0 (0x0001)—Validation

Enables debugging-level CCB validation. When enabled, addressability to all buffers defined by the CCB will be verified. This verification is only considered necessary for programs under development, so it defaults to disabled to improve performance.

Bit 1 (0x0002)—CCBs

Enables tracing of CCB contents at CCB3 entry, CCB3 immediate return, and CCB3 final completion, plus CCB3 receive frame events where the Receive CCB3 does not complete.- CCB2s will also be traced because when they are received they are converted to CCB3s and then sent to the CCB3 entry.

Bit 2 (0x0004)—CCB Parameter Tables

Enables tracing of the CCB parameter table contents at CCB entry, CCB final completion, and CCB receive frame events where the Receive CCB does not complete. This bit also enables the tracing of the CCB information table at CCB final completion.

Bit 3 (0x0008)—Status Events

Enables tracing of DLC Status, Adapter Check, Network Status, PC Error, and System Action events.

Bit 4 (0x0010)—DLL CCBs

Enables tracing of CCB2 contents at CCB entry.

Bit 5 (0x0020)—CCB Interface Received Data

Enables tracing of the contents of the SAP buffers containing received data.

Bit 6 (0x0040)—CCB Interface Transmit Data

Enables tracing of the contents of the transmit data supplied with a transmit CCB.

Bit 7 (0x0080)—Data Trace Size

Enables large size data traces for transmit and receive at both the CCB and NDIS interfaces. For Transmit Data traces at both the CCB and NDIS interface, the first 128 bytes of data of each frame are traced if large size is enabled. If large size is disabled, only the first 32 bytes of each frame are traced. For Receive Data traces at the NDIS interface, the first 128 bytes of data of each frame are traced if large size is enabled. If large size is disabled, only the first 32 bytes of each frame are traced. For Receive Data traces at the CCB interface, the first 128 bytes of SAP buffer contents of each frame are traced if large size is enabled. If large size is disabled, only the first 64 bytes of SAP buffer contents are traced. The SAP buffer contents will include header information in addition to the frame data. The SAP buffer contents are described in Chapter 3, "The Command Control Blocks."

Bit 8 (0x0100)—NDIS Interface Received Data

Enables tracing of the contents of the data received from the NDIS interface. This will include user-generated frames destined for the CCB interface, as well as LLC control frames such as SABME and DISC.

Bit 9 (0x0200)—NDIS Interface Transmit Data

Enables tracing of the data sent to the NDIS interface. This will include user-generated frames from the CCB interface, as well as LLC control frames such as SABME and DISC.

Table 6-3 on page 6-17 provides a summary of the IEEE 802.2 trace point minor codes and the TRACE parameter bit flags.

Table 6-3. IEEE 802.2 TRACE Bit Definition

	TRACE Bits									
Trace Point Minor Codes	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0x40 CCB RECEIVED ENTRY		X								
0x41 CCB RECEIVED EXIT		X								
0x42 CCB FINAL COMPLETION		X								
0x43 CCB FINAL COMPLETION		X								
0x44 CCB PARAMETER TABLE			X							
0x45 DLC STATUS EVENT				X						
0x46 EXCEPTION EVENT				X						
0x47 CCB INFORMATION TABLE			X							
0x48 NDIS DATA, RECEIVE								*	X	
0x49 NDIS DATA, RECEIVE LAST								*	X	
0x4A NDIS DATA, TRANSMIT								*		X
0x4B NDIS DATA, TRANSMIT LAST								*		X
0x4C CCB DATA, RECEIVE						X		*		
0x4D CCB DATA, RECEIVE LAST						X		*		
0x4E CCB DATA, TRANSMIT							X	*		
0x4F CCB DATA, TRANSMIT LAST							X	*		
0x50 CCB RECEIVED FRAME EVENT		X								
0x51 DLR CCB RECEIVED ENTRY					X					

* Bit 7 determines the trace data size for all the indicated trace minor codes. If bit 7 is off, the trace data size is 32 bytes, except for CCB Receive Data, which is 64 bytes. If bit 7 is on, the trace data size is 128 bytes.

Refer to *3Com/Microsoft LAN Manager Network Driver Interface Specification Version 2.01 Final* for more information on ReceiveLookahead, ReceiveChain, and TransmitChain.

Minor Codes for NETBIOS Traces

The following minor codes are defined for NETBIOS:

0x60—TRANSMIT I FRAME

The first 14 bytes of the trace data are the NETBIOS I-frame header. The next 2 bytes contain the return code received from the network adapter driver on the TransmitChain primitive. The last 2 bytes are the adapter number.

0x61—TRANSMIT UI FRAME

The first 44 bytes of the trace data are the NETBIOS UI-frame header. The next 2 bytes contain the return code received from the network adapter driver on the TransmitChain primitive. The last 2 bytes are the adapter number.

0x62—TRANSMIT COMPLETE

The first 14 bytes of the trace data are the NETBIOS I-frame header and the last 2 bytes are the adapter number.

0x63—RECEIVE I FRAME

The first 14 bytes of the trace data are the NETBIOS I-frame header and the last 2 bytes are the adapter number.

0x64—RECEIVE UI FRAME

The first 44 bytes of the trace data are the NETBIOS UI-frame header and the last 2 bytes are the adapter number.

0x65—NETWORK STATUS

The first 2 bytes of the trace data are the Ring Status bit mask and the next 2 bytes are the adapter number.

0x66—ADAPTER CHECK

The first 2 bytes of the trace data are the Adapter Check reason code and the next 2 bytes are the adapter number.

0x67—DLC STATUS

Table 6-4 describes the contents of the DLC status event trace entry.

Table 6-4. DLC Status Event Trace Entry

Offset	**Byte Length**	**Field**
0	2	Adapter number
2	2	Reserved
4	2	DLC status
6	5	FRMR data
11	1	Pad
12	6	Reserved

0x68—NCB RECEIVED

This trace entry indicates the NCB was received by the NETBEUI device driver. The first 64 bytes are the NCB contents and the last 4 bytes are the NCB address.

0x69—NCB PENDING

This trace entry indicates the NCB is pending in the NETBEUI device driver. The first 64 bytes are the NCB contents and the last 4 bytes are the NCB address.

0x6A—NCB COMPLETE

This trace entry indicates the NCB is completed by the NETBEUI device driver. The first 64 bytes are the NCB contents and the last 4 bytes are the NCB address.

0x78, 0x79—NDIS DATA, RECEIVE

This trace data is the data received from the NDIS interface. This will also include LLC control frames such as SABME and DISC. The last trace entry per frame will be minor code 0x79 and all others, if any, will be minor code 0x78.

0x7A, 0x7B—NDIS DATA, TRANSMIT

This trace data is the data sent to the NDIS interface. This will also include LLC control frames such as SABME and DISC. The last trace entry per frame will be minor code 0x7B and all others, if any, will be minor code 0x7A.

PROTOCOL.INI OS2TRACEMASK Parameter—NETBIOS

The OS2TRACEMASK parameter in the NETBEUI section of PROTOCOL.INI consists of 16 bit flags. The format of this parameter for each minor code is as follows:

Bit 0 (0x0001)—Transmit I Frame

Enables tracing of NETBIOS I frames transmitted.

Bit 1 (0x0002)—Transmit UI Frame

Enables tracing of NETBIOS UI frames transmitted.

Bit 2 (0x0004)—Transmit Complete

Enables tracing of completion of NETBIOS I frames transmitted.

Bit 3 (0x0008)—Receive I Frame

Enables tracing of NETBIOS I frames received.

Bit 4 (0x0010)—Receive UI Frame

Enables tracing of NETBIOS UI frames received.

Bit 5 (0x0020)—Network Status

Enables tracing of Network Status events.

Bit 6 (0x0040)—Adapter Check

Enables tracing of Adapter Check events.

Bit 7 (0x0080)—DLC Status

Enables tracing of DLC Status events.

Bit 8 (0x0100)—NCB Received

Enables tracing of the NCB contents at entry to the NETBEUI device driver.

Bit 9 (0x0200)—NCB Pending

Enables tracing of the NCB contents at immediate return from the NETBEUI device driver if the NCB return code is X'FF', which indicates a pending state.

Bit 10 (0x0400)—NCB Complete

Enables tracing of the NCB contents at completion of the NCB in the NETBEUI device driver.

Bit 11 (0x0800)—NDIS Data

Enables data traces for transmit and receive at the NDIS interface. This bit enables minor codes X'78' through X'7B'.

Bit 12 (0x1000)—NDIS Size

Enables large size data traces for transmit and receive at the NDIS interface. For Transmit Data traces and for Receive Data traces, the first 128 bytes of data of each frame are traced if large size is enabled. If large size is disabled, only the first 32 bytes of each frame are traced.

Table 6-5 on page 6-20 provides a summary of the NETBIOS trace point minor codes and the OS2TRACEMASK parameter bit flags.

Table 6-5. NETBIOS OS2TRACEMASK Bit Definition

	OS2TRACEMASK Bit Definitions												
Trace Point Minor Codes	**00**	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**12**
0x60 TRANSMIT I FRAME	X												
0x61 TRANSMIT UI FRAME		X											
0x62 TRANSMIT COMPLETE			X										
0x63 RECEIVE I FRAME				X									
0x64 RECEIVE UI FRAME					X								
0x65 NETWORK STATUS						X							
0x66 ADAPTER CHECK							X						
0x67 DLC STATUS								X					
0x68 NCB RECEIVED									X				
0x69 NCB PENDING										X			
0x6A NCB COMPLETE											X		
0x78 NDIS RECEIVE												X	*
0x79 NDIS RECEIVE LAST												X	*
0x7A NDIS TRANSMIT												X	*
0x7B NDIS TRANSMIT LAST												X	*

* Bit 12 determines the trace data size for all of the indicated trace minor codes. If bit 12 is off, the trace data size is 32 bytes. If bit 12 is on, the trace data size is 128 bytes.

Refer to other chapters in this book for documentation about the following topics:

NCB content
NETBIOS UI-frame header content and format
NETBIOS I-frame header content and format
Network Status code definition
Adapter Check reason code definition
DLC Status code definition.

Refer to the *3Com/Microsoft LAN Manager Network Driver Interface Specification Version 2.01 Final* for return code information on the TransmitChain primitive.

Tracing for OS/2 2.0 Virtual DOS LAN Support

Major code 164 (0xA4) enables tracing for OS/2 2.0 Virtual DOS LAN Support. The minor codes 162 (0xA2), 163 (0xA3), and 164 (0xA4) are defined for OS/2 2.0 Virtual DOS LAN Support.

Minor Codes for OS/2 2.0 Virtual DOS LAN Support

The following minor codes are defined for OS/2 2.0 Virtual DOS LAN Support:

0xA2—NCB ISSUED

This trace entry indicates that the NCB issued by DOS was received by OS/2 2.0 Virtual DOS LAN Support. The first 64 bytes are the contents of the DOS NCB. The last 4 bytes are the linear address of the DOS NCB.

0xA3—NCB COMPLETE

This trace entry indicates that the DOS NCB was completed by OS/2 2.0 Virtual DOS LAN Support. The first 64 bytes are the contents of the DOS NCB. The last 4 bytes are the linear address of the DOS NCB.

0xA4—NCB IMMEDIATE RETURN

This trace entry indicates the immediate return from interrupt 5C. The first 64 bytes are the contents of the DOS NCB. The last 4 bytes are the linear address of the DOS NCB.

Appendix A. Valid Commands

The following tables show the valid commands found in control blocks. An X indicates what interfaces the command can be directed to. Some commands sent to NETBIOS can request that the adapter complete the command before allowing the application program to continue. An additional bit (high order) is included in the hexadecimal command code to request this wait for completion.

Notes:

1. Command codes not listed are reserved.
2. The commands used to operate the adapter directly, without the adapter support software, are not listed here. Refer to the books in the Bill of Forms titled *LAN Technical Reference: Adapter Interfaces*, SBOF-6221 as shown in "Related IBM Publications" on page xviii for information about the commands used to operate the supported adapters at the adapter interface.

DLC and Direct Interface Commands

Table A-1 (Page 1 of 2). DLC and Direct Interface Commands

Pseudo Command Name	Page	Hex Code	CCB1	CCB2	CCB3
BUFFER.FREE	3-4	27	x	x	x
BUFFER.GET	3-5	26	x	x	x
DIR.CLOSE.ADAPTER	3-6	04	x	x	x
DIR.CLOSE.DIRECT	3-8	34		x	x
DIR.DEFINE.MIF.ENVIRONMENT	3-9	2B	x		
DIR.INITIALIZE	3-11	20	x	x	x
DIR.INTERRUPT	3-16	00	x	x	x
DIR.MODIFY.OPEN.PARMS	3-16	01	x		
DIR.OPEN.ADAPTER	3-17	03	x	x	x
DIR.OPEN.DIRECT	3-33	35		x	x
DIR.READ.LOG	3-34	08	x	x	x
DIR.RESET.MULT.GROUP.ADDRESS	3-37	42	x		
DIR.RESTORE.OPEN.PARMS	3-37	02	x		
DIR.SET.EXCEPTION.FLAGS	3-38	2D		x	x
DIR.SET.FUNCTIONAL.ADDRESS	3-39	07	x	x	x
DIR.SET.GROUP.ADDRESS	3-41	06	x	x	x
DIR.SET.MULT.GROUP.ADDRESS	3-41	41	x		
DIR.SET.USER.APPENDAGE	3-43	2D	x		
DIR.STATUS	3-44	21	x	x	x
DIR.TIMER.CANCEL	3-51	23	x	x	x
DIR.TIMER.CANCEL.GROUP	3-51	2C	x	x	x
DIR.TIMER.SET	3-52	22	x	x	x
DLC.CLOSE.SAP	3-53	16	x	x	x
DLC.CLOSE.STATION	3-54	1A	x	x	x
DLC.CONNECT.STATION	3-55	1B	x	x	x

Table A-1 (Page 2 of 2). DLC and Direct Interface Commands

Pseudo Command Name	Page	Hex Code	CCB1	CCB2	CCB3
DLC.FLOW.CONTROL	3-56	1D	x	x	x
DLC.MODIFY	3-57	1C	x	x	x
DLC.OPEN.SAP	3-60	15	x	x	x
DLC.OPEN.STATION	3-66	19	x	x	x
DLC.REALLOCATE	3-70	17	x	x	x
DLC.RESET	3-71	14	x	x	x
DLC.SET.THRESHOLD	3-73	33		x	
DLC.STATISTICS	3-74	1E	x	x	x
PDT.TRACE.OFF	3-76	25	x		
PDT.TRACE.ON	3-77	24	x		
PURGE.RESOURCES	3-85	36			x
READ	3-86	31		x	x
READ.CANCEL	3-94	32		x	x
RECEIVE	3-95	28	x	x	x
RECEIVE.CANCEL	3-102	29	x	x	x
RECEIVE.MODIFY	3-103	2A	x		x
TRANSMIT.DIR.FRAME	3-107	0A	x	x	x
TRANSMIT.I.FRAME	3-108	0B	x	x	x
TRANSMIT.TEST.CMD	3-109	11	x	x	x
TRANSMIT.UI.FRAME	3-109	0D	x	x	x
TRANSMIT.XID.CMD	3-109	0E	x	x	x
TRANSMIT.XID.RESP.FINAL	3-110	0F	x	x	x
TRANSMIT.XID.RESP.NOT.FINAL	3-110	10	x	x	x

NETBIOS Commands

Table A-2. NETBIOS Commands

		Hex Code	
Pseudo Command Name	**Page**	**No Wait**	**Wait**
NCB.ADD.GROUP.NAME	4-15	B6	36
NCB.ADD.NAME	4-16	B0	30
NCB.CALL	4-16	90	10
NCB.CANCEL	4-17		35
NCB.CHAIN.SEND	4-18	97	17
NCB.CHAIN.SEND.NO.ACK	4-19	F2	72
NCB.DELETE.NAME	4-20	B1	31
NCB.FIND.NAME	4-21	F8	78
NCB.HANG.UP	4-22	92	12
NCB.LAN.STATUS.ALERT	4-23	F3	
NCB.LISTEN	4-24	91	11
NCB.RECEIVE	4-25	95	15
NCB.RECEIVE.ANY	4-26	96	16
NCB.RECEIVE.BROADCAST.DATAGRAM	4-27	A3	23
NCB.RECEIVE.DATAGRAM	4-27	A1	21
NCB.RESET	4-28		32
NCB.SEND	4-32	94	14
NCB.SEND.BROADCAST.DATAGRAM	4-33	A2	22
NCB.SEND.DATAGRAM	4-34	A0	20
NCB.SEND.NO.ACK	4-34	F1	71
NCB.SESSION.STATUS	4-35	B4	34
NCB.STATUS	4-37	B3	33
NCB.TRACE (For Local Area Network Support Program only)	4-41	F9	79
NCB.UNLINK	4-45		70

Appendix B. Return Codes

About This Appendix B-2
CCB Return Codes Listed by Interface B-2
  CCB Return Codes Listed by Command B-5
    Return Codes for CCB1 Commands B-5
    Return Codes for CCB2 Commands B-8
    Return Codes for CCB3 Commands B-11
  CCB Return Codes Cause and Action B-14
DLC Status Codes B-28
  DLC Status Table B-29
  DLC Status Codes B-30
  Suggested Actions in Response to DLC Status B-31
NCB Return Codes B-32
  NCB Return Codes Listed by Command B-34
  NCB Return Codes Cause and Action B-35
Adapter Status Parameter Table B-44
Frame Status B-46
Exception Indications B-46
Adapter Check B-46
  Adapter Check for CCB1 . . . . . B-46
  Adapter Check for CCB2 . . . . . B-47
  Adapter Check for CCB3 . . . . . B-49
Token-Ring Network Adapter Check Reason Codes for All CCBs B-50
PC Network and Ethernet Adapter Check Reason Codes for All CCBs B-52
Network Status B-52
  Network Status for CCB1 . . . . . B-52
  Network Status for CCB2 . . . . . B-52
  Network Status for CCB3 . . . . . B-54
  Token-Ring Network Status Codes for All CCBs B-55
  PC Network and Ethernet Status Codes for All CCBs B-56
Bring-Up Errors for All CCBs B-57
Token-Ring Network Adapter Open Errors for All CCBs B-58
  Open Error Codes for All CCBs B-58
    Phases B-58
    Errors B-59
  Suggested Actions in Response to Open Errors B-59
    Recommended Actions Table B-63
PC Network and Ethernet Adapter Open Errors for All CCBs B-63
PC System Detected Errors B-63
  PC System Detected Errors for CCB1 . . . . . B-63
  PC System Detected Errors for CCB2 . . . . . B-64
  PC System Detected Errors for CCB3 . . . . . B-67
System Action Exceptions for OS/2 EE 1.3 . . . . . B-69
  System Action Exceptions for CCB2 . . . . . B-69
  System Action Exceptions for CCB3 . . . . . B-72

About This Appendix

This appendix includes:

- DLC and direct interface return codes
- NETBIOS return codes
- Other reason and status codes
- Exception indications
- Formats of special returned tables.

CCB Return Codes Listed by Interface

Table B-1 (Page 1 of 3). CCB Return Codes Listed by Interface

Code	Meaning	CCB1	CCB2	CCB3
00	Operation completed successfully	X	X	X
01	Invalid command code	X	X	X
02	Duplicate command, one already outstanding	X	X	X
03	Adapter open, should be closed	X	X	X
04	Adapter closed, should be open	X	X	
05	Required parameter not provided	X	X	X
06	Option invalid or incompatible	X	X	X
07	Command canceled, unrecoverable failure	X	X	X
08	Unauthorized access priority	X	X	X
09	Adapter not initialized, should be initialized	X		
0A	Command canceled by user request	X	X	X
0B	Command canceled, adapter closed while command in progress	X	X	X
0C	Command completed successfully, adapter not open	X		
10	Adapter open, NETBIOS not operational	X		
11	DIR.TIMER.SET or DIR.TIMER.CANCEL error	X	X	X
12	Available work area exceeded	X		
13	Invalid LOG.ID	X	X	X
14	Invalid shared RAM segment or size	X		
15	Lost log data, inadequate buffer space, log reset	X	X	X
16	Requested buffer size exceeds pool length	X	X	X
17	Command invalid, NETBIOS operational	X		
18	Invalid buffer length	X	X	X
19	Inadequate buffers available for request	X	X	X
1A	User length too large for buffer length	X	X	X
1B	The CCB_PARM_TAB pointer is invalid	X	X	X
1C	A pointer in the CCB parm table is invalid	X	X	X
1D	Invalid CCB_ADAPTER value	X	X	X
1E	Invalid functional address	X		
20	Lost data on receive, no buffers available	X	X	X

Table B-1 (Page 2 of 3). CCB Return Codes Listed by Interface

Code	Meaning	CCB1	CCB2	CCB3
21	Lost data on receive, inadequate buffer space	X	X	X
22	Error on frame transmission, check TRANSMIT.FS data	X	X	X
23	Error in frame transmit or strip process	X	X	X
24	Unauthorized MAC frame	X	X	X
25	Maximum commands exceeded	X	X	X
26	Unrecognized command correlator	X		
27	Link not transmitting I frames, state changed from link opened	X	X	X
28	Invalid transmit frame length	X	X	X
30	Inadequate receive buffers for adapter to open	X	X	X
32	Invalid NODE_ADDRESS	X	X	X
33	Invalid adapter receive buffer length defined	X	X	X
34	Invalid adapter transmit buffer length defined	X	X	X
35	Unable to set group addresses	X		
36	Invalid group address	X		
40	Invalid STATION_ID	X	X	X
41	Protocol error, link in invalid state for command	X	X	X
42	Parameter exceeded maximum allowed	X	X	X
43	Invalid SAP_VALUE or value already in use	X	X	X
44	Invalid routing information field	X	X	X
45	Requested group membership in non-existent group SAP	X	X	X
46	Resources not available	X	X	X
47	SAP cannot close unless all link stations are closed	X	X	X
48	Group SAP cannot close, individual SAPs not closed	X	X	X
49	Group SAP has reached maximum membership	X	X	X
4A	Sequence error: incompatible command in progress	X	X	X
4B	Station closed without remote acknowledgment	X	X	X
4C	Sequence error, cannot close, DLC commands outstanding	X	X	X
4D	Unsuccessful link station connection attempted	X	X	X
4E	Member SAP not found in group SAP list	X	X	X
4F	Invalid remote address, may not be a group address	X	X	X
50	Invalid pointer in CCB_POINTER field		X	
52	Invalid application program ID		X	X
53	Invalid application program key code		X	X
54	Invalid System Key code		X	X
55	Buffer is smaller than buffer size given on DLC.OPEN.SAP		X	X
56	Adapter's system process is not installed		X	
57	Inadequate stations available		X	X
58	Invalid CCB_PARAMETER_1 parameter		X	X
59	Inadequate queue elements to satisfy request		X	X

Table B-1 (Page 3 of 3). CCB Return Codes Listed by Interface

Code	Meaning	CCB1	CCB2	CCB3
5A	Initialization failure, cannot open adapter		X	X
5B	Error detected in chained READ command		X	
5C	Direct stations not assigned to application program		X	X
5D	DD interface not installed		X	
5E	Requested adapter is not installed		X	X
5F	Chained CCBs must all be for the same adapter		X	
60	Adapter initializing, command not accepted		X	X
61	Number of allowed application programs has been exceeded		X	X
62	Command canceled by system action		X	X
63	Direct stations not available		X	X
64	Invalid DDNAME parameter		X	X
65	Inadequate GDT selectors to satisfy request		X	X
67	Command cancelled, CCB resources purged			X
68	Application program ID not valid for interface		X	X
69	Segment associated with request cannot be locked		X	
FF	Command in progress	X	X	X

CCB Return Codes Listed by Command

Return Codes for CCB1 Commands

	Return Codes in Hex																												
CCB1 Commands	**00**	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0A**	**0B**	**0C**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**1A**	**1B**	**1C**	**1D**	**1E**	**20**
BUFFER.FREE	x	x			x					x															x	x	x		
BUFFER.GET	x	x			x					x													x		x		x		
DLC.CLOSE.SAP	x	x			x			x		x		x															x		
DLC.CLOSE.STATION	x	x			x			x		x		x															x		
DLC.CONNECT.STATION	x	x	x		x			x		x		x													x	x	x		
DLC.FLOW.CONTROL	x	x			x			x		x		x															x		
DLC.MODIFY	x	x			x			x	x	x		x													x		x		
DLC.OPEN.SAP	x	x			x		x	x	x	x		x								x		x			x	x	x		
DLC.OPEN.STATION	x	x			x	x		x	x	x		x													x	x	x		
DLC.REALLOCATE	x	x			x			x	x																		x		
DLC.RESET	x	x			x			x		x		x													x		x		
DLC.STATISTICS	x	x			x			x		x		x							x						x		x		
RECEIVE	x	x	x		x			x		x	x	x												x	x	x	x		x
RECEIVE.CANCEL	x	x			x					x																	x		
RECEIVE.MODIFY	x	x	x		x			x		x	x	x												x	x	x	x		x
TRANSMIT.DIR.FRAME	x	x			x			x	x	x	x	x													x		x		
TRANSMIT.I.FRAME	x	x			x			x	x	x	x	x													x		x		
TRANSMIT.TEST.CMD	x	x			x			x	x	x	x	x													x		x		
TRANSMIT.UI.FRAME	x	x			x			x	x	x	x	x													x		x		
TRANSMIT.XID.CMD	x	x			x			x	x	x	x	x													x		x		
TRANSMIT.XID.RESP.FINAL	x	x			x			x	x	x	x	x													x		x		
TRANSMIT.XID.RESP.NOT.FINAL	x	x			x			x	x	x	x	x													x		x		
DIR.CLOSE.ADAPTER	x	x			x	x		x		x																	x		
DIR.DEFINE.MIF.ENVIRONMENT	x	x																							x	x			
DIR.INITIALIZE	x	x	x					x										x							x	x	x		
DIR.INTERRUPT	x	x						x		x																	x		
DIR.MODIFY.OPEN.PARMS	x	x	x		x			x		x										x		x			x	x	x		
DIR.OPEN.ADAPTER	x	x	x	x		x		x		x				x		x				x		x			x	x	x	x	
DIR.READ.LOG	x	x			x			x		x		x					x		x						x		x		
DIR.RESET.MULT.GROUP.ADDRESS	x	x			x			x		x		x															x		
DIR.RESTORE.OPEN.PARMS	x	x			x		x	x		x																	x		
DIR.SET.FUNCTIONAL.ADDRESS	x	x			x			x		x		x															x	x	
DIR.SET.GROUP.ADDRESS	x	x			x			x		x		x															x		
DIR.SET.MULT.GROUP.ADDRESS	x	x			x			x		x		x															x		
DIR.SET.USER.APPENDAGE	x	x			x			x		x															x		x		
DIR.STATUS	x	x								x			x												x		x		
DIR.TIMER.CANCEL	x	x								x					x												x		
DIR.TIMER.CANCEL.GROUP	x	x								x																	x		
DIR.TIMER.SET	x	x						x		x	x	x			x												x		
PDT.TRACE.ON	x		x				x				x														x				
PDT.TRACE.OFF	x																												
CCB1 Commands	**00**	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0A**	**0B**	**0C**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**1A**	**1B**	**1C**	**1D**	**1E**	**20**
	Return Codes in Hex																												

CCB Return Codes

	Return Codes in Hex																											
CCB1 Commands	21	22	23	24	25	26	27	28	30	31	32	33	34	35	36	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D
BUFFER.FREE																x													
BUFFER.GET																x													
DLC.CLOSE.SAP																x							x	x				x	
DLC.CLOSE.STATION																x							x				x	x	
DLC.CONNECT.STATION																x	x			x						x			x
DLC.FLOW.CONTROL																x													
DLC.MODIFY																x		x			x				x				
DLC.OPEN.SAP																		x	x		x	x			x				
DLC.OPEN.STATION																x		x	x			x							
DLC.REALLOCATE																x													
DLC.RESET																x													
DLC.STATISTICS																x													
RECEIVE	x															x													
RECEIVE.CANCEL																x													
RECEIVE.MODIFY	x															x													
TRANSMIT.DIR.FRAME		x	x	x	x		x	x								x	x			x						x			
TRANSMIT.I.FRAME		x	x	x	x		x	x								x	x			x						x			
TRANSMIT.TEST.CMD		x	x	x	x		x	x								x	x			x						x			
TRANSMIT.UI.FRAME		x	x	x	x		x	x								x	x			x						x			
TRANSMIT.XID.CMD		x	x	x	x		x	x								x	x			x						x			
TRANSMIT.XID.RESP.FINAL		x	x	x	x		x	x								x	x			x						x			
TRANSMIT.XID.RESP.NOT.FINAL		x	x	x	x		x	x								x	x			x						x			
DIR.CLOSE.ADAPTER																													
DIR.DEFINE.MIF.ENVIRONMENT																													
DIR.INITIALIZE																													
DIR.INTERRUPT																													
DIR.MODIFY.OPEN.PARMS																													
DIR.OPEN.ADAPTER									x		x	x	x																
DIR.READ.LOG																													
DIR.RESET.MULT.GROUP.ADDRESS														x	x														
DIR.RESTORE.OPEN.PARMS																													
DIR.SET.FUNCTIONAL.ADDRESS																													
DIR.SET.GROUP.ADDRESS																													
DIR.SET.MULT.GROUP.ADDRESS														x	x														
DIR.SET.USER.APPENDAGE																													
DIR.STATUS																													
DIR.TIMER.CANCEL																													
DIR.TIMER.CANCEL.GROUP																													
DIR.TIMER.SET																													
PDT.TRACE.ON																													
PDT.TRACE.OFF																													
CCB1 Commands	21	22	23	24	25	26	27	28	30	31	32	33	34	35	36	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D
	Return Codes in Hex																												

	Return Codes in Hex																												
CCB1 Commands	**4E**	**4F**	**FF**																										
BUFFER.FREE																													
BUFFER.GET																													
DLC.CLOSE.SAP																													
DLC.CLOSE.STATION																													
DLC.CONNECT.STATION																													
DLC.FLOW.CONTROL																													
DLC.MODIFY	x																												
DLC.OPEN.SAP																													
DLC.OPEN.STATION		x																											
DLC.REALLOCATE																													
DLC.RESET																													
DLC.STATISTICS																													
RECEIVE																													
RECEIVE.CANCEL																													
RECEIVE.MODIFY																													
TRANSMIT.DIR.FRAME																													
TRANSMIT.I.FRAME																													
TRANSMIT.TEST.CMD																													
TRANSMIT.UI.FRAME																													
TRANSMIT.XID.CMD																													
TRANSMIT.XID.RESP.FINAL																													
TRANSMIT.XID.RESP.NOT.FINAL																													
DIR.CLOSE.ADAPTER																													
DIR.DEFINE.MIF.ENVIRONMENT																													
DIR.INITIALIZE																													
DIR.INTERRUPT																													
DIR.MODIFY.OPEN.PARMS																													
DIR.OPEN.ADAPTER																													
DIR.READ.LOG																													
DIR.RESET.MULT.GROUP.ADDRESS																													
DIR.RESTORE.OPEN.PARMS																													
DIR.SET.FUNCTIONAL.ADDRESS																													
DIR.SET.GROUP.ADDRESS																													
DIR.SET.MULT.GROUP.ADDRESS																													
DIR.SET.USER.APPENDAGE																													
DIR.STATUS																													
DIR.TIMER.CANCEL																													
DIR.TIMER.CANCEL.GROUP																													
DIR.TIMER.SET																													
PDT.TRACE.ON																													
PDT.TRACE.OFF																													
CCB1 Commands	**4E**	**4F**	**FF**																										
	Return Codes in Hex																												

Return Codes for CCB2 Commands

	Return Codes in Hex																												
CCB2 Commands	00	01	02	03	04	05	06	07	08	0A	0B	0C	11	13	15	16	18	19	1A	1B	1C	1D	20	21	22	23	24	25	27
BUFFER.FREE	x	x			x			x												x	x	x							
BUFFER.GET	x	x			x			x										x		x		x							
DLC.CLOSE.SAP	x	x			x			x			x											x							
DLC.CLOSE.STATION	x	x	x		x			x			x											x							
DLC.CONNECT.STATION	x	x	x		x	x		x			x									x	x	x							
DLC.FLOW.CONTROL	x	x			x			x			x											x							
DLC.MODIFY	x	x			x	x		x	x		x									x	x	x							
DLC.OPEN.SAP	x	x			x	x	x	x	x		x					x	x			x	x	x							
DLC.OPEN.STATION	x	x			x	x		x	x		x									x	x	x							
DLC.REALLOCATE	x	x			x	x		x			x									x	x	x							
DLC.RESET	x	x			x			x			x											x							
DLC.SET.THRESHOLD	x	x			x	x		x												x		x							
DLC.STATISTICS	x	x			x	x		x			x				x					x	x	x							
READ	x	x			x	x	x	x		x	x									x		x							
READ.CANCEL	x	x			x			x														x							
RECEIVE	x	x	x		x	x	x	x		x	x								x	x		x	x	x					
RECEIVE.CANCEL	x	x			x			x														x							
TRANSMIT.DIR.FRAME	x	x			x	x	x	x	x		x									x	x	x			x	x	x	x	x
TRANSMIT.I.FRAME	x	x			x	x	x	x	x		x									x	x	x			x	x	x	x	x
TRANSMIT.TEST.CMD	x	x			x	x	x	x	x		x									x	x	x			x	x	x	x	x
TRANSMIT.UI.FRAME	x	x			x	x	x	x	x		x									x	x	x			x	x	x	x	x
TRANSMIT.XID.CMD	x	x			x	x	x	x	x		x									x	x	x			x	x	x	x	x
TRANSMIT.XID.RESP.FINAL	x	x			x	x	x	x	x		x									x	x	x			x	x	x	x	x
TRANSMIT.XID.RESP.NOT.FINAL	x	x			x	x	x	x	x		x									x	x	x			x	x	x	x	x
DIR.CLOSE.ADAPTER	x	x			x			x														x							
DIR.CLOSE.DIRECT	x	x			x			x			x											x							
DIR.INITIALIZE	x	x				x		x												x		x							
DIR.INTERRUPT	x	x			x			x			x											x							
DIR.OPEN.ADAPTER	x	x		x		x	x	x												x	x	x							
DIR.OPEN.DIRECT	x	x			x	x	x	x	x		x					x	x			x	x	x							
DIR.READ.LOG	x	x			x	x		x			x			x	x					x	x	x							
DIR.SET.EXCEPTION.FLAGS	x	x			x	x		x												x		x							
DIR.SET.FUNCTIONAL.ADDRESS	x	x			x			x			x											x							
DIR.SET.GROUP.ADDRESS	x	x			x			x			x											x							
DIR.STATUS	x	x			x	x		x				x								x		x							
DIR.TIMER.CANCEL	x	x			x			x					x									x							
DIR.TIMER.CANCEL.GROUP	x	x			x			x					x									x							
DIR.TIMER.SET	x	x			x			x		x	x		x									x							
CCB2 Commands	00	01	02	03	04	05	06	07	08	0A	0B	0C	11	13	15	16	18	19	1A	1B	1C	1D	20	21	22	23	24	25	27
	Return Codes in Hex																												

	Return Codes in Hex																												
CCB2 Commands	**28**	**30**	**32**	**33**	**34**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**4A**	**4B**	**4C**	**4D**	**4E**	**4F**	**50**	**52**	**53**	**54**	**55**	**56**	**57**	**58**
BUFFER.FREE						x																x	x	x		x	x		
BUFFER.GET						x																x	x	x			x		
DLC.CLOSE.SAP						x							x	x				x				x	x	x			x		
DLC.CLOSE.STATION						x											x	x				x	x	x			x		
DLC.CONNECT.STATION						x	x			x						x			x			x	x	x			x		
DLC.FLOW.CONTROL						x																x	x	x			x		
DLC.MODIFY						x		x			x				x					x		x	x	x			x		
DLC.OPEN.SAP								x	x		x	x			x							x	x	x			x		
DLC.OPEN.STATION						x		x	x			x									x	x	x	x			x		
DLC.REALLOCATE						x																x	x	x			x	x	
DLC.RESET						x																x	x	x	x		x		
DLC.SET.THRESHOLD						x																x	x	x			x		
DLC.STATISTICS						x																x	x	x			x		
READ						x																x	x	x			x		
READ.CANCEL																						x	x	x			x		x
RECEIVE						x																x	x	x			x		
RECEIVE.CANCEL						x																x	x	x			x		
TRANSMIT.DIR.FRAME	x					x	x			x						x						x	x	x		x	x		
TRANSMIT.I.FRAME	x					x	x			x						x						x	x	x		x	x		
TRANSMIT.TEST.CMD	x					x	x			x						x						x	x	x		x	x		
TRANSMIT.UI.FRAME	x					x	x			x						x						x	x	x		x	x		
TRANSMIT.XID.CMD	x					x	x			x						x						x	x	x		x	x		
TRANSMIT.XID.RESP.FINAL	x					x	x			x						x						x	x	x		x	x		
TRANSMIT.XID.RESP.NOT.FINAL	x					x	x			x						x						x	x	x		x	x		
DIR.CLOSE.ADAPTER																						x	x	x	x		x		
DIR.CLOSE.DIRECT																						x	x	x	x		x		
DIR.INITIALIZE																						x			x		x		
DIR.INTERRUPT																						x	x	x			x		
DIR.OPEN.ADAPTER		x	x	x	x																	x					x		
DIR.OPEN.DIRECT								x														x	x	x			x		
DIR.READ.LOG																						x	x	x	x		x		
DIR.SET.EXCEPTION.FLAGS																						x	x	x			x		
DIR.SET.FUNCTIONAL.ADDRESS																						x	x	x	x		x		
DIR.SET.GROUP.ADDRESS																						x	x	x	x		x		
DIR.STATUS																						x	x	x			x		
DIR.TIMER.CANCEL																						x	x	x			x		
DIR.TIMER.CANCEL.GROUP																						x	x	x			x		
DIR.TIMER.SET																						x	x	x			x		
CCB2 Commands	**28**	**30**	**32**	**33**	**34**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**4A**	**4B**	**4C**	**4D**	**4E**	**4F**	**50**	**52**	**53**	**54**	**55**	**56**	**57**	**58**
	Return Codes in Hex																												

CCB Return Codes

	Return Codes in Hex															
CCB2 Commands	**59**	**5A**	**5B**	**5C**	**5D**	**5E**	**5F**	**60**	**61**	**62**	**63**	**64**	**65**	**68**	**69**	**FF**
BUFFER.FREE	x		x		x	x	x						x	x	x	
BUFFER.GET	x		x		x	x	x						x	x	x	
DLC.CLOSE.SAP	x		x		x	x	x			x			x	x	x	
DLC.CLOSE.STATION	x		x		x	x	x			x			x	x	x	
DLC.CONNECT.STATION	x		x		x	x	x			x			x	x	x	
DLC.FLOW.CONTROL	x		x		x	x	x			x			x	x	x	
DLC.MODIFY	x		x		x	x	x			x			x	x	x	
DLC.OPEN.SAP	x		x		x	x	x			x			x	x	x	
DLC.OPEN.STATION	x		x		x	x	x			x			x	x	x	
DLC.REALLOCATE	x		x		x	x	x			x			x			
DLC.RESET	x		x		x	x	x			x			x	x	x	
DLC.SET.THRESHOLD	x		x		x	x	x						x	x	x	
DLC.STATISTICS	x		x		x	x	x			x			x	x	x	
READ	x				x	x	x			x			x	x	x	
READ.CANCEL	x				x	x	x						x	x	x	
RECEIVE	x		x		x	x	x			x			x	x	x	
RECEIVE.CANCEL	x		x		x	x	x						x	x	x	
TRANSMIT.DIR.FRAME	x		x		x	x	x			x			x	x	x	
TRANSMIT.I.FRAME	x		x		x	x	x			x			x	x	x	
TRANSMIT.TEST.CMD	x		x		x	x	x			x			x	x	x	
TRANSMIT.UI.FRAME	x		x		x	x	x			x			x	x	x	
TRANSMIT.XID.CMD	x		x		x	x	x			x			x	x	x	
TRANSMIT.XID.RESP.FINAL	x		x		x	x	x			x			x	x	x	
TRANSMIT.XID.RESP.NOT.FINAL	x		x		x	x	x			x			x	x	x	
DIR.CLOSE.ADAPTER	x		x		x	x	x						x	x	x	
DIR.CLOSE.DIRECT	x		x	x	x	x	x			x			x	x	x	
DIR.INITIALIZE	x				x	x	x						x		x	
DIR.INTERRUPT	x		x		x	x	x			x			x	x	x	
DIR.OPEN.ADAPTER	x	x			x	x	x	x	x				x		x	
DIR.OPEN.DIRECT	x		x		x	x	x			x	x		x	x	x	
DIR.READ.LOG	x		x	x	x	x	x			x			x	x	x	
DIR.SET.EXCEPTION.FLAGS	x		x		x	x	x						x	x	x	
DIR.SET.FUNCTIONAL.ADDRESS	x		x		x	x	x			x			x	x	x	
DIR.SET.GROUP.ADDRESS	x		x		x	x	x			x			x	x	x	
DIR.STATUS	x		x		x	x	x						x	x	x	
DIR.TIMER.CANCEL	x		x		x	x	x						x	x	x	
DIR.TIMER.CANCEL.GROUP	x		x		x	x	x						x	x	x	
DIR.TIMER.SET	x		x		x	x	x			x			x	x	x	
CCB2 Commands	**59**	**5A**	**5B**	**5C**	**5D**	**5E**	**5F**	**60**	**61**	**62**	**63**	**64**	**65**	**68**	**69**	**FF**
	Return Codes in Hex															

Return Codes for CCB3 Commands

	Return Codes in Hex																												
CCB3 Commands	**00**	**01**	**02**	**03**	**05**	**06**	**07**	**08**	**0A**	**0B**	**0C**	**11**	**13**	**15**	**16**	**18**	**19**	**1A**	**1B**	**1C**	**1D**	**20**	**21**	**22**	**23**	**24**	**25**	**27**	**28**
BUFFER.FREE	x	x					x												x	x	x								
BUFFER.GET	x	x			x		x										x		x		x								
DLC.CLOSE.SAP	x	x					x			x											x								
DLC.CLOSE.STATION	x	x	x				x			x											x								
DLC.CONNECT.STATION	x	x	x		x		x			x									x	x	x								
DLC.FLOW.CONTROL	x	x					x			x											x								
DLC.MODIFY	x	x			x		x	x		x									x	x	x								
DLC.OPEN.SAP	x	x			x	x	x	x		x					x	x			x	x	x								
DLC.OPEN.STATION	x	x			x		x	x		x									x	x	x								
DLC.REALLOCATE	x	x			x		x			x									x	x	x								
DLC.RESET	x	x					x			x											x								
DLC.STATISTICS	x	x			x		x			x				x					x	x	x								
PURGE.RESOURCES	x	x					x														x								
RECEIVE	x	x	x		x	x	x		x	x								x	x		x	x	x						
RECEIVE.CANCEL	x	x					x														x								
RECEIVE.MODIFY	x	x	x		x		x		x	x								x	x		x	x	x						
TRANSMIT.DIR.FRAME	x	x			x	x	x	x		x									x	x	x			x	x	x	x	x	x
TRANSMIT.I.FRAME	x	x			x	x	x	x		x									x	x	x			x	x	x	x	x	x
TRANSMIT.TEST.CMD	x	x			x	x	x	x		x									x	x	x			x	x	x	x	x	x
TRANSMIT.UI.FRAME	x	x			x	x	x	x		x									x	x	x			x	x	x	x	x	x
TRANSMIT.XID.CMD	x	x			x	x	x	x		x									x	x	x			x	x	x	x	x	x
TRANSMIT.XID.RESP.FINAL	x	x			x	x	x	x		x									x	x	x			x	x	x	x	x	x
TRANSMIT.XID.RESP.NOT.FINAL	x	x			x	x	x	x		x									x	x	x			x	x	x	x	x	x
DIR.CLOSE.ADAPTER	x	x					x														x								
DIR.CLOSE.DIRECT	x	x					x			x											x								
DIR.INITIALIZE	x	x			x		x												x		x								
DIR.INTERRUPT	x	x					x			x											x								
DIR.OPEN.ADAPTER	x	x			x	x	x												x	x	x								
DIR.OPEN.DIRECT	x	x			x	x	x	x		x					x	x			x	x	x								
DIR.READ.LOG	x	x			x		x			x			x	x					x	x	x								
DIR.SET.EXCEPTION.FLAGS	x	x			x		x												x		x								
DIR.SET.FUNCTIONAL.ADDRESS	x	x					x			x											x								
DIR.SET.GROUP.ADDRESS	x	x					x			x											x								
DIR.STATUS	x	x			x		x				x								x		x								
DIR.TIMER.CANCEL	x	x					x					x									x								
DIR.TIMER.CANCEL.GROUP	x	x					x					x									x								
DIR.TIMER.SET	x	x					x		x	x		x									x								
CCB3 Commands	**00**	**01**	**02**	**03**	**05**	**06**	**07**	**08**	**0A**	**0B**	**0C**	**11**	**13**	**15**	**16**	**18**	**19**	**1A**	**1B**	**1C**	**1D**	**20**	**21**	**22**	**23**	**24**	**25**	**27**	**28**
	Return Codes in Hex																												

CCB Return Codes

	Return Codes in Hex																												
CCB3 Commands	**30**	**32**	**33**	**34**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**4A**	**4B**	**4C**	**4D**	**4E**	**4F**	**52**	**53**	**54**	**55**	**57**	**58**	**59**	**5A**	**5C**
BUFFER.FREE					x																x	x		x			x		
BUFFER.GET					x																x	x					x		
DLC.CLOSE.SAP					x							x	x				x				x	x					x		
DLC.CLOSE.STATION					x											x	x				x	x					x		
DLC.CONNECT.STATION					x	x			x						x			x			x	x					x		
DLC.FLOW.CONTROL					x																x	x					x		
DLC.MODIFY					x		x			x				x					x		x	x					x		
DLC.OPEN.SAP							x	x		x	x			x							x	x					x		
DLC.OPEN.STATION					x		x	x			x									x	x	x					x		
DLC.REALLOCATE					x																x	x			x		x		
DLC.RESET					x																x	x	x				x		
DLC.STATISTICS					x																x	x					x		
PURGE.RESOURCES																					x	x					x		
RECEIVE					x																x	x					x		
RECEIVE.CANCEL					x																x	x					x		
RECEIVE.MODIFY					x																x	x					x		
TRANSMIT.DIR.FRAME					x	x			x						x						x	x		x			x		
TRANSMIT.I.FRAME					x	x			x						x						x	x		x			x		
TRANSMIT.TEST.CMD					x	x			x						x						x	x		x			x		
TRANSMIT.UI.FRAME					x	x			x						x						x	x		x			x		
TRANSMIT.XID.CMD					x	x			x						x						x	x		x			x		
TRANSMIT.XID.RESP.FINAL					x	x			x						x						x	x		x			x		
TRANSMIT.XID.RESP.NOT.FINAL					x	x			x						x						x	x		x			x		
DIR.CLOSE.ADAPTER																					x	x	x				x		
DIR.CLOSE.DIRECT																					x	x	x				x		x
DIR.INITIALIZE																							x				x		
DIR.INTERRUPT																					x	x					x		
DIR.OPEN.ADAPTER	x	x	x	x																							x	x	
DIR.OPEN.DIRECT							x														x	x					x		
DIR.READ.LOG																					x	x	x				x		x
DIR.SET.EXCEPTION.FLAGS																					x	x					x		
DIR.SET.FUNCTIONAL.ADDRESS																					x	x	x				x		
DIR.SET.GROUP.ADDRESS																					x	x	x				x		
DIR.STATUS																					x	x					x		
DIR.TIMER.CANCEL																					x	x					x		
DIR.TIMER.CANCEL.GROUP																					x	x					x		
DIR.TIMER.SET																					x	x					x		
CCB3 Commands	**30**	**32**	**33**	**34**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**4A**	**4B**	**4C**	**4D**	**4E**	**4F**	**52**	**53**	**54**	**55**	**57**	**58**	**59**	**5A**	**5C**
	Return Codes in Hex																												

	Return Codes in Hex									
CCB3 Commands	**5E**	**60**	**61**	**62**	**63**	**64**	**65**	**67**	**68**	**FF**
BUFFER.FREE	x								x	
BUFFER.GET	x								x	
DLC.CLOSE.SAP	x			x				x	x	
DLC.CLOSE.STATION	x			x				x	x	
DLC.CONNECT.STATION	x			x				x	x	
DLC.FLOW.CONTROL	x			x				x	x	
DLC.MODIFY	x			x				x	x	
DLC.OPEN.SAP	x			x				x	x	
DLC.OPEN.STATION	x			x				x	x	
DLC.REALLOCATE	x			x				x	x	
DLC.RESET	x			x				x	x	
DLC.STATISTICS	x			x				x	x	
PURGE.RESOURCES	x								x	
RECEIVE	x			x				x	x	
RECEIVE.CANCEL	x								x	
RECEIVE.MODIFY	x			x			x	x	x	
TRANSMIT.DIR.FRAME	x			x				x	x	
TRANSMIT.I.FRAME	x			x				x	x	
TRANSMIT.TEST.CMD	x			x				x	x	
TRANSMIT.UI.FRAME	x			x				x	x	
TRANSMIT.XID.CMD	x			x				x	x	
TRANSMIT.XID.RESP.FINAL	x			x				x	x	
TRANSMIT.XID.RESP.NOT.FINAL	x			x				x	x	
DIR.CLOSE.ADAPTER	x								x	
DIR.CLOSE.DIRECT	x			x				x	x	
DIR.INITIALIZE	x									
DIR.INTERRUPT	x			x				x	x	
DIR.OPEN.ADAPTER	x	x	x			x				
DIR.OPEN.DIRECT	x			x	x			x	x	
DIR.READ.LOG	x			x				x	x	
DIR.SET.EXCEPTION.FLAGS	x								x	
DIR.SET.FUNCTIONAL.ADDRESS	x			x				x	x	
DIR.SET.GROUP.ADDRESS	x			x				x	x	
DIR.STATUS	x								x	
DIR.TIMER.CANCEL	x								x	
DIR.TIMER.CANCEL.GROUP	x								x	
DIR.TIMER.SET	x			x				x	x	
CCB3 Commands	**5E**	**60**	**61**	**62**	**63**	**64**	**65**	**67**	**68**	**FF**
	Return Codes in Hex									

CCB Return Codes Cause and Action

Hex 00

Explanation: Operation was completed successfully.

Hex 01

Explanation: Invalid command code.

Cause: CCB_COMMAND did not contain a recognized command code.

Action: Try again, using a valid command.

Hex 02

Explanation: Duplicate command—one is already pending.

Cause: Only one command of this type can be pending at one time.

Action: Wait for the previously issued command to be completed.

Hex 03

Explanation: Adapter open—it should be closed.

Cause: This command can be issued only when the adapter is not open.

Action: Close the adapter or issue the correct command.

Note: For an exception to the above, see the DIR.OPEN.ADAPTER command on page 3-17.

Hex 04

Explanation: Adapter closed—it should be open.

Cause: This command can be issued only when the adapter is open.

Action: Open the adapter.

Hex 05

Explanation: Required parameters not provided.

Cause: At least one required parameter for which no default is available is coded as zero.

Action: Correct the value and try again.

Hex 06

Explanation: Options invalid or incompatible.

Cause: The options selected are not a valid combination. For example, this return code can occur if an attempt is made to open a SAP if that SAP has an XID command handling option different from that of the GSAP it is associated with. In that case, the command will have been completed up to the point where the failing item in the GSAP list was encountered. Otherwise, no action will have been taken for the command.

Action: Correct the options and try again, or issue a DLC.MODIFY command for the remaining GSAP list members.

Hex 07

Explanation: Command canceled—unrecoverable failure.

Cause: The adapter has been closed because of an error condition.

Action: Analyze the error indications. If the error is not permanent, issue the commands DIR.INITIALIZE (for DOS) and DIR.OPEN.ADAPTER (for DOS and OS/2).

Hex 08

Explanation: Unauthorized access priority.

Cause: The requested access priority has not been authorized.

Action: Lower the priority specified with either the OPTIONS_PRIORITY or the ACCESS_PRIORITY value and reissue the command.

Hex 09

Explanation: Adapter not initialized—it should be initialized.

Cause: This command can be completed only if the adapter is initialized.

Action: Issue the DIR.INITIALIZE command.

Hex 0A

Explanation: Command canceled by user request.

Cause: This is the expected response when a command is canceled by an application program command.

Action: None.

Hex 0B

Explanation: Command canceled—the adapter closed while a command was in progress.

Cause: A DIR.CLOSE.ADAPTER command was issued while this command was in progress.

Action: As appropriate for the application program.

Hex 0C

Explanation: Command was completed successfully—adapter not open.

Cause: Information only. The command may execute even though the adapter is not open.

Action: None.

Hex 10

Explanation: Adapter open—NETBIOS not operational.

Cause: One of the following:

- The DIR.OPEN.ADAPTER command was passed NETBIOS parameters and NETBIOS code is not loaded.
- One or more NETBIOS parameters in the DIR.OPEN.ADAPTER command was incorrect.

  Note: This can occur if the NCB_MAX_NAMES or NCB_MAX_SESSIONS values are not less than 255, or if there is insufficient work space available to satisfy the values of NCB_STATIONS, NCB_MAX_NAMES, NCB_MAX, and NCB_MAX_SESSIONS.

Action:

- To continue without NETBIOS, do nothing. The adapter is open.
- To use NETBIOS, close the adapter, make appropriate changes, and reissue the DIR.OPEN.ADAPTER command.

Hex 11

Explanation: DIR.TIMER.SET or DIR.TIMER.CANCEL error.

If DIR.TIMER.SET:

Cause: The TIMER_VALUE is not in the 0 to 13107 range.

Action: Set a valid value and try again.

If DIR.TIMER.CANCEL:

Cause: The DIR.TIMER.SET command to be canceled was not found.

Action: None.

Hex 12

Explanation: Available work area exceeded.

Cause: Requested parameters exceeded allotted memory. Either the adapter support software work area or the work area provided by the application program is not adequate.

Action: Reduce MAX.STATION or MAX.SAP values or increase memory to the value returned in parameter DLC.WORK.LEN.ACT.

Hex 13

Explanation: Invalid LOG.ID.

Cause: The requested LOG.ID is not defined.

Action: Adjust the value accordingly.

Hex 14

Explanation: Invalid shared RAM segment or size.

Cause: The value is not an allowable value.

Action: Adjust the value accordingly.

Hex 15

Explanation: Lost log data, inadequate buffer space; log reset.

Cause: The buffer pointed to by DIR.READ.LOG or DLC.STATISTICS was too short to continue the entire log contents. The information that could not be placed in the buffer was *lost* if the command indicated "reset."

Action: The next time the command is issued, increase the size of the buffer.

Hex 16

Explanation: Requested buffer size exceeds pool length.

Cause: The buffer pool is not large enough to hold one buffer.

Action: Issue the command with either smaller buffers or larger pool.

Hex 17

Explanation: Command invalid—NETBIOS operational.

Cause: The command being issued would cause a change to NETBIOS parameters that are currently operational.

Action: To issue the command, the adapter must be closed and then re-opened, either without NETBIOS, or with NETBIOS parameters that avoid the conflict.

Hex 18

Explanation: Invalid SAP buffer length.

Cause: The specified buffer size must be at least 80 bytes and a multiple of 16.

Action: Specify the buffer size accordingly.

Hex 19

Explanation: Inadequate buffers available for request.

Cause: A request was made for more buffers than were available.

Action: Either issue the command requesting fewer buffers, or wait until more buffers become available and try again.

Hex 1A

Explanation: USER_LENGTH value too large for buffer length.

Cause: The user requested area is too large.

Action: Reduce the user space specified by the USER_LENGTH field.

Hex 1B

Explanation: The CCB_PARM_TAB pointer is invalid.

Cause: The CCB_PARM_TAB field value is either pointing into the workstation interrupt vector, area or the offset is too near the end of the segment and wrap-around will occur on some of the fields.

Action: Reissue the command with the CCB_PARM_TAB field corrected .

Hex 1C

Explanation: A pointer in the CCB parameter table is invalid.

Cause: A pointer value in the CCB parameter table is either pointing into the workstation interrupt vector area, or the offset is too near the end of the segment and wrap-around will occur on some of the fields.

Action: Reissue the command with the CCB_PARM_TAB field corrected .

Hex 1D

Explanation: Invalid CCB_ADAPTER value.

Cause: The value is outside the prescribed range.

Action: Set an acceptable value.

Hex 1E

Explanation: Invalid functional address.

Cause: The number of bits set for the functional address exceeds the number of available multicast addresses.

Action: Reduce the number of bits that are set on in the functional address.

Hex 20

Explanation: Lost data on receive; no buffers available.

Cause: There were no available buffers in the SAP's buffer pool. The frame was lost. This return code will not occur if the frame was an I frame.

Action: Free some buffers via BUFFER.FREE, and reissue the RECEIVE command. The frame was lost if it was for a link station.

Hex 21

Explanation: Lost data on receive; inadequate buffer space.

Cause: There was inadequate buffer space in the SAP's buffer pool to contain the entire frame. As much of the frame as possible was placed into receive buffers. The remainder of the message was lost. This return code will not occur if the frame was an I frame.

Action: Free some buffers with BUFFER.FREE and reissue the receive command.

Hex 22

Explanation: Error on frame transmission—check TRANSMIT_FS data.

Cause: The frame may or may not have been received by the destination adapter, as indicated by the FS byte.

Action: As appropriate for the application program.

Hex 23

Explanation: Error in frame transmit or read-back checking.

Cause: An error was detected either during the frame transmission or when the frame was read back and checked.

For CCB1, when using NDIS adapters, this code is returned if the frame is contained in more buffers than are supported by the NDIS MAC driver. Up to eight buffers are always available.

Action: As appropriate for the application program. For CCB1, when using NDIS adapters, see the maximum number of data blocks defined by the NDIS MAC driver for your adapter to determine how many buffers are supported.

Hex 24

Explanation: Unauthorized MAC frame.

Cause: Possible causes:

- The adapter is not authorized to send a MAC frame with the specified source class.
- The source class was zero.
- An attempt has been made to transmit a MAC frame via a SAP.
- An attempt has been made to transmit a MAC frame on the PC Network or an Ethernet network.

Action: Adjust the value and try again.

Hex 25

Explanation: Maximum number of commands was exceeded.

Cause: The maximum number of transmit commands that can be pending for a given station at any time (128) has been exceeded.

Action: Issue the transmit command at some later time.

Hex 26

Explanation: Unrecognized command correlator.

Cause: The command correlation sent to the adapter during ASB communications is invalid.

Action: The application program will never see this return code, since the adapter support software will assume a PC Hard Error state.

Hex 27

Explanation: Link not transmitting I frames—state changed from link opened.

Cause: This return code will be set in a transmit CCB whenever the link station leaves link-opened state because of a received frame (for instance, DISC), or because of a timeout. It will *not* be set if the link leaves link-opened state because of receipt of a CCB (for instance, DLC.CLOSE.STATION).

Action: The LINK STATION can be closed with the DLC.CLOSE.STATION command, or an attempt can be made to reestablish the connection with the DLC.CONNECT.STATION command. If the remote station is on a different ring, a different route may be required in order to reestablish the link.

Hex 28

Explanation: Invalid transmit frame length.

Cause: The frame length, as specified, is either too short to contain sufficient header information, or too long for the adapter's transmit buffer. If the transmit was for a link station, it has entered the disconnected state.

Action: Make sure that the transmit frames are no longer than the maximum transmit length, as defined by DIR.OPEN.ADAPTER.

Hex 30

Explanation: Inadequate receive buffers for adapter to open.

Cause: The requested DIR.OPEN.ADAPTER parameters have not allowed adequate receive buffer space.

Action: Reduce the buffer space by reconfiguring with either the configuration aid (OS/2) or the DIR.OPEN.ADAPTER command (DOS). Resources that may be reduced first to free larger amounts of buffer space are data hold buffers (if more than one is specified) and the number of queue elements. In addition, you can reduce the number of receive buffers if doing so does not affect the expected performance level.

If you are using the DOS NDIS protocol driver DXME0MOD.SYS, from the LAN Support Program, you should increase the work space.

Hex 32

Explanation: Invalid NODE_ADDRESS.

Cause: The defined node address is invalid.

Action: Adjust the value accordingly. Refer to *IBM Token-Ring Network Architecture Reference* for node address restrictions.

Hex 33

Explanation: Invalid adapter receive buffer length defined.

Cause: The value is either greater than the allowable maximum, less than the allowable minimum, or not a multiple of 8.

Action: Adjust the value accordingly.

Hex 34

Explanation: Invalid adapter transmit buffer length defined.

Cause: The value is either greater than the allowable maximum, less than the allowable minimum, or not a multiple of 8.

Action: Adjust the value accordingly.

Hex 35

Explanation: Unable to set group addresses.

Cause: Two causes are possible, as follows:

1. The number of group addresses exceeds the number supported by the LAN Support Program.
2. The number of group addresses exceeds the number of addresses supported by the adapter.

Action: Adjust the value accordingly. See the DIR.STATUS command for information about obtaining the list of multiple group addresses and the DIR.SET.MULT.GROUP.ADDRESS command for information about the number of group addresses supported by the LAN Support Program.

Hex 36

Explanation: Invalid group address.

Cause: Three causes are possible, as follows:

1. The I/G bit is not set in the adapter.
2. The adapter finds the address value incorrect.
3. An incorrect attempt was made to set the all-stations broadcast address.

Action: Check the I/G bit to make sure it is set on to specify that this is a group address. Verify the group address to make sure it is correct. See the DIR.STATUS command for information on how to obtain the list of multiple group addresses.

Hex 40

Explanation: Invalid STATION_ID.

Cause: Either the requested station ID does not exist, or the command code is invalid for the station type.

Action: Make the appropriate changes and reissue the command.

Hex 41

Explanation: Protocol error; link is in an invalid state for command.

Cause: The requested command cannot be accepted because of the existing primary link state of the link station. A DLC.CONNECT.STATION command will not be accepted if the link is in the disconnected or closed state. A transmit command will not be accepted if the link is in any state other than opened.

Action: According to the situation.

Hex 42

Explanation: Parameter exceeded maximum allowed.

Cause: One of the parameter values is greater than acceptable.

Action: Use an acceptable value.

Hex 43

Explanation: Invalid SAP_VALUE or value already in use.

Cause: For a DLC.OPEN.SAP command, this return code indicates that the SAP_VALUE has already been used or the specified SAP is the null or global SAP.

For a DLC.OPEN.STATION command, this return code indicates that this SAP already has a link to the specified RSAP_VALUE and DESTINATION_ADDR combination, or that the remote SAP specified was the null SAP, global SAP, or a group SAP.

Action: Use an acceptable value. (Do not use X'00'.)

Hex 44

Explanation: Invalid routing field length.

Cause: The indicated routing field is either too short, greater than 18 bytes long, or is an odd number of bytes long.

Action: Set the length field to a correct value.

Hex 45

Explanation: Requested group membership in non-existent group SAP.

Cause: Membership has been requested in a group SAP that is not open.

Note: The command has been completed up to the point at which the adapter encountered the error. Other parameters have been changed if the command was DLC.MODIFY. However, the SAP has not been added as a member to a group SAP.

Action: Either change the group SAP value to a group SAP that has been opened or open the group SAP before requesting its membership.

Hex 46

Explanation: Inadequate number of link stations.

Cause:

- DLC.OPEN.SAP: There are inadequate link stations or SAPs available to satisfy the open.
- DLC.OPEN.STATION: All link stations assigned to this SAP are in use.

Action:

- DLC.OPEN.SAP: Close other SAPs, reduce the number of link stations being requested for the SAP, or wait for these resources to be freed.

- DLC.OPEN.STATION: Close other link stations for the SAP, reallocate some link stations using the DLC.REALLOCATE command, close the SAP and reserve additional link stations, or wait for these resources to be freed.

Hex 47

Explanation: The SAP cannot close unless all link stations are closed.

Cause: At least one link station is open for this SAP.

Action: Close all link stations and try again.

Note: If an error code 47 results when a DLC.CLOSE.SAP command closely follows a DLC.CLOSE.STATION command for the last open station for that SAP, reissue the DLC.CLOSE.SAP command.

Hex 48

Explanation: Group SAP cannot close—all member SAPs are not closed.

Cause: At least one individual member SAP of this group SAP is open.

Action: Delete all SAPs in the group using the DLC.MODIFY command and try again.

Hex 49

Explanation: The group SAP has reached maximum membership.

Cause: See Explanation.

Note: The command has completed the operation up to the point at which the adapter encountered the error. other parameters have been changed if the command was DLC.MODIFY.

Action: According to the application program.

Hex 4A

Explanation: Sequence error; incompatible command in progress.

Cause: The station is in the process of closing or establishing a connection.

Action: Await completion or issue a DLC.RESET command.

Hex 4B

Explanation: Station closed without remote acknowledgment.

Cause: The adapter issued a DISC command to the remote station as a result of receiving a DLC.CLOSE.STATION SRB. No acknowledgment has been received from the remote adapter, and the link station has been closed.

Action: According to the application program.

Hex 4C

Explanation: Sequence error; cannot close while commands are pending.

Cause: Commands are in process. This prevents closing the SAP or link station.

Action: Wait until all pending commands are complete, or issue a reset.

Hex 4D

Explanation: Unsuccessful link station connection attempt.

Cause: The DLC.CONNECT.STATION command could not establish a requested connection.

Action: Determine the cause for the failure (for example, verify RSAP values, routing information, MAC address, and connection between the two workstations) and try again when the problem is resolved.

Hex 4E

Explanation: Member SAP not found in group SAP list.

Note: The command has completed the operation up to the point at which the adapter encountered the error. Other parameters have been changed if the command was DLC.MODIFY.

Cause: A request was issued to delete an individual member SAP from a group SAP. The SAP was not found to be assigned to the group.

Action: Verify the SAP value.

Hex 4F

Explanation: Invalid remote address; cannot be a group address.

Cause: The remote address parameter has the high bit of the high byte set to 1, which indicates a group address. A group address is not allowed to be specified for this command.

Action: Correct the remote address, and reissue the command.

Hex 50

Explanation: Invalid pointer in the CCB_POINTER field.

Cause: See Explanation.

Action: Check to ensure that all 4-byte pointers in the CCB's pointer field are accessible to the OS/2 process issuing the command.

Hex 52

Explanation: Invalid application program ID.

Cause: See Explanation.

Action: Use the CCB_APPL_ID field returned on the DIR.OPEN.ADAPTER command.

Hex 53

Explanation: Invalid application program key code.

Cause: See Explanation.

Action: Use the CCB_APPL_KEY field provided on the DIR.OPEN.ADAPTER command.

Hex 54

Explanation: Invalid System Key code.

Cause: See Explanation.

Action: Use the System Key code as defined by the configuration parameters.

Hex 55

Explanation: Buffer is smaller than buffer size given on the DIR.OPEN.SAP command.

Cause: See Explanation.

Action: Increase the size of the buffer being used to at least that of the buffers in the SAP buffer pool.

Hex 56

Explanation: The adapter's system process is not installed.

Cause: See Explanation.

Action: Load the adapter support software's system process. Insert the DETACH command into the STARTUP.BAT file and specify the system process load module name.

Hex 57

Explanation: Not enough stations are available.

Cause: Command has completed the operation; however, only a portion of the stations requested have been reserved. No more stations are available for reservation.

Action: None.

Hex 58

Explanation: Invalid CCB_PARAMETER_1.

Cause: See Explanation.

Action: Adjust the parameter defined in the CCB_PARAMETER_1 field, and reissue the command.

Hex 59

Explanation: Inadequate number of queue elements to satisfy request.

Cause: See Explanation.

Action: On a short-term basis, wait for other requests to complete. Otherwise, increase queue elements using the configuration aid.

Hex 5A

Explanation: Initialization failure; cannot open the adapter.

Cause: See Explanation.

Action: Check Bring-up error code for details of initialization failure.

Hex 5B

Explanation: Error detected in chained READ command.

Cause: Bad return code given on a chained READ command.

Action: Correct READ command problem, and reissue the command.

Hex 5C

Explanation: Direct stations are not assigned to this application program.

Cause: The invoking application program must request ownership of direct stations before issuing any requests involving the direct stations.

Action: Issue a DIR.OPEN.DIRECT command to gain ownership of the direct stations.

Hex 5D

Explanation: The DD interface is not installed.

Cause: See Explanation.

Action: Insert the DEVICE command into the CONFIG.SYS file and specify the device driver load module name.

Hex 5E

Explanation: The requested adapter is not installed.

Cause: See Explanation.

Action: Install the adapter into the system. Check that the adapter is configured correctly for the primary or alternate adapter.

Hex 5F

Explanation: Chained commands must all be for the same adapter.

Cause: See Explanation.

Action: Only chain multiple command requests that are for the same adapter. Do not mix primary and alternate adapter requests.

Hex 60

Explanation: Adapter initializing; command not accepted.

Cause: See Explanation.

Action: Reissue the DIR.OPEN.ADAPTER command until the command is completed successfully.

Hex 61

Explanation: The number of allowed application programs has been exceeded.

Cause: See Explanation.

Action: On a short-term basis, terminate one of the application programs using the adapter support software. Otherwise, adjust the configuration parameter MAX_USERS.

Hex 62

Explanation: Command canceled; system action.

Cause: The system administrator has issued a command using the System Key.

Action: Re-establish session with the adapter support software if necessary.

Hex 63

Explanation: Direct stations are not available.

Cause: Direct stations have already been assigned.

Action: The application program owning the direct stations must issue the DIR.OPEN.DIRECT command to relinquish ownership of direct stations in order for the command to be completed successfully.

Hex 64

Explanation: Invalid DDNAME parameter.

Cause: Either the device driver name given was invalid or the device driver did not provide the proper interdevice driver communication information in its device driver header.

Action: Make sure the device driver is loaded and has a valid header.

Hex 65

Explanation: Not enough GDT selectors to satisfy request.

Cause: The number of GDT selectors defined during configuration is not adequate to support the number of control blocks and buffers passed to the adapter support software.

Action: Increase the number of GDT selectors when configuring the adapter support software using the configuration aid.

Hex 67

Explanation: Command canceled; CCB resources purged.

Cause: The command has been canceled as a result of the PURGE.RESOURCES command.

Action: None.

Hex 68

Explanation: The application program ID is not valid.

Cause: The application program ID provided was not obtained from the interface used for the command request.

Action: The application program ID can only be used at the interface where it was obtained. Use the other interface (for example, DLR interface or DD interface).

Hex 69

Explanation: The segment associated with the request cannot be locked.

Cause: Too many processes are running concurrently and the system has run out of resources. The segment cannot be locked.

Action: Reduce the number of OS/2 processes (the overall number of different segments controlled by the adapter support software), or wait until the resources are available. Memory references passed to the application program that contain control blocks accessed when the adapter support software processes an adapter interrupt (for example, CCBs and buffers in the SAP buffer pool) are locked so they are not moved or swapped by OS/2. By canceling processes using the application program or reducing the number of control blocks passed to the application program, more memory will be available for other requests that require their control blocks locked. Adding memory to your system may also alleviate this problem.

Hex FF

Explanation: Command in process.

Cause: See Explanation.

Action: None.

DLC Status Codes

For CCB1: Certain conditions that arise in the DLC function of the adapter (for Token-Ring Network) or the DLC function of the adapter support software in the workstation (for PC Network) are reported to a DLC status appendage. DLC status codes are presented to the appendage in the AX register. The CX register contains the adapter number (0 or 1). Register SI contains a user value as defined in the USER_STAT_VALUE parameter of the DLC.OPEN.SAP command. Registers ES and BX point to the DLC status table that contains additional data for certain status codes. See "Appendages" on page 2-4 for more information about providing an appendage to use these codes.

For CCB2: Certain conditions that occur in the DLC function are returned in the READ command for DLC status change events. The format of the DLC status table in the READ command's CCB is shown in Table B-2 on page B-29. This table starts at offset 10 in the READ command's parameter table.

For CCB3: Certain conditions that arise in the DLC function of the adapter (for Token-Ring Network) or the DLC function of the adapter support software in the workstation (for PC Network) are reported to a DLC status appendage. DLC status codes are presented to the appendage in the AX register. the CX register contains the adapter number (0 or 1). Register SI contains a user value as defined in the USER_STAT_VALUE parameter of the

DLC.OPEN.SAP command. Registers ES and BX point to the DLC status table that contains additional data for certain status codes. See "Appendages" on page 2-4 for more information about providing an appendage to use these codes.

The DI register contains the offset of the DLC appendage. The DS register contains the application program's device driver protect mode data segment. An invocation code of X'0001' has been pushed onto the stack. Before returning to the adapter support software, the application program must remove the invocation code from the stack.

DLC Status Table

Table B-2. DLC Status Table

Offset	Parameter Name	Byte Length	8086 Type	Description
0	STATION_ID	2	DW	Station ID.
2	DLC_STATUS_CODE	2	DW	DLC status code.
4	FRMR_DATA	5	DB	Five bytes of reason code that are applicable when an FRMR is either transmitted or received.
9	ACCESS_PRIORITY	1	DB	The new access priority that is applicable when status bit 5 is on. The format is B'nnn00000' where "nnn" is the access priority. This byte is not used and is set to zero when used on the PC Network or on Ethernet.
10	REMOTE_NODE	6	DB	The 6-byte remote node value of a newly opened link station. Applicable when status bit 10 is on.
16	REMOTE_SAP	1	DB	The 1-byte remote SAP value of a newly opened link station. Applicable when status bit 10 is on.
For CCB2 and CCB3:				
17		1	DB	Reserved.
18	USER_STAT_VALUE	2	DW	User value as defined in the DLC.OPEN.SAP command.

DLC Status Codes

Table B-3. DLC Status Codes

Bit	Function	Additional Data
15	Link lost	
14	DM or DISC received, or DISC acknowledged	
13	FRMR received	Five bytes of reason code data are contained in the area pointed to by the address in the ES:BX register (FRMR_DATA).
12	FRMR sent	Five bytes of reason code data are contained in the area pointed to by the address in the ES:BX register (FRMR_DATA).
11	SABME received for an open link station	
10	SABME received, link station opened	A new link station has been opened by the adapter. A DLC.OPEN.STATION command should NOT be issued for that link station. A DLC.CONNECT.STATION command must be issued to accept the SABME or a DLC.CLOSE.STATION command to reject it. The local STATION_ID, remote node address (DEST_ADDR), and remote SAP value (RSAP_VALUE) are provided in the status table pointed to by the address in the ES and BX registers.
9	Remote station has entered a "local busy" condition	
8	Remote station has left a "local busy" condition	
7	Ti has expired	
6	DLC counter overflow	One or more of the DLC link station's DLC log counters has reached one-half the maximum value. A DLC.STATISTICS command should be issued.
5	Access Priority lowered	The new access priority (ACCESS_PRIORITY) is in the area pointed to by the address in the ES:BX register. This bit is not set on the PC network or Ethernet.
4-1	Reserved	
0	Local station has entered a "local busy" condition	This code is reported only when the state has changed because of a buffer pool (out-of-buffers) condition when the adapter support software cannot accept an I frame. It is not reported because of a DLC.FLOW.CONTROL command being issued by the network application program. It is the responsibility of the application program to issue a "flow control on" command to reset the "local busy" condition.

Suggested Actions in Response to DLC Status

Link Lost

It appears that the connection to the remote partner has been lost, or that the remote station has been closed. A DLC.CLOSE.STATION command can be issued to free the control block, or a DLC.CONNECT.STATION command (possibly with different routing information) can be issued to attempt to reestablish the connection. Any pending transmit commands will be returned with a CCB_RETCODE of X'27'.

DM or DISC received

The remote partner is attempting to terminate the connection. A DLC.CLOSE.STATION command should be issued. Any pending transmit commands will be returned with a CCB_RETCODE of X'27'.

FRMR Received

The remote partner has detected a DLC protocol error in the frame received from this station. Either a DLC.CLOSE.STATION or DLC.CONNECT.STATION command should be issued. Any pending transmit commands will be returned with a CCB_RETCODE of X'27'.

FRMR Sent

The local link station has detected a DLC protocol error in a frame received from the remote partner. However, if a Ti Timer-expired DLC status interrupt is received after receipt of this interrupt, a DLC.CLOSE.STATION or DLC.CONNECT.STATION command should be issued to the local station. Any pending transmit commands will be returned with a CCB_RETCODE of X'27'.

SABME Received for an Open Link Station

The remote station wants to reset an existing connection. A DLC.CONNECT.STATION command can be issued to reestablish the connection, or a DLC.CLOSE.STATION command can be issued to terminate it. Any pending transmit commands will be returned with a CCB_RETCODE of X'27'.

SABME Received, Link Station Opened

A control block has been allocated and a station has been opened, in disconnected state, in response to a SABME received from a remote station. The connection request can be accepted by issuing a DLC.CONNECT.STATION command, or rejected by issuing a DLC.CLOSE.STATION command.

Remote Station Has Entered Local Busy

The remote station has temporarily stopped receiving I frames, probably because of buffer congestion. The local station will stop sending I frames. The application program can choose to issue transmit commands for the affected station, up to the maximum number accepted by the adapter, but they will be queued until the remote station leaves the local busy state.

Remote Station Has Left Local Busy

The local station will resume I-frame transmission.

Ti Timer Expired

This status is not returned while the link is in link opened state. In other states it is returned to indicate that there is no activity on the link and that the workstation may, therefore, want to close the link to free the control block.

DLC Counter Overflow
One or more of the error counters maintained for the link station has reached half of its maximum value. The counter will wrap back to zero when it reaches its maximum value. The application program should issue a DLC.STATISTICS command to read and reset the counters.

Access Priority Reduced
The access priority requested for this SAP or link station was greater than that authorized for the adapter, and the access priority has been reduced. The new priority is in the Adapter Status Table, or if the adapter is being operated without the adapter support software, in ARB byte 13. There is no workstation application program action required as this is for information only. However, a DLC.MODIFY command can be issued to change the access priority. Access priority is not set on the PC Network or on Ethernet. Token-ring network NDIS adapters may not use it either. When using a token-ring NDIS adapter, consult the adapter documentation to determine whether the access priority is set.

NCB Return Codes

Table B-4 (Page 1 of 2). NCB Return Codes

Code	Meaning
00	Good return.
01	Illegal buffer length.
03	Invalid command.
05	Command timed out.
06	Message incomplete.
07	Data for one or more SEND type NO.ACK commands was not received.
08	Illegal local session number.
09	No resource available.
0A	Session closed.
0B	Command canceled.
0D	Duplicate name in local name table.
0E	Name table full.
0F	Command completed—name has active session and is now deregistered.
11	Local session table full.
12	Session open rejected.
13	Illegal name number.
14	Cannot find name called.
15	Name not found or cannot specify "*" or null.
16	Name in use on remote NETBIOS.
17	Name deleted.
18	Session ended abnormally.
19	Name conflict detected.

Table B-4 (Page 2 of 2). NCB Return Codes

Code	Meaning
21	Interface busy.
22	Too many commands pending.
23	Invalid number in NCB_LANA_NUM field.
24	Command completed while cancel occurring.
26	Command not valid to cancel.
30	Name defined by another environment.
34	Environment not defined, RESET must be issued.
35	Required operating system resources exhausted, retry later.
36	Maximum application programs exceeded.
37	No SAPs available for NETBIOS.
38	Requested resources not available.
39	Invalid NCB address or length does not fit in segment.
3A	RESET cannot be issued from a NETBIOS adapter appendage.
3B	Invalid NCB_DD_ID value.
3C	NETBIOS attempted to lock user storage and the lock failed.
3F	NETBIOS Device Driver open error.
40	OS/2 error detected.
4E	Network status—one or more of bits 12, 14, 15 on for longer than 60 seconds.
4F	Network status—one or more of bits 8-11 on.
F6	Unexpected error on CCB completion.
F7	Error on implicit DIR.INITIALIZE.
F8	Error on implicit DIR.OPEN.ADAPTER.
F9	Adapter protocol driver internal error.
FA	Adapter check.
FB	NETBIOS program not loaded in PC.
FC	DIR.OPEN.ADAPTER or DLC.OPEN.SAP failed; check parameters.
FD	Unexpected adapter close.
FE	NETBIOS not operational and application program explicitly opened the adapter.

NCB Return Codes Listed by Command

	Return Codes in Hex																											
NCB Commands	**00**	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0A**	**0B**	**0C**	**0D**	**0E**	**0F**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**21**	**22**
NCB.ADD.GROUP.NAME	x			x										x	x							x	x			x	x	x
NCB.ADD.NAME	x			x										x	x							x	x			x	x	x
NCB.CALL	x			x		x				x		x						x	x		x	x			x	x	x	x
NCB.CANCEL	x	x		x																							x	
NCB.CHAIN.SEND	x	x		x		x		x	x		x	x													x		x	x
NCB.CHAIN.SEND.NO.ACK	x	x		x				x	x		x	x													x		x	x
NCB.DELETE.NAME	x			x												x						x					x	x
NCB.FIND.NAME	x	x		x		x																				x	x	x
NCB.HANG.UP	x			x		x			x		x	x													x		x	x
NCB.LAN.STATUS.ALERT	x			x								x															x	x
NCB.LISTEN	x			x						x		x						x				x		x	x	x	x	x
NCB.RECEIVE	x	x		x		x	x	x	x		x	x													x		x	x
NCB.RECEIVE.ANY	x	x		x			x	x			x	x								x				x	x	x	x	x
NCB.RECEIVE.BROADCAST.DATAGRAM	x	x		x			x					x								x				x		x	x	x
NCB.RECEIVE.DATAGRAM	x	x		x			x					x								x				x	x	x	x	x
NCB.RESET	x	x		x																							x	
NCB.SEND	x	x		x		x		x	x		x	x													x		x	x
NCB.SEND.BROADCAST.DATAGRAM	x	x		x																x						x	x	x
NCB.SEND.DATAGRAM	x	x		x																x						x	x	x
NCB.SEND.NO.ACK	x	x		x				x	x		x	x													x		x	x
NCB.SESSION.STATUS	x	x		x			x															x				x	x	x
NCB.STATUS	x	x		x		x	x					x														x	x	x
NCB.TRACE	x	x		x										x						x								
NCB.UNLINK	x			x																							x	
NCB Commands	**00**	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0A**	**0B**	**0C**	**0D**	**0E**	**0F**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**21**	**22**
	Return Codes in Hex																											

	Return Codes in Hex																
NCB Commands	**23**	**24**	**26**	**30**	**34**	**35**	**36**	**37**	**38**	**39**	**3A**	**3B**	**3C**	**3F**	**40**	**4x**	**Fx**
NCB.ADD.GROUP.NAME	x				x	x				x		x	x		x	x	x
NCB.ADD.NAME	x			x	x	x				x		x	x		x	x	x
NCB.CALL	x			x	x	x				x		x	x		x	x	x
NCB.CANCEL	x	x	x		x	x				x		x	x		x	x	x
NCB.CHAIN.SEND	x			x	x	x				x		x	x		x	x	x
NCB.CHAIN.SEND.NO.ACK	x			x	x	x				x		x	x		x	x	x
NCB.DELETE.NAME	x				x	x				x		x	x		x	x	x
NCB.FIND.NAME	x				x	x				x		x	x		x	x	x
NCB.HANG.UP	x				x	x				x		x	x		x	x	x
NCB.LAN.STATUS.ALERT	x				x	x				x		x	x		x	x	x
NCB.LISTEN	x			x	x	x				x		x	x		x	x	x
NCB.RECEIVE	x				x	x				x		x	x		x	x	x
NCB.RECEIVE.ANY	x				x	x				x		x	x		x	x	x
NCB.RECEIVE.BROADCAST.DATAGRAM	x			x	x	x				x		x	x		x	x	x
NCB.RECEIVE.DATAGRAM	x			x	x	x				x		x	x		x	x	x
NCB.RESET	x						x		x			x	x		x	x	x
NCB.SEND	x				x	x				x		x	x		x	x	x
NCB.SEND.BROADCAST.DATAGRAM	x			x	x	x				x		x	x		x	x	x
NCB.SEND.DATAGRAM	x			x	x	x				x		x	x		x	x	x
NCB.SEND.NO.ACK	x			x	x	x				x		x	x		x	x	x
NCB.SESSION.STATUS	x			x	x	x				x		x	x		x	x	x
NCB.STATUS	x			x	x	x				x		x	x		x	x	x
NCB.TRACE	x																x
NCB.UNLINK	x											x	x			x	x
NCB Commands	**23**	**24**	**26**	**30**	**34**	**35**	**36**	**37**	**38**	**39**	**3A**	**3B**	**3C**	**3F**	**40**	**4x**	**Fx**
	Return Codes in Hex																

NCB Return Codes Cause and Action

Hex 00

Explanation: Operation was completed successfully.

Hex 01

Explanation: Illegal buffer length.

Cause: The requested buffer length (or invalid buffer selector if using NETBIOS 3.0) is illegal.

Action: Specify the correct size for the buffer and retry.

Hex 03

Explanation: Invalid command.

Cause: See Explanation.

Action: Issue the correct command.

Hex 05

Explanation: Command timed out.

Cause: See Explanation.

Action: Reissue the same command or another command. If a send command timed out, there cannot be a receive pending from the other name.

Hex 06

Explanation: Message incomplete.

Cause: The application program received part of a message because the specified buffer length is not large enough to receive the full message.

Action:

- For NCB.RECEIVE and NCB.RECEIVE.ANY: Issue another receive to get the rest of the message before the remote side times out.
- For NCB.STATUS, NCB.SESSION.STATUS, NCB.RECEIVE.DATAGRAM, and NCB.RECEIVE.BROADCAST.DATAGRAM: The remaining data is lost.

  Note: If the command was NCB.STATUS, this error code could occur because the remote side could not transmit the entire status update if the data was longer than the maximum length UI frame that can be transmitted.

Hex 07

Explanation: Data for one or more SEND type NO.ACK commands was not received.

Cause: Data sent by a previous NCB.SEND.NO.ACK or NCB.CHAIN.SEND.NO.ACK command was either not received at all or only partially received by the remote application program.

Action: The application program will need to initiate any data recovery needed.

Hex 08

Explanation: Illegal local session number.

Cause: The session number specified is not one of the active sessions.

Action: Reissue the command with the correct active session number.

Hex 09

Explanation: No resource available.

Cause: You are trying to establish a session with a remote application program that has no more room in the session table.

Action: Reissue the command at a later time.

Hex 0A

Explanation: Session closed.

Cause: The name from the transmitting side closed the session. The session has terminated normally.

Action: None.

Hex 0B

Explanation: Command canceled.

Cause: See Explanation.

Action: None.

Hex 0D

Explanation: Duplicate name in local name table.

Cause: You tried to specify a name that is currently in the name table.

Action: Reissue the command and specify another name.

Hex 0E

Explanation: Name table full.

Cause: The number of names defined has exceeded the number defined at initialization (default=17).

Action: Wait until a delete name is issued so an entry will become available.

Hex 0F

Explanation: Command was completed; name has active session and is now deregistered.

Cause: The name to be deleted is active in a session now, but is deregistered. When the name is marked deregistered and has active sessions, it still occupies a slot in the table. The name is unusable for any new sessions.

Action: Close all the sessions using this name.

Hex 11

Explanation: Local session table full.

Cause: There are no available entries on the session table. (The number of sessions is user specified in NCB.RESET.)

Action: Wait until a session has closed so an entry will become available.

Hex 12

Explanation: Session open rejected.

Cause: No LISTEN command is pending on the remote NETBIOS.

Action: Wait until a LISTEN is issued on the remote NETBIOS.

Hex 13

Explanation: Illegal name number.

Cause: The number of the name has been changed or was never specified.

Action: Use the most recent number that was assigned to the name.

Hex 14

Explanation: Cannot find name called or no answer.

Cause: No response to the NCB.CALL command was received.

Action: Try again later.

Hex 15

Explanation: Name not found or cannot specify "*" or null.

Cause: The name specified is not in the table, or the first character of the name is either an ASCII asterisk or "00."

Action: Try again with another name that has been verified to be correct.

Hex 16

Explanation: Name in use on remote NETBIOS.

Cause: Name found in another table. Names used in the network are unique and can be used only in one place. The name is already defined on another node.

Action: Either specify another name or have the name changed at the remote end.

Hex 17

Explanation: Name deleted.

Cause: See Explanation.

Action: Add the name to the table and reissue the command.

Hex 18

Explanation: Session ended abnormally.

Cause: The most probable cause is that a send-type NCB timed out because no receive command was available in the remote node.

Action:

- If a send timed out, reestablish the session and ensure that the remote node has issued a receive.
- If the session cannot be reestablished, maintenance procedures should be initiated for the node in question.

Hex 19

Explanation: Name conflict detected.

Cause: Network protocol has detected two or more identical names on the network.

Action: Identical names on the network should be removed.

Hex 21

Explanation: Interface busy.

Cause: NETBIOS is either busy or out of local resources.

Note: This condition can also be caused by any of the network status bits 12, 14, or 15 being on.

Action: Try again later.

Hex 22

Explanation: Too many commands pending.

Cause: See Explanation.

Action: Try again later.

Hex 23

Explanation: Invalid number in NCB_LANA_NUM field.

Cause: You tried to specify a value other than "00" or "01," or the adapter is not present.

Action: Verify that the adapter is present, or correct the value and try the command again. Use "00" for the primary adapter and "01" for the alternate.

Hex 24

Explanation: Command completed while cancel occurring.

Cause: You tried to cancel a command that had already been completed.

Action: None.

Hex 26

Explanation: Command not valid to cancel.

Cause: Tried to cancel a command that is invalid to cancel.

Action: Verify the correctness of the cancel command.

Hex 30

Explanation: Name defined by another environment.

Cause: Another environment has already defined the name.

Action: Choose another name.

Hex 34

Explanation: Environment not defined; RESET must be issued.

Cause: See Explanation.

Action: Issue RESET.

Hex 35

Explanation: Required operating system resources exhausted; retry later.

Cause: See Explanation.

Action: Retry command later.

Hex 36

Explanation: Maximum number of application programs was exceeded.

Cause: The maximum number of application programs defined at NETBIOS 3.0 load-time are executing.

Action: Wait until another application program ends.

Hex 37

Explanation: No SAPs available for NETBIOS.

Cause: The adapter has no SAPs available for NETBIOS. SAPs relinquishes use of a SAP.

Note: Return code not currently used by NETBIOS.

Hex 38

Explanation: Requested resources not available.

Cause: See Explanation.

Action: Operate with a smaller number of resources or end the operation.

Hex 39

Explanation: Invalid NCB address or length does not fit in segment.

Cause: See Explanation.

Action: Application program error. Correct NCB address and selector length.

Note: In the case of this return code, since the NCB is in doubt, the value is returned only in register AL. No attempt is made to place the return code into the NCB.

Hex 3A

Explanation: RESET cannot be issued from a NETBIOS adapter appendage.

Cause: The RESET command was issued when the NETBIOS 3.0 adapter was processing a hardware interrupt.

Action: Application program error. Do not issue RESET in this situation.

Notes:

1. Return code applies only to the DD interface.
2. Return code not currently used by NETBIOS.

Hex 3B

Explanation: Invalid NCB_DD_ID value.

Cause: The value in NCB_DD_ID is not identical to the value returned by NETBIOS 3.0 in the first RESET issued by the device driver application program. Note that NCB_DD_ID must be X'0000' in the first RESET issued for a given device driver application program.

Action: Application program error. Correct NCB_DD_ID value.

Note: Return code applies only to the DD interface.

Hex 3C

Explanation: NETBIOS attempted to lock user storage and the lock failed.

Cause: See Explanation.

Action: Try the command at a later time.

Hex 3F

Explanation: NETBIOS device driver open error.

Cause: Either the device driver had an actual problem in its open process, or the NETBIOS device driver was not loaded.

Action: Load the appropriate code before executing NETBIOS application programs.

Note: Return code not currently used by NETBIOS.

Hex 40

Explanation: OS/2 error detected.

Cause: During processing, an unexpected error was indicated by OS/2.

Hex 4E

Explanation: Network status; one or more of bits 12, 14, or 15 are on for more than 60 seconds.

Cause: See Explanation.

Action: Check the extended status last network status code. The only NETBIOS command that can be issued is NCB.RESET.

Note: This return code is not reported at all if some status bits (8-11) are also on. This return code is reported to the application program only if the status bits 12, 14, or 15 remain on longer than 60 seconds.

See "Token-Ring Network Status Codes for All CCBs" on page B-55 for a description of the status bits.

Hex 4F

Explanation: Network status; one or more of bits 8 - 11 on.

Cause: See Explanation.

Action: Check the extended status last network status code. The only NETBIOS command that can be issued is NCB.RESET.

See "Token-Ring Network Status Codes for All CCBs" on page B-55 for a description of the status bits.

Hex F6

Explanation: Unexpected error on CCB completion.

Cause: This is a NETBIOS 2.X return code that indicates that a CCB has completed with an unexpected bad return code. NETBIOS 1.X returned a X'FA' in these situations.

Action: The only NETBIOS command that can be issued is NCB.RESET.

Hex F7

Explanation: Error on implicit DIR.INITIALIZE.

Cause: See Explanation.

Action: Check the extended status bring-up error code. The only NETBIOS command that can be issued is NCB.RESET.

Note: Return code for DOS NETBIOS only.

Hex F8

Explanation: Error on implicit DIR.OPEN.ADAPTER.

Cause: See Explanation.

Action: Check the extended status bring-up error code. The only NETBIOS command that can be issued is NCB.RESET.

Notes:

1. There is a possibility that a DIR.OPEN.ADAPTER could fail because of a temporary timing condition. Because of this, before reporting this return code, the

2. This error could be caused by an attempt to open on a Token-Ring Network with the adapter set to the wrong data rate. Check the data rate setting.

Hex F9

Cause: See Explanation.

Action: Check the PC System detected error code. The only NETBIOS command that can be issued is NCB.RESET.

Hex FA

Explanation: Adapter check.

Cause: See Explanation.

Action: Check the adapter check reason code. The only NETBIOS command that can be issued is NCB.RESET.

Hex FB

Explanation: NETBIOS code not loaded in the workstation.

Cause: NETBIOS is not loaded or is loaded and not operational due to an error at load time, greater than X'03' in the first field.

Action: Load and start NETBIOS, or correct the conditions that cause a load error. Then reissue the command or

Note: Return code for DOS NETBIOS only.

Hex FC

Explanation: DIR.OPEN.ADAPTER or DLC.OPEN.SAP failed; check parameters.

Cause: See Explanation.

Action: Correct the parameters in error and execute the DIR.OPEN.ADAPTER command again. Note that the DLC.OPEN.SAP command is executed on initial start and restart of NETBIOS. The parameters used are obtained from the DIR.OPEN.ADAPTER command (executed either explicitly or implicitly).

Hex FD

Explanation: Unexpected adapter close.

Cause: The adapter was closed while NETBIOS was executing.

Action: Issue a NCB.RESET command.

Hex FE

Explanation: NETBIOS is not operational and the application program explicitly opened the adapter.

Cause: The adapter has been explicitly opened by the application program and NETBIOS is not operational.

Action: Close the adapter and reissue the NETBIOS command.

See "Token-Ring Network Status Codes for All CCBs" on page B-55 for a description of the status bits.

Notes:

For the following codes X'F7' to X'FD':

1. The condition to be reported through NCB_RETCODE is the last to have occurred.
2. Extended status information, with the exception of adapter counters, is available in the NCB_RESERVE field of the command block. In the case of the NCB.RESET command, it is the status before the NCB.RESET.
3. Network status information:
   - Any network status bits 8-11 set on cause error code X'4F'.
   - Any network status bits 12, 14, or 15 set on for longer than 60 seconds cause error code X'4E'. Code X'4F' has priority over code X'4E'.
   - Network status bits 6 and 7 do not cause errors. If bit 7 (counter overflow) is on, nothing is reported. If no network status appendage is defined, the local NETBIOS counters will be updated with the DIR.READ.LOG command. Bit 6 (single station) is ignored.
   - Return code for DOS NETBIOS only.

Adapter Status Parameter Table

This information is placed in workstation memory by the adapter support software in response to a DIR.STATUS command. The adapter support software places a pointer address in the ADAPTER_PARMS_ADDR field of the DIR.STATUS command's parameter table.

Note: For an NDIS adapter, the Adapter Status Parameter Table is updated when a RING STATUS occurs and at certain intervals based on a timer.

Table B-5 (Page 1 of 2). Token-Ring Network Adapter Status Parameter Table

Offset	Parameter Name	Byte Length	8086 Type	Description
0	PHYS_ADDR	4	DB	Adapter physical address
4	UP_NODE_ADDR	6	DB	The upstream node address
10	UP_PHYS_ADDR	4	DB	The upstream physical address
14	POLL_ADDR	6	DB	Last poll address
20	AUTH_ENV	2	DB	Authorized environment
22	ACC_PRIORITY	2	DB	Transmit access priority
24	SOURCE_CLASS	2	DB	Source class authorization
26	ATT_CODE	2	DB	Last attention code
28	SOURCE_ADDR	6	DB	Last source address
34	BEACON_TYPE	2	DB	Last beacon type
36	MAJOR_VECTOR	2	DB	Last major vector

* These fields are valid only when the ring is beaconing.

Table B-5 (Page 2 of 2). Token-Ring Network Adapter Status Parameter Table

Offset	Parameter Name	Byte Length	8086 Type	Description
38	NETW_STATUS	2	DB	Network status
40	SOFT_ERROR	2	DB	Soft error timer value
42	FE_ERROR	2	DB	Front end error counter
44	LOCAL_RING	2	DB	Ring number
46	MON_ERROR	2	DB	Monitor error code
48	BEACON_TRANSMIT	2	DB	Beacon transmit type
50	BEACON_RECEIVE	2	DB	Beacon receive type
52	FRAME_CORREL	2	DB	Frame correlation save
54	BEACON_NAUN *	6	DB	Beaconing station NAUN
60		4	DB	Reserved
64	BEACON_PHYS *	4	DB	Beaconing station physical address

* These fields are valid only when the ring is beaconing.

Table B-6. PC Network and Ethernet Adapter Status Parameter Table

Offset	Parameter Name	Byte Length	8086 Type	Description
0		28	DB	Reserved
28	SOURCE_ADDR	6	DB	Last source address
34		2	DB	Reserved
36	MAJOR_VECTOR	2	DB	Last major vector
38	NETWORK_STATUS	2	DW	Network status
40	REPORT_ERROR	2	DW	Error report timer value
42	REPORT_ERROR_TICK	2	DW	Error report timer tick counter
44	LOCAL_BUS_NUMBER	2	DB	Local bus number
46		6	DB	Reserved
52	FRAME_CORRELATOR	2	DB	Frame correlation save
54		6	DB	Reserved
60	NETWORK_SAMPLES	2	DW	Number of network utilization samples
62	NETWORK_BUSY	2	DW	Number of network busy samples
64		4	DB	Reserved

Frame Status

The frame status (FS) byte is returned to the application program for some commands on the Token-Ring Network only.

Note: This FS byte is also referred to as the TRANSMIT_PCFE field for transmit commands.

Some values and their meanings are:

X'CC'	The frame was copied.
X'00'	The address was not received and the frame was not copied (the destination adapter must not be on the ring).
X'88'	The destination adapter recognized the frame, but did not copy it (possibly due to being overloaded).

Refer to *IBM Token-Ring Network Architecture Reference* for more about the FS byte.

Exception Indications

The exception indications include:

- Adapter Check
- Network Status
- Bring-Up and Open Errors
- PC System Detected Errors
- System Action Exceptions for OS/2.

Adapter Check

The following sections document adapter checks for CCB1, CCB2, and CCB3.

Adapter Check for CCB1

When an adapter check occurs, the adapter support software closes the adapter and all ring communication ceases. The adapter support software assumes the adapter has encountered an unrecoverable error. An adapter check appendage (ADAPTER_CHECK_EXIT), if defined by these commands (DIR.INITIALIZE, DIR.OPEN.ADAPTER, DIR.MODIFY.OPEN.PARMS, DIR.SET.USER.APPENDAGE), will be taken. On entry, the CX register will contain the adapter number, the AX will contain the adapter check reason code, and the ES and BX registers will point to Table B-7. While interrogating the information, the application program should either move the data to private memory or keep all interrupts masked off.

Table B-7 (Page 1 of 2). Adapter Check for CCB1

Offset	Parameter Name	Byte Length	8086 Type	Description
0	NOTIFICATION_FLAG	4	DD	A pointer to the first of a queue of commands that were pending when the adapter closed

Table B-7 (Page 2 of 2). Adapter Check for CCB1

Offset	Parameter Name	Byte Length	8086 Type	Description
4	REASON_CODE	2	DW	Adapter check reason code. See "Token-Ring Network Adapter Check Reason Codes for All CCBs" on page B-50.
6	PARAMETER_0	2	DW	Parameter 0: Set per reason code
8	PARAMETER_1	2	DW	Parameter 1: Set per reason code
10	PARAMETER_2	2	DW	Parameter 2: Set per reason code

Adapter Check for CCB2

In some instances, the adapter hardware or software is in a state in which operation is not possible. If this is the case, the following occurs:

1. If possible, the adapter closes and all network communications cease.
2. All adapters assume a closed state.
3. SAPs and link stations close because the adapter closes, and SAP and direct station buffer pools, pending receive frames, and CCBs can be returned to the application program if the ADAPTER_CHECK_FLAG is set.
4. If the ADAPTER_CHECK_FLAG is set the application program can be notified of this event. In order for an application program to receive notification of an adapter check, a READ command must be issued before the event occurs requesting notification of Critical Exceptions. When the event occurs, the READ command will be posted using a semaphore. The information listed in Table B-8 will be copied into the READ command's CCB parameter table.

Table B-8 (Page 1 of 2). Adapter Check for CCB2

Offset	Parameter Name	Byte Length	8086 Type	Description
0	NOTIFICATION_FLAG	4	DD	User exception flag.
4	CCB_COUNT	2	DW	Count of CCBs chained to EVENT_CCB_POINTER.
6	EVENT_CCB_POINTER	4	DD	Pointer to the first of a queue of commands that were pending when the adapter closed.
10	BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR.
12	FIRST_BUFFER_ADDR	4	DD	Address of first buffer in SAP and direct station buffer pools.

Table B-8 (Page 2 of 2). Adapter Check for CCB2

Offset	Parameter Name	Byte Length	8086 Type	Description
16	RCV_FRAME_COUNT	2	DW	Count of received frames chained to RCV_FRAME_ADDR.
18	RCV_FRAME_ADDR	4	DD	Address of the first received frame of a possible chain of frames.
22	REASON_CODE	2	DW	Adapter check reason code. See "Token-Ring Network Adapter Check Reason Codes for All CCBs" on page B-50.
Note: The next three fields are event error data.				
24	PARAMETER_0	2	DW	Parameter 0: set per reason code.
26	PARAMETER_1	2	DW	Parameter 1: set per reason code.
28	PARAMETER_2	2	DW	Parameter 2: set per reason code.

NOTIFICATION_FLAG

Explanation: This user exception flag is ADAPTER_CHECK_FLAG as defined using the DIR.SET.EXCEPTION.FLAGS command.

FIRST_BUFFER_ADDR

Explanation: Address of the first buffer in the SAP or direct buffer pool.

Buffers provided with the DLC.OPEN.SAP or DIR.OPEN.DIRECT commands are returned to the application program when the adapter enters the closed state.

RCV_FRAME_ADDR

Explanation: Address of the first received frame.

All received frames for this application program that were on the completion list at the time of the exception are queued to this field when the adapter enters the closed state. The first buffer of each frame will point to the next frame.

REASON_CODE

Explanation: Reason code for the adapter check. See "Token-Ring Network Adapter Check Reason Codes for All CCBs" on page B-50.

Note: Only one reason code will be reported at a time.

PARAMETERS - 0,1,2

Explanation: PARAMETER_0, PARAMETER_1, and PARAMETER_2 provide additional information on a per reason code basis. The information can be useful for maintenance purposes and is not intended for application program use.

Adapter Check for CCB3

In some instances, the adapter hardware or software is in a state in which operation is not possible. If this is the case, the following will occur:

1. If possible, the adapter closes and all network communications cease.
2. All adapters assume a closed state.
3. If SAPs and link stations close because the adapter closes and the adapter check appendage (ADAPTER_CHECK_APPNDG_OFFSET) is defined, the SAP and direct station buffer pools and CCBs can be returned to the application program in the information table pointed to by registers ES and BX. When the adapter support software calls the application program's device driver, the appropriate event appendage offset is passed in register DI. See the DIR.SET.EXCEPTION.FLAGS command on page 3-38.
4. If the adapter check appendage offset is specified, the application program can be notified of this event. Once the adapter check occurs, the adapter support software notifies the application program of the event by calling the application program's device driver with the appropriate event appendage offset passed in register DI. The information listed in Table B-9 will be provided in the table pointed to by registers ES and BX when the adapter support software calls the application program's device driver.

Application Program Calls

When the adapter support software calls the application program's device driver entry point, the following information is provided to the application program:

- An invocation code of X'0001' has been pushed onto the stack. Before returning to the adapter support software, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the adapter check appendage as defined by the DIR.SET.EXCEPTIONS.FLAG command.
- Register DS contains the application program's device driver protect mode data segment.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of a 14-byte information table.
- Register AX contains the error code.

Table B-9 (Page 1 of 2). Adapter Check for CCB3

Offset	Parameter Name	Byte Length	8086 Type	Description
0	CCB_COUNT	2	DW	Count of CCBs chained to EVENT_CCB_POINTER.
2	EVENT_CCB_POINTER	4	DD	Pointer to the first of a queue of commands that were pending when the adapter closed.
6	BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR.

Table B-9 (Page 2 of 2). Adapter Check for CCB3

Offset	Parameter Name	Byte Length	8086 Type	Description
8	FIRST_BUFFER_ADDR	4	DD	Address of first buffer in SAP and direct station buffer pools.
12	REASON_CODE	2	DW	Adapter check reason code, see "Token-Ring Network Adapter Check Reason Codes for All CCBs" on page B-50.
Note: The next three fields are event error data.				
14	PARAMETER_0	2	DW	Parameter 0: set per reason code.
16	PARAMETER_1	2	DW	Parameter 1: set per reason code.
18	PARAMETER_2	2	DW	Parameter 2: set per reason code.

FIRST_BUFFER_ADDR

Explanation: Address of the first buffer in the SAP and direct station buffer pool.

Buffers provided with the DLC.OPEN.SAP or DIR.OPEN.DIRECT commands are returned to the application program when the adapter enters the closed state.

REASON_CODE

Explanation: Reason code for the adapter check. See "Token-Ring Network Adapter Check Reason Codes for All CCBs" on page B-50.

PARAMETERS - 0,1,2

Explanation: PARAMETER_0, PARAMETER_1 and PARAMETER_2 provide additional information on a per reason code basis. The information can be useful for maintenance purposes only and is not intended for application program use.

Token-Ring Network Adapter Check Reason Codes for All CCBs

Table B-10. IBM Token-Ring Network Adapter Check Reason Codes for All CCBs

Value	Function	Meanings/Parameters
8000	Adapter inoperative	See note.
4000	Reserved	
2000	Reserved	
1000	Illegal op code	The adapter detected an illegal op code (micro failure).
0800	Local bus parity error	The adapter local bus detected a parity error.
0400	External parity error	
0200	Reserved	
0100	Internal parity error	
0080	Parity error, ring transmit	The adapter local bus detected a parity error while transmitting on the ring.
0040	Parity error, ring receive	The adapter local bus detected a parity error while receiving from the ring.
0020	Transmit underrun	
0010	Receive overrun	
0008	Unrecognized interrupt	
0004	Unrecognized error interrupt	
0003	Adapter detected no workstation service	
0002	Unrecognized supervisory request	
0001	Program detected error	

Adapter inoperative (8000): When a machine check occurs in the adapter processor, it is reported to the adapter support software via an "adapter check interrupt." The workstation can receive this interrupt before the adapter processor is able to set the adapter check bits. Therefore the adapter support software does the following:

1. If a reason code is set, that code is passed to the application program.
2. If a reason code is not set, the adapter support software goes into a tight loop for 250 ms. The adapter support software then checks the reason code set by the adapter processor and does one of the following:
   a. If a code is set, the adapter support software passes that code to the application program.
   b. If no code is set, the adapter support software assumes that the adapter processor's machine check handler was not capable of executing because of the severity of the processor's problem. The adapter support software then sets a value of (8000) in the adapter check reason code and passes that code to the application program.

PC Network and Ethernet Adapter Check Reason Codes for All CCBs

The values shown in Table B-11 for the reason codes are as if the values were contained in the AX register.

Table B-11. IBM PC Network Adapter Check Reason Codes for All CCBs

Code	Meaning
X'0100'	Program Detected Error
X'0200'	Reserved
X'0300'	Invalid Supervisor Request Code
X'0400'	Invalid Task ID on Supervisor Call
X'0500'	Invalid Ready Task Request
X'0600'	Invalid Adapter Number on Supervisor Call
X'0700'	Invalid ES Value on Supervisor Call

Network Status

The following sections document network status for CCB1, CCB2, and CCB3.

Network Status for CCB1

Whenever network status changes, the application program is notified if the network status appendage has been defined in the NETW_STATUS_EXIT field of the CCB for a DIR.INITIALIZE or in the NETW_STATUS_EXIT field of the DIRECT_PARMS table of the DIR.OPEN.ADAPTER command. See "Token-Ring Network Status Codes for All CCBs" on page B-55 for the information returned.

The AX register contains the network status code and the CX register contains the adapter number. Registers ES and BX point to the chain of pending CCBs if the adapter was closed.

Network Status for CCB2

In some instances the network status indicates that the adapter closed due to an unrecoverable error (Critical Network Status). When the adapter is closed, link stations are closed and the SAP and direct station buffer pools, pending receive frames, and CCBs can be returned to the application program if the NETWORK_STATUS_FLAG is set. See the DIR.SET.EXCEPTION.FLAGS command on page 3-38.

Whenever network status changes, the application program can be notified if the NETWORK_STATUS_FLAG is set. In order for an application program to receive notification of a network status (non-critical), a READ command must be issued. If a READ command is already pending, it will be posted immediately using a semaphore.

In order for an application program to receive notification of a Critical Network Status (the adapter closes), a READ command must be issued before the event occurs requesting notification of critical exceptions. When the event occurs, the READ command will be posted using a semaphore.

The information in Table B-12 on page B-53 will be copied into the READ command's parameter table.

Table B-12. Network Status for CCB2

Parameter Name	Byte Length	8086 Type	Description
NOTIFICATION_FLAG	4	DD	User exception flag
CCB_COUNT	2	DW	Count of CCBs chained to EVENT_CCB_POINTER *
EVENT_CCB_POINTER	4	DD	Pointer to the first of a queue of commands that were pending when the adapter closed *
BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR *
FIRST_BUFFER_ADDR	4	DD	Address of first buffer in SAP and direct station buffer pools *
RCV_FRAME_COUNT	2	DW	Count of received frames chained to RCV_FRAME_ADDR *
RCV_FRAME_ADDR	4	DD	Address of the first received frame of a possible chain of frames *
NETWORK_STATUS_CODE	2	DW	Network status code

* Indicates that this value is returned only when the adapter is closed as a result of encountering an unrecoverable error state.

NOTIFICATION_FLAG

Explanation: User notification flag.

This user exception flag is NETWORK_STATUS_FLAG as defined using the DIR.SET.EXCEPTION.FLAGS command.

FIRST_BUFFER_ADDR

Explanation: Address of the first buffer in the SAP and direct station buffer pools.

Buffers provided with the DLC.OPEN.SAP and DIR.OPEN.DIRECT commands are returned to the application program if the adapter closes.

RCV_FRAME_ADDR

Explanation: Address of the first received frame.

All received frames for this application program that were on the completion list at the time of the exception will be queued to this field if the adapter closes. The first buffer of each frame will point to the next frame.

NETWORK_STATUS_CODE

Explanation: Network status code being reported.

See "Token-Ring Network Status Codes for All CCBs" on page B-55 when using a Token-Ring Network adapter, and "PC Network and Ethernet Status Codes for All CCBs" on page B-56 when using a PC Network or Ethernet adapter.

Network Status for CCB3

In some instances the network status indicates that the adapter closed due to an unrecoverable error (Critical Network Status). When the adapter is closed, link stations are closed, and the SAP, direct station buffer pools and CCBs can be returned to the application program if the NETWORK_STATUS_APPNDG_OFFSET is set. See the DIR.SET.EXCEPTION.FLAGS command on page 3-38.

Whenever the network status changes, the application program is notified by a NETWORK_STATUS_APPNDG_OFFSET defined using the DIR.SET.EXCEPTION.FLAGS command. In order for an application program to receive notification of a network status exception (critical/non-critical), the DIR.SET.EXCEPTION.FLAGS command must have been executed, and the adapter support software must have been supplied with an appendage offset for network status.

When the event occurs, the adapter support software calls the application program's device driver entry point with the appropriate event appendage offset passed in register DI. The information listed in Table B-13 on page B-55 is supplied to the application program by the registers ES and BX. ES and BX point to the information table when the adapter support software calls to the application program's device driver.

Application Program Calls: When the adapter support software calls the application program at the address obtained by the ATTACHDD function, the following information is provided to the application program:

- An invocation code of X'0001' has been pushed onto the stack. Before returning to the adapter support software, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the network status appendage as defined by the DIR.SET.EXCEPTIONS.FLAGS command.
- Register DS contains the application program device driver's protect mode data segment.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of the following 14-byte information table.
- Register AX contains the network status code.

Table B-13. Network Status for CCB3

Parameter Name	Byte Length	8086 Type	Description
CCB_COUNT	2	DW	Count of CCBs chained to EVENT_CCB_POINTER *
EVENT_CCB_POINTER	4	DD	Pointer to the first of a queue of commands that were pending when the adapter closed *
BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR *
FIRST_BUFFER_ADDR	4	DD	Address of first buffer in SAP and direct station buffer pools *
NETWORK_STATUS_CODE	2	DW	Network status code

* Indicates that this value is returned only when the adapter is closed as a result of encountering an unrecoverable error.

FIRST_BUFFER_ADDR

Explanation: Address of the first buffer in the SAP and direct station buffer pools.

Buffers provided with the DLC.OPEN.SAP and DIR.OPEN.DIRECT commands are returned to the application program if the adapter closes.

NETWORK_STATUS_CODE

Explanation: Network status code being reported.

See "Token-Ring Network Status Codes for All CCBs" on page B-55 when using a Token-Ring Network adapter, and "PC Network and Ethernet Status Codes for All CCBs" on page B-56 when using a PC Network or Ethernet adapter.

Token-Ring Network Status Codes for All CCBs

Table B-14 (Page 1 of 2). Token-Ring Network Status Codes for All CCBs

Bit	Function	Meaning
15	Signal loss	Absence of any received signal detected.
14	Hard error	Beacon frames are being transmitted or received.
13	Soft error	This adapter has transmitted a soft error report MAC frame.
12	Transmit beacon	The adapter is transmitting beacon frames.
11	Lobe wire fault	An open or short circuit has been detected in the lobe data path. The adapter will be closed.
10	Auto-removal error 1	An adapter hardware error has been detected following the beacon auto-removal process. The adapter has been removed from the ring. The adapter will be closed.
9	Reserved	
8	Remove received	A remove MAC frame has been received. The adapter will be closed.
7	Counter overflow	One of the adapter error log counters has been incremented from 254 to 255. The DIR.READ.LOG command should be issued.

Table B-14 (Page 2 of 2). Token-Ring Network Status Codes for All CCBs

Bit	Function	Meaning
6	Single station	The adapter has opened and is the only station on the ring. The bit will be reset when another station is detected.
5	Ring Recovery	The adapter is transmitting or receiving Monitor Contention (claim token) MAC frames. This bit will be reset upon receipt of a Ring Purge MAC frame.
0-4	Reserved	

Multiple bits can be set when a network status change is posted.

For a beaconing condition, the following network status events will be reported in the following order:

- For the station that is initially transmitting the beacon frames
  - Ring Recovery (bit 5) set
  - Then after 1 second:
    - Signal Loss (bit 15) set
    - Hard Error (bit 14) set
    - Transmit Beacon (bit 12) set
- For the station that is initially receiving the beacon frames
  - Ring Recovery (bit 5) set
  - Then after 1 second, Hard Error (bit 14) set.

The Ring Recovery bit remains on for the entire time that the ring is beaconing. When using OS/2, if the ring is beaconing when an application program issues the DIR.OPEN.ADAPTER command and then issues the DIR.STATUS command, the Ring Recovery bit will be set. However, bits 15, 14, and 12 can toggle depending on the immediate state of the ring seen by each adapter as a result of adapters doing diagnostic testing. When the ring stops beaconing all bits including the Ring Recovery bit will be zero.

In existing applications for the Token-Ring, the application assumes that the adapter has closed upon receiving a lobe wire fault, an auto-removal, or a remove-received network status flag.

PC Network and Ethernet Status Codes for All CCBs

Table B-15 (Page 1 of 2). PC Network and Ethernet Status Codes for All CCBs

Bit	Function	Meaning
15	No Carrier	No carrier on this card during transmit. **
14	Reserved	
13	Reserved	
12	Continuous Carrier	The adapter has detected Continuous Carrier on the network for more than 30 seconds. **
11	Reserved	
10	Continuous Carrier	This adapter is generating Continuous Carrier. (The adapter will be closed.) **

** This value is not set on Ethernet.

Table B-15 (Page 2 of 2). PC Network and Ethernet Status Codes for All CCBs

Bit	Function	Meaning
9	Reserved	
8	Remove Received	A network management REMOVE frame has been received. (The adapter will be closed.) **
7	Counter Overflow	One of the Error Log Counters has incremented from 254 to 255.
0-6	Reserved	

** This value is not set on Ethernet.

Bring-Up Errors for All CCBs

For CCB1: Bring-up testing is done when the DIR.INITIALIZE command is executed. If these tests are not completed successfully indicating an adapter failure, the bring-up error code will be returned in the BRING_UP field of the DIR.INITIALIZE parameter table. The CCB_RETCODE in the CCB will also contain X'07' (command canceled: unrecoverable failure) when the command is terminated.

For CCB2 and CCB3: Bring-up testing is done during initialization when the adapter support software is loaded and when the DIR.INITIALIZE command is issued with the correct System Key. The results of the bring-up tests are returned to application programs when the DIR.OPEN.ADAPTER and DIR.INITIALIZE commands are executed.

The bring-up error code is included in the ADAPTER_PARMS table of the DIR.OPEN.ADAPTER command.

If during system initialization bring-up testing is not successful, error messages are displayed and logged in the ACSLAN.LOG file.

The values shown for the bring-up codes are as if the values were contained in the AX register.

Table B-16 (Page 1 of 2). Bring-up Error Codes for All CCBs (Token-Ring Network Adapters)

Code	8086 Type	Meaning
X'0020'	DW	Diagnostics could not execute.
X'0022'	DW	ROM (ROS) diagnostics failed.
X'0024'	DW	Shared RAM diagnostics failed.
X'0026'	DW	Processor instruction test failed.
X'0028'	DW	Processor interrupt test failed.
X'002A'	DW	Shared RAM interface register diagnostics failed.
X'002C'	DW	Protocol-handler diagnostics failed.
X'0040'	DW	Adapter's programmable timer for the workstation failed (set by the workstation code).
X'0042'	DW	Cannot write to shared RAM (set by the workstation code).

Table B-16 (Page 2 of 2). Bring-up Error Codes for All CCBs (Token-Ring Network Adapters)

Code	8086 Type	Meaning
X'0044'	DW	Reading from shared RAM read-only area caused an invalid error indication (interrupt) (set by the workstation code).
X'0046'	DW	Writing into shared RAM read-only area did not cause an error indication (interrupt) (set by the workstation code).
X'0048'	DW	Initialization timed out.

Token-Ring Network Adapter Open Errors for All CCBs

For CCB1: Adapter open testing is done when the DIR.OPEN.ADAPTER command is executed. If these tests are not completed successfully, indicating either an adapter failure or a ring problem, the open error codes will be returned in the OPEN_ERROR_CODE field of the DIR.OPEN.ADAPTER parameter table. The CCB_RETCODE in the CCB will also contain X'07' (command canceled—unrecoverable failure) when the command is terminated.

For CCB2 and CCB3: Adapter open testing is done at system initialization, or when a physical open is issued as a result of the DIR.OPEN.ADAPTER command being issued. If these tests do not execute successfully, any subsequent DIR.OPEN.ADAPTER commands will be terminated with a return code of X'07'.

The open error codes are passed back to the user in the DIR.OPEN.ADAPTER parameter table.

Open Error Codes for All CCBs

The open errors are returned in 2 bytes. The high-order byte is always zero and the low-order byte contains the following:

1. The phase of testing in which the error was encountered is in the high-order nibble (half-byte) of the low-order byte.
2. The error condition is in the low-order nibble of the low-order byte.

Phases

Table B-17. Phases

Value	Meaning
'1n'	Lobe media test
'2n'	Physical insertion
'3n'	Address verification
'4n'	Roll call poll (neighbor notification)
'5n'	Request parameters

Errors

Table B-18. Errors

Value	Meaning
'n1'	Function failure
'n2'	Signal loss
'n3'	Reserved
'n4'	Reserved
'n5'	Timeout
'n6'	Ring failure
'n7'	Ring beaconing
'n8'	Duplicate node address
'n9'	Parameter request
'nA'	Remove received
'nB'	Reserved
'nC'	Reserved
'nD'	No monitor detected
'nE'	Monitor contention failed for RPL

Suggested Actions in Response to Open Errors

When the following *Phase - Error* combination values are presented, they are the result of certain specific occurrences. Explanation of the occurrences follows with recommended actions listed. Table B-19 on page B-63 lists the recommended actions for both the application program and the workstation operator.

X'11' **Lobe Media, Function Failure**

Failure Definition: The testing of the lobe between the adapter and the access unit has been unsuccessful because the lobe has a bit-error rate that is too high, or the adapter cannot receive successfully.

Recommended Actions: 1, 3, and 5

X'26' **Physical Insertion, Ring Failure**

Failure Definition: The adapter, acting as an active monitor, was unable to complete the ring purge function successfully. This indicates that an error condition has occurred since the successful completion of monitor contention (claim token), when this adapter became the active monitor.

Recommended Actions: 1 and 2a

X'27' **Physical Insertion, Ring Beaconing**

Failure Definition: The adapter has detected one of the following conditions.

- The adapter tried to insert on a ring that was operating at a different data rate.
- A monitor contention (claim token) failure occurred.

- The adapter received a beacon MAC frame from the ring.

Recommended Actions: 1, 2, and 2b

X'2A' **Physical Insertion, Timeout**

Failure Definition: The adapter has received a remove ring station MAC frame indicating that a network management function has directed this adapter to get off the ring.

Recommended Actions: 2a and 4

X'2D' **No Monitor Detected**

Failure Definition: RPL station is the first station attempting to insert onto the ring.

Recommended Actions: 1 and 2a

X'2E' **Monitor Contention Failed for RPL**

Failure Definition: Physical insertion failure of RPL station.

Recommended Actions: 2

X'32' **Address Verification, Signal Loss**

Failure Definition: The adapter has detected a 250-ms signal loss (receiver cannot recognize signal) indicating that an error condition has occurred since the adapter successfully completed the ring signal recognition phase of the open operation.

Recommended Actions: 1 and 2a

X'35' **Address Verification, Timeout**

Failure Definition: The insertion timer has expired before this function completed, indicating that the ring can be congested, experiencing a high bit-error rate, or losing an abnormally high number of tokens or frames, thus preventing successful Address Verification MAC frame transmissions.

Recommended Actions: 1 and 2a

X'36' **Address Verification, Ring Failure**

Failure Definition: The adapter, acting as an active monitor, was unable to complete the ring purge function successfully. This indicates that an error condition has occurred since the successful completion of monitor contention (claim token), when this adapter became the active monitor.

Recommended Actions: 1 and 2a

X'37' **Address Verification, Ring Beaconing**

Failure Definition: The adapter has either detected a monitor contention (claim token) failure or received a beacon MAC frame from the ring.

Recommended Actions: 1 and 2b

X'38' **Address Verification, Duplicate Node Address**

Failure Definition: The adapter has detected that another station on the ring has an adapter address which is the same as the adapter address being tested.

Recommended Actions: 4

X'3A' **Address Verification, Remove Received**

Failure Definition: The adapter has received a remove ring station MAC frame indicating that a network management function has directed this specific address to get off the ring.

Recommended Actions: 2a and 4

X'42' **Ring Poll, Signal Loss**

Failure Definition: The adapter has detected a 250-ms signal loss (receiver cannot recognize signal) indicating that an error condition has occurred since the adapter successfully completed the ring signal recognition phase of the open operation.

Recommended Actions: 1 and 2a

X'45' **Ring Poll, Timeout**

Failure Definition: The insertion timer has expired before this function completed, indicating that the ring can be congested, experiencing a high bit-error rate, or losing an abnormally high number of tokens or frames. This prevents the adapter's successful reception of either the ring poll request or response MAC frame, or transmission of the required ring poll response MAC frame.

Recommended Actions: 1 and 2a

X'46' **Ring Poll, Ring Failure**

Failure Definition: The adapter, acting as an active monitor, was unable to complete the ring purge function successfully. This indicates that an error condition has occurred since the successful completion of monitor contention (claim token), when this adapter became the active monitor.

Recommended Actions: 1 and 2a

X'47' **Ring Poll, Ring Beaconing**

Failure Definition: The adapter has either detected a monitor contention (claim token) failure or received a beacon MAC frame from the ring.

Recommended Actions: 1 and 2b

X'4A' **Ring Poll, Remove Received**

Failure Definition: The adapter has received a remove ring station MAC frame, indicating that a network management function has directed this adapter to get off the ring.

Recommended Actions: 2a and 4

X'55' **Request Parameters, Timeout**

Failure Definition: The insertion timer has expired before this function completed, indicating that the ring can be congested, experiencing a high bit-error rate, or losing an abnormally high number of tokens or frames. This prevents successful transmission of the request parameter MAC frame or reception of either the set parameters 1 or set parameters 2 MAC frame (required response to the adapter's request).

Recommended Actions: 1 and 2a

X'56' **Request Parameters, Ring Failure**

Failure Definition: The adapter, acting as an active monitor, was unable to complete the ring purge function successfully. This indicates that an error condition has occurred since the successful completion of monitor contention (when this adapter became the active monitor).

Recommended Actions: 1 and 2a

X'57' **Request Parameters, Ring Beaconing**

Failure Definition: The adapter has received a beacon MAC frame from the ring.

Recommended Actions: 1 and 2b

X'59' **Request Parameters, Parameter Request**

Failure Definition: The adapter has detected that the ring parameter server is present on the ring but that the required response (set parameters 1 or set parameter 2 MAC frame) has not been received in the allotted time. This indicates that the ring can be congested, experiencing a high bit-error rate, or losing an abnormally high number of tokens or frames.

Recommended Actions: 1 and 2a

X'5A' **Request Parameters, Remove Received**

Failure Definition: The adapter has received a remove ring station MAC frame, indicating that a network management function has directed this adapter to get off the ring.

Recommended Actions: 2a and 4

Recommended Actions Table

Table B-19. Recommended Actions Table

No.	Description
1	After delaying at least 30 seconds, retry the open two times, inserting the same delay between each retry.
2	After delaying at least 30 seconds, check the adapter configuration (especially the adapter data rate) and retry the open.
2a	If this error persists, direct the workstation operator to contact the network administrator for assistance and provide "Open Error" information.
2b	If this error persists, direct the workstation operator to contact the network administrator for assistance and provide information from the "Adapter Status Parameter Table" on page B-44.
3	Direct the workstation operator to contact the network administrator for assistance and provide "Open Error" information.
4	Direct the workstation operator to contact the network administrator for assistance and provide "Node Address" information and try attaching to the ring after 6 minutes.
5	If this error persists, problem determination of the adapter or lobe is necessary. Contact your network administrator for problem determination assistance.

PC Network and Ethernet Adapter Open Errors for All CCBs

The open error codes are returned in a DW. The high order byte is always zero.

Table B-20. PC Network and Ethernet Adapter Open Errors for All CCBs

Value	Meaning
X'0022'	No carrier **
X'0023'	Continuous carrier **
X'0033'	Unable to transmit
X'0038'	Duplicate node address
X'003A'	REMOVE frame received **

** This value is not set on Ethernet.

PC System Detected Errors

The following sections document PC detected errors for CCB1, CCB2, and CCB3.

PC System Detected Errors for CCB1

These types of errors are encountered by the protocol drivers during operation.

Note: The PC System detected error is not intended as a substitute for normal workstation error detection, for example, a divide check.

If the adapter support software detects an error condition in the workstation, the sequence of events will be determined by the type of error, which is passed to the appendage in register AL.

This appendage is defined by the DIR.OPEN.ADAPTER, DIR.SET.USER.APPENDAGE, DIR.INITIALIZE, and DIR.MODIFY.OPEN.PARMS commands.

The following information is passed to the appendage:

1. Register AL contains the error code.
2. Register AH contains information on a per error code basis.

   If register AH is used, it is indicated in the status code explanation.
3. Register CX contains the adapter number.
4. Registers ES and BX:

   If both registers are not X'0000', they are pointers to a queue of commands that were pending when the error occurred.

   If both registers are X'0000', either the command being executed could not be determined or it is not applicable.

The error code is passed to the appendage in register AX.

The PC System detected error codes are:

Code	**Meaning**
X'0000'	Spurious interrupt detected.
X'0001'	Access violation. An attempt to write into the read-only portion of shared RAM has occurred.
X'01xx'	An ARB command code error, where "xx" is the command code.
X'02xx'	An ARB return code error, where "xx" is the return code from the adapter.
X'03xx'	An SRB/SSB command code error, where "xx" is the CCB command code.
X'04xx'	ARB transmit data request error (the transmit CCB was not found), where "xx" is the command correlator from the adapter.

PC System Detected Errors for CCB2

These types of errors are encountered by the adapter support software during operation.

Note: The PC System detected error is not intended as a substitute for normal workstation error detection, for example, a divide check.

If the adapter support software detects an error condition in the workstation or OS/2 generates a return code that is not acceptable for the given situation, an error code is passed to the application program through the READ command.

In these cases the adapter will be closed, and adapter support software will assume the adapter has encountered an unrecoverable error.

When the unrecoverable error is entered, link stations are closed and SAP, direct station buffer pools, pending receive frames, and CCBs can be returned to the application program if the PC_ERROR_FLAG is set. See DIR.SET.EXCEPTION.FLAGS on page 3-38.

Whenever a PC System detected error occurs, the application program can be notified if the PC_ERROR_FLAG is set. In order for an application program to receive notification of a PC System Detected error a READ command must be issued before the event occurs requesting notification of critical exceptions. When the event occurs, the READ command will be posted using a semaphore.

The information listed in Table B-21 will be copied into the READ command parameter table.

Table B-21. PC System Detected Errors for CCB2

Parameter Name	Byte Length	8086 Type	Description
NOTIFICATION_FLAG	4	DD	User exception flag
CCB_COUNT	2	DW	Count of CCBs chained to EVENT_CCB_POINTER
EVENT_CCB_POINTER	4	DD	Pointer to the first of a queue of commands that were pending when the adapter closed
BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR
FIRST_BUFFER_ADDR	4	DD	Address of first buffer in SAP and direct station buffer pools
RCV_FRAME_COUNT	2	DW	Count of received frames chained to RCV_FRAME_ADDR
RCV_FRAME_ADDR	4	DD	Address of the first received frame of a possible chain of frames
ERROR_DATA/ERROR_CODE	2	DW	error data and error code
FUNCTION_CODE	1	DB	OS/2 command code that failed
	1	DB	Reserved
	2	DW	Reserved
PROCESS_ID	2	DW	Process ID

NOTIFICATION_FLAG

Explanation: User notification flag.

This user exception flag is the PC_ERROR_FLAG as defined using the DIR.SET.EXCEPTIONS.FLAG command.

FIRST_BUFFER_ADDR

Explanation: Address of the first buffer in SAP and direct station buffer pools.

Buffers provided with the DLC.OPEN.SAP and DIR.OPEN.DIRECT commands are returned to the application program when the adapter closes as a result of an unrecoverable error.

RCV_FRAME_ADDR

Explanation: Address of the first received frame.

All received frames for this application program that were on the completion list at the time of the exception will be queued to this field when the adapter encounters an unrecoverable error. The first buffer of each frame will point to the next frame.

ERROR_DATA/ERROR_CODE

Explanation: Theses two fields contain the error code and associated data for the PC System detected error.

The following list indicates both the ERROR_DATA and the ERROR_CODE. The format is X'ddcc' where dd is the error data and cc is the error code.
byte.

Code	Meaning
X'0000'	A Token-Ring Network adapter interrupt has occurred and the interrupt was unexpected. For example, the adapter is not open.
X'0001'	Access violation: an attempt was made to write to the read-only portion of shared RAM.
X'01xx'	ARB command code error: an undefined ARB was returned from the adapter. The "xx" indicates the command code.
X'02xx'	ASB return code error: there was an unexpected ASB interrupt from the adapter as a result of an ARB command. The "xx" will be set to the return code from the adapter.
X'03xx'	SRB/SSB command code error. The "xx" will be set to the CCB command code.
X'04xx'	ARB transmit data request error: the transmit CCB was not found. The "xx" is the command correlator from the adapter.
X'05xx'	Unacceptable error conditions resulting from OS/2 return code values. If, while processing on the thread of an application program in the adapter support software device driver, an OS/2 return code is generated that is not acceptable for a given situation, the return code in the AX register will contain X'04'.
X'06xx'	An adapter support software internal error. The "xx" will be set to an internal error code.

Note: Codes not shown (X'07' through X'7F') are reserved.

FUNCTION_CODE

Explanation: The OS/2 command code that failed.

This field is only used for an ERROR_CODE of X'05'.

This field will contain the function code of a Device Help command that resulted in this PC System detected error.

PROCESS_ID

Explanation: The OS/2 process ID.

This field is only used for the error codes X'05' and X'06', and will contain the OS/2 process ID that was dispatched while the error occurred. A process ID of X'0000' is used when an error occurs while processing an interrupt.

PC System Detected Errors for CCB3

These types of errors are encountered by the adapter support software during operation.

Note: The PC System detected error is not intended as a substitute for normal workstation error detection, for example, a divide check.

If the adapter support software detects an error condition in the workstation, or OS/2 generates a return code that is not acceptable for the given situation, an error code is passed to the application program when the adapter support software calls the application program's device driver with the appropriate event appendage offset passed in register DI. Upon entry into the application program's device driver, the AX register will contain the error code.

In these cases the adapter will be closed, and the adapter support software will assume that the adapter has encountered an unrecoverable error.

When the unrecoverable error is entered:

- Link stations are closed.
- SAP and direct station buffer pools, and CCBs can be returned to the application program if the PC detected error appendage (PCERROR_APPNDG_OFFSET) has been defined using the DIR.SET.EXCEPTION.FLAGS command.

See DIR.SET.EXCEPTION.FLAGS on page 3-38. If the appendage has been defined, the adapter support software will call the application program's device driver with the PC system detected appendage offset passed in register DI to notify the user of the error.

Whenever a PC System detected error occurs, the application program can be notified if the PC detected error appendage has been defined using the DIR.SET.EXCEPTION.FLAGS command.

The information listed in Table B-22 on page B-68 is contained in the table pointed to by registers ES and BX when the adapter support software calls the application program's device driver with the appropriate event appendage offset passed in register DI.

Application Program Calls: When the adapter support software calls the application program at the address obtained by the ATTACHDD function, the following information is provided to the using application program.

- An invocation code of X'0001' has been pushed onto the stack. Before returning to the adapter support software, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the adapter check appendage as defined by the DIR.SET.EXCEPTIONS.FLAG command.

- Register DS contains the application program device driver's protect mode data segment.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of the following 20-byte information table.
- Register AX contains the error code.

Table B-22. PC System Detected Errors for CCB3

Parameter Name	Byte Length	8086 Type	Description
CCB_COUNT	2	DW	Count of CCBs chained to EVENT_CCB_POINTER
EVENT_CCB_POINTER	4	DD	Pointer to the first of a queue of commands that were pending when the adapter closed
BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR
FIRST_BUFFER_ADDR	4	DD	Address of first buffer in SAP and direct station buffer pools
ERROR_DATA/ERROR_CODE	2	DW	Error data and error code
FUNCTION_CODE	1	DB	OS/2 command code that failed
	1	DB	Reserved
	2	DW	Reserved
	2	DW	Reserved

FIRST_BUFFER_ADDR

Explanation: Address of the first buffer in SAP and direct station buffer pools.

Buffers provided with the DLC.OPEN.SAP and DIR.OPEN.DIRECT commands are returned to the application program when the adapter closes as a result of an unrecoverable error.

ERROR DATA/ERROR CODE

Explanation: These two fields contain the error code and associated data for the PC System detected error.

The following list indicates both the ERROR_DATA and the ERROR_CODE. The format is X'ddcc' where dd is the error data and cc is the error code.
byte.

Code	Meaning
X'0000'	A Token-Ring Network adapter interrupt has occurred and the interrupt was unexpected. For example, the adapter is not open.
X'0001'	Access violation: an attempt was made to write to the read-only portion of shared RAM.
X'01xx'	ARB command code error: an undefined ARB was returned from the adapter. The "xx" indicates the command code.

X'02xx'	ASB return code error: there was an unexpected ASB interrupt from the adapter as a result of an ARB command. The "xx" will be set to the return code from the adapter.
X'03xx'	SRB/SSB command code error. The "xx" will be set to the CCB command code.
X'04xx'	ARB transmit data request error: the transmit CCB was not found. The "xx" is the command correlator from the adapter.
X'05xx'	Unacceptable error conditions resulting from OS/2 return code values. If, while processing on the thread of an application program in the adapter support software device driver, an OS/2 return code is generated that is not acceptable for a given situation the immediate return code in the AX register will contain X'04'.
X'06xx'	An adapter support software internal error. The "xx" will be set to an internal error code.

Note: Codes not shown (X'07' through X'7F') are reserved.

FUNCTION_CODE

Explanation: The OS/2 command code that failed.

This field is only used for an ERROR_CODE of X'05'.

This field will contain the Function Code of a Device Help command that resulted in this PC System detected error.

System Action Exceptions for OS/2 EE 1.3

The following sections document System Action exceptions for CCB2 and CCB3.

System Action Exceptions for CCB2

This exception is the result of a system administrator issuing commands using the System Key defined by the configuration parameters. The following commands when issued with the System Key will result in a system action exception.

DIR.CLOSE.ADAPTER
DIR.CLOSE.DIRECT
DIR.INITIALIZE
DIR.READ.LOG
DIR.SET.FUNCTIONAL.ADDRESS
DIR.SET.GROUP.ADDRESS
DLC.RESET

System action exceptions result in an adapter closing, an adapter initializing, reading of either the adapter or direct interface logs, modification of the functional or group addresses, resetting link stations, or forced availability (closing) of the direct stations. When link stations are closed or the direct stations are closed due to a System Action exception, the link and direct station's buffer pools, pending receive frames, and CCBs can be returned to the application program if the SYSTEM_ACTION_FLAG is set. See the DIR.SET.EXCEPTION.FLAGS command on page 3-38.

System Action Exceptions

Whenever a System Action occurs, the application program is notified if the SYSTEM_ACTION_FLAG is set and the using code has a READ pending. To receive notification of a System Action exception resulting from the DIR.CLOSE.ADAPTER and DIR.INITIALIZE commands, the READ command must be issued before the event occurs. The information listed in Table B-23 is copied into the READ command's parameter table and the READ command's semaphore is cleared to post the application program.

For the commands and associated application programs that follow, if the SYSTEM_ACTION_FLAG is set, the adapter support software will notify the application program of system action exceptions.

- DIR.CLOSE.ADAPTER for all application programs
- DIR.CLOSE.DIRECT for an application program owning the direct interface (a previously issued DIR.OPEN.DIRECT command was successful)
- DIR.INITIALIZE for all application programs
- DIR.READ.LOG for an application program owning the direct interface (a previously issued DIR.OPEN.DIRECT command was successful)
- DIR.SET.FUNCTIONAL.ADDRESS for all application programs
- DIR.SET.GROUP.ADDRESS for all application programs
- DLC.RESET for application programs owning the affected SAPs

Table B-23. System Action Exceptions for CCB2

Parameter Name	Byte Length	8086 Type	Description
NOTIFICATION_FLAG	4	DD	User exception flag
CCB_COUNT	2	DW	Count of CCBs chained to EVENT_CCB_POINTER
EVENT_CCB_POINTER	4	DD	Pointer to the first of a queue of commands that were pending when the adapter closed or when direct stations are closed
BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR
FIRST_BUFFER_ADDR	4	DD	Address of first buffer in SAP or DIRECT buffer pools
RCV_FRAME_COUNT	2	DW	Count of received frames chained to RCV_FRAME_ADDR
RCV_FRAME_ADDR	4	DD	Address of the first received frame of a possible chain of frames
SYSTEM_ACTION_ID	1	DB	System action identifier
SAP_STATION_RESET	1	DB	Link station reset

NOTIFICATION_FLAG

Explanation: User notification flag.

This user exception flag is SYSTEM_ACTION_FLAG as defined using the DIR.SET.EXCEPTION.FLAGS command.

FIRST_BUFFER_ADDR

Explanation: Address of the first buffer in SAP and direct station buffer pools.

Buffers provided with the DLC.OPEN.SAP and DIR.OPEN.DIRECT commands are returned to the application program if link stations are closed or reset.

RCV_FRAME_ADDR

Explanation: Address of the first received frame.

All received frames for the affected link stations that were on the completion list at the time of the exception will be queued to this field. The first buffer of each frame will point to the next frame.

SYSTEM_ACTION_ID

Explanation: System action identifier code.

The system action identifier code is passed to the user in the SYSTEM_ACTION_ID field. This code identifies the command issued that generated the system action exception.

Code	Meaning
X'01'	DIR.CLOSE.ADAPTER command issued resulting in the adapter closing.
X'02'	DIR.INITIALIZE command issued reinitializing the adapter.
X'03'	DIR.READ.LOG command issued reading the adapter or direct interface logs.
X'04'	DIR.SET.FUNCTIONAL.ADDRESS command issued modifying the functional address.
X'05'	DIR.SET.GROUP.ADDRESS command issued modifying the group address.
X'06'	DLC.RESET command issued resetting a single SAP. When a single SAP is reset, the SAP_STATION_RESET field will contain the SAP that was reset.
X'07'	DLC.RESET command issued resetting all link stations. When all link stations are reset, no link station values are returned.
X'08'	DIR.CLOSE.DIRECT command issued resulting in the direct stations closing and becoming available.

SAP_STATION_RESET

Explanation: Link station reset.

If the SYSTEM_ACTION_ID is set to X'06', then this value contains the SAP number that was reset.

System Action Exceptions for CCB3

This exception is the result of a system administrator issuing commands using the System Key defined by the configuration parameters. The following commands when issued with the System Key will result in a system action exception.

DIR.CLOSE.ADAPTER
DIR.CLOSE.DIRECT
DIR.INITIALIZE
DIR.READ.LOG
DIR.SET.FUNCTIONAL.ADDRESS
DIR.SET.GROUP.ADDRESS
DLC.RESET

System action exceptions result in an adapter closing, an adapter initializing, reading of either the adapter or direct interface logs, modification of the functional or group addresses, resetting link stations, or forced availability (closing) of the direct stations. When link stations are closed or the direct stations are closed due to a System Action exception, the link and direct station's buffer pools and CCBs can be returned to the application program if the user has passed the system action appendage (SYSTEM_ACTION_APPNDG_OFFSET) to the adapter support software by issuing the DIR.SET.EXCEPTION.FLAGS command. See the DIR.SET.EXCEPTION.FLAGS command on page 3-38.

Whenever a System Action occurs, the application program is notified if the SYSTEM_ACTION_APPNDG_OFFSET has been defined to the adapter support software. Once the exception has occurred, the information listed in Table B-24 on page B-73 is copied into the table referenced by the ES and BX registers when the adapter support software calls the application program's device driver with the appropriate event appendage offset passed in register DI.

For the commands and associated application programs that follow, if the SYSTEM_ACTION_APPNDG_OFFSET is set, the adapter support software will notify the application program of system action exceptions.

- DIR.CLOSE.ADAPTER for all application programs
- DIR.CLOSE.DIRECT for an application program owning the direct interface (a previously issued DIR.OPEN.DIRECT command was successful)
- DIR.INITIALIZE for all application programs
- DIR.READ.LOG for an application program owning the direct interface (a previously issued DIR.OPEN.DIRECT command was successful)
- DIR.SET.FUNCTIONAL.ADDRESS for all application programs
- DIR.SET.GROUP.ADDRESS for all application programs
- DLC.RESET for application programs owning the affected SAPs

Application Program Calls: When the adapter support software calls the application program device driver entry point, the following information is provided to the using application program.

- An invocation code of X'0001' has been pushed onto the stack. Before returning to the adapter support software, the application program must remove the invocation code from the stack.
- Register DI contains the offset of the adapter check appendage as defined by the DIR.SET.EXCEPTIONS.FLAG command.
- Register DS contains the application program device driver's protect mode data segment.
- Register CX contains the adapter number.
- Registers ES and BX contain the address of the following 14-byte information table.
- Register AL contains the system action ID.
- Register AH contains the SAP value associated with the system action ID.

Table B-24. System Action Exceptions for CCB3

Parameter Name	Byte Length	8086 Type	Description
CCB_COUNT	2	DW	Count of CCBs chained to EVENT_CCB_POINTER
EVENT_CCB_POINTER	4	DD	Pointer to the first of a queue of commands that were pending when the adapter closed or when the direct stations are closed
BUFFER_COUNT	2	DW	Count of buffers chained to FIRST_BUFFER_ADDR
FIRST_BUFFER_ADDR	4	DD	Address of first buffer in SAP or direct station buffer pools
SYSTEM_ACTION_ID	1	DB	System action identifier
SAP_STATION_RESET	1	DB	Link station reset

FIRST_BUFFER_ADDR

Explanation: Address of the first buffer in SAP and direct station buffer pools.

Buffers provided with the DLC.OPEN.SAP and DIR.OPEN.DIRECT commands are returned to the application program if link stations are closed or reset.

SYSTEM_ACTION_ID

Explanation: System action identifier code.

The system action identifier code is passed to the user in the SYSTEM_ACTION_ID field. This code identifies the command issued that generated the system action exception.

Code	**Meaning**
X'01'	DIR.CLOSE.ADAPTER command issued resulting in the adapter closing.
X'02'	DIR.INITIALIZE command issued reinitializing the adapter.
X'03'	DIR.READ.LOG command issued reading the adapter and/or direct interface logs.
X'04'	DIR.SET.FUNCTIONAL.ADDRESS command issued modifying the functional address.
X'05'	DIR.SET.GROUP.ADDRESS command issued modifying the group address.
X'06'	DLC.RESET command issued resetting a single SAP When a single SAP is reset, the SAP_STATION_RESET field will contain the SAP that was reset.
X'07'	DLC.RESET command issued resetting all link stations. When all link stations are reset no link station values are returned.
X'08'	DIR.CLOSE.DIRECT command issued resulting in the direct stations closing.

SAP_STATION_RESET

Explanation: Link station reset.

If the SYSTEM_ACTION_ID is set to X'06', then this value contains the SAP number that was reset.

Appendix C. Local Area Network Sample Program Listings Diskette

Sample program listings are included on the diskette provided with this manual. These listings provide examples of NETBIOS use and the IEEE 802.2 interface use for both DOS and OS/2.

See the DOC files on the diskette for a description of the sample program listings.

The materials on this diskette could include technical inaccuracies, typographical errors, or incompatibilities with IBM products, and could be modified or excluded from the generally available reference materials. IBM expressly reserves the right, without notice to you, to modify the information contained in this book in a manner that affects the compatibility or usability of the application programs developed by you using the materials on this diskette.

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS DISKETTE "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

It is possible that this diskette may contain reference to, or information about IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country.

System Administrators may copy and distribute the sample programs on this diskette in any form without payment to IBM, for the purpose of developing, using, marketing, or distributing application programs for use with the IBM Token-Ring Network and IBM PC Network.

Attach a label that contains the following copyright notice to each copy:

Version 3.0 (C) Copyright International Business Machines Corp. 1986, 1990, 1991, 1993

Appendix D. The Local Area Network Support Program Interrupt Arbitrator

About This Appendix

This appendix describes the software interface and the registration process of the Local Area Network Support Program interrupt arbitrator. This module makes the X'5C' interrupt available to the program interfaces that use it.

Note: For all other information about the LAN Support Program, including descriptions of the parameters, refer to the *Local Area Network Support Program User's Guide* for your version of the LAN Support Program.

The Local Area Network Support Program Interrupt Arbitrator (DXMA0MOD.SYS)

This section describes the software interface to and the registration process of the Local Area Network Support Program interrupt arbitrator. The Local Area Network Support Program interrupt arbitrator is a software module provided with the Local Area Network Support Program diskette. The purpose of the Local Area Network Support Program interrupt arbitrator is to remove the majority of the load order dependencies of the interfaces provided by the Local Area Network Support Program.

Important: In this appendix the acronym CCB (command control block) will apply only to the IEEE 802.2 adapter support software supplied with the Local Area Network Support Program.

Currently there are two software interfaces to IBM's network adapters:

- NETBIOS
- The IEEE 802.2 Interface.

A usability problem arises because both of these interfaces utilize the X'5C' software interrupt in their API.

Since two network adapters can be installed when using the Local Area Network Support Program, multiple IEEE 802.2 interfaces or NETBIOS may be present in a workstation at one time. This situation currently requires that each interface know whether another interface is also using the X'5C' interrupt, so that it can pass any control blocks that it does not understand (or that are not for its adapter) to the next X'5C' interrupt handler in the chain.

This need to know whether the other interfaces are using the X'5C' interrupt results in replicated logic at the entry points to the interrupt handlers. In addition, due to migration considerations, the interrupt handlers may have to be loaded in a specific sequence to avoid conflicts.

The Local Area Network Support Program interrupt arbitrator eliminates these problems by having a single owner of the X'5C' software interrupt. It takes over the X'5C' interrupt vector and then allows the other users of the X'5C' interrupt to register with itself. Registering programs specify which interface they provide (IEEE

802.2 or NETBIOS) and for which network adapter (0 or 1). The Local Area Network Support Program interrupt arbitrator then monitors the flow of control blocks across the X'5C' interrupt and routes the control block to the appropriate interface.

The use of the Local Area Network Support Program interrupt arbitrator eliminates the two previously stated problems. Since there is a single owner of the X'5C' interrupt, all logic for determining how to route the control blocks is contained there, not replicated in each individual interface. In addition, since each interface registers its needs with the Local Area Network Support Program interrupt arbitrator, the interface modules can be loaded in any sequence. The only requirement is that the Local Area Network Support Program interrupt arbitrator must be loaded first.

Registration Process Overview

When a network interface program wants to register with the Local Area Network Support Program interrupt arbitrator, it should perform the following functions during its initialization sequence:

1. Test that the Local Area Network Support Program Interrupt Arbitrator Present flag (bit 5 at memory location 0040:00A1) is set. If the Local Area Network Support Program Interrupt Arbitrator Present flag is not set, the requesting interface will abort its load and print the appropriate error message (in English) indicating that the required software is not present.
2. Build a language request CCB (See "Language Request CCB" on page D-3 for the definition of this CCB).
3. Place the address of the language request CCB in registers ES and BX.
4. Execute an INT X'5C' instruction.
5. Save the returned language code index.
6. Test the CCB_RETCODE field.
   - If CCB_RETCODE is X'83', abort the load.
   - Otherwise continue the load sequence.
7. Build the appropriate network interrupt registration request CCB (see "Interface Registration CCB" on page D-4 and "NCB Handler Registration for NETBIOS using IEEE 802.2" on page D-6 for the definitions of these CCBs).
8. Place the address of the registration request CCB in registers ES and BX.
9. Execute an INT X'5C' instruction.
10. Test the return code field; if this field is non-zero, the registration request has failed.
    - If the registration request fails, the requesting program should print an error message to the standard output device, and abort the load. The error message displayed depends on the return code.
    - If the return code field is zero, the registration request has been accepted and the requesting program can continue its load sequence.

Language Request CCB

LANGUAGE REQUEST

Return Codes (CCB_RETCODE):

X'00'	Operation was completed successfully.
X'83'	Local Area Network Support Program Interrupt arbitrator load failed.

Table D-1. Interrupt Arbitrator Return Codes

Offset	Field Name	Byte Length	8086 Type	Description
0	CCB_ADAPTER	1	DB	Adapter 0 or 1
1	CCB_COMMAND	1	DB	X'31': Language Request
2	CCB_RETCODE	1	DB	Set by network arbitrator on return
3	CCB_LC	1	DB	Language code for messages *
4		2	DB	Reserved
6		4	DD	Address of the DXMA0MOD.SYS software interrupt entry point *
10		6	DB	Reserved

* Indicates a returned value (for example, a value set by the Local Area Network Support Program interrupt arbitrator).

The address of the DXMA0MOD.SYS software entry point is returned only by LAN Support Program Versions 1.3 and higher. This address may be called to pass CCBs and NCBs to DXMA0MOD.SYS rather than using an INT X'5C'. When you call this address, you must be sure ES:BX points to the CCB or NCB and that the calling application program simulates an interrupt by doing a PUSHF followed by a CALL FAR.

This entry point should not be called by any application program loaded after Microsoft Windows** has been loaded.

CCB_ADAPTER

Explanation: This command is adapter-independent. However, the value contained in this field must be less than X'04'.

CCB_LC

Explanation: The Local Area Network Support Program interrupt arbitrator uses this field to return the DOS country code to the registering interface. The registering interface should use this value to determine the language in which to print any error messages. See the *Local Area Network User's Guide* or the user's guide for DOS to determine the meaning of the values returned in this field.

Interface Registration CCB
REGISTRATION REQUEST

Return Codes (CCB_RETCODE):

X'00'	Operation was completed successfully
X'80'	CCB interface already registered for requested adapter
X'81'	NETBIOS already registered for requested adapter
X'83'	Local Area Network Support Program interrupt arbitrator load failed.

Table D-2. Interface Registration CCB Parameter Table

Offset	Field Name	Byte Length	8086 Type	Description
0	CCB_ADAPTER	1	DB	Adapter 0 or 1
1	CCB_COMMAND	1	DB	X'32': Network interrupt registration
2	CCB_RETCODE	1	DB	Return code*
3	CCB_WORK	1	DB	Reserved
4	5C_ENTRY	4	DD	@ to pass control for any CCB and NCBs
8	NCB_FLAGS	4	DD	@ of flags passed to NETBIOS is AL
12	CCB_PARM	1	DB	Defines type of registration request
13	CCB_CONFIG	1	DB	Interface configuration data
14	CCB_TRACE	2	DW	Offset to CCB trace routine

* Indicates a returned value (for example, a value set by the Local Area Network Support Program interrupt arbitrator).

@ Indicates an address throughout this document.

CCB_ADAPTER

Explanation: This field indicates which adapter the registration request is for. If an interface wants to register for both adapters, it must issue two separate registration requests.

5C_ENTRY

Explanation: This is the address to which the Local Area Network Support Program interrupt arbitrator passes control when it detects a CCB or NCB for the specified adapter. The Local Area Network Support Program interrupt arbitrator pushes all registers (except for AX) onto the stack and then transfers control to the specified address by a simulated INT instruction. The registers (except for AX) will be restored after the registered interface executes the IRET instruction to return control to the application program that issued the CCB or NCB.

The registers are pushed in the following order: BX, CX, DX, DI, SI, BP, ES, DS.

NCB_FLAGS

Explanation: This is the address of the 1-byte flag field that is to be passed to the NETBIOS interface that is registered for the same adapter number when an appropriate NETBIOS is detected. The byte pointed to by this field is loaded into the AL register before control is passed to NETBIOS. This field is only meaningful for a CCB interface registration. The contents of this field are ignored for a NETBIOS interface registration.

CCB_PARM

Explanation: This is a bit-significant field that is used to determine the type of registration request. The definition of the bits contained in this field are as follows:

Bit	**Meaning**
0	CCB interface (IEEE 802.2 interface) This bit indicates that the requesting interface is an IEEE 802.2 CCB interface.
1	NETBIOS (NETBIOS not using IEEE 802.2) This bit indicates that the requesting interface is a NETBIOS interface.
2-6	Reserved
7	Override Indicator This bit indicates that the requesting interface wants to override any currently registered interface on the specified adapter. An example of the use of this bit would be for the Local Area Network Support Program product. Since the Local Area Network Support Program converts the NCB interface on an original PC Network Adapter to an IEEE 802.2 CCB interface, the override bit would be required because the original PC Network Adapter would already be "auto-registered" as an NCB interface. This bit is only meaningful when used in conjunction with bit 1 of this field.

CCB_CONFIG

Explanation: This field is used to determine the configuration characteristics of the registering interface. For CCB interfaces, the contents of this field are passed to any registering NCB interfaces in the CCB_WORK field of the DEFINE.MIF command used to register the NCB interface. See "NCB Handler Registration for NETBIOS using IEEE 802.2" on page D-6 for the definition of the DEFINE.MIF command. The definition of the bits contained in this field are:

Bit	**Meaning**
0	3270 Workstation Program Supported This bit indicates that the requesting interface supports the 3270 Workstation Program with the XMA adapter. When this bit is set the Local Area Network Support Program interrupt arbitrator activates logic that provides the required environment data to the registered interface via the interface defined in "3270 Workstation Support" on page D-8.
1-7	Reserved

CCB_TRACE

Explanation: This field contains the offset portion of the address to the CCB trace routine. The segment portion of the address is assumed to be the same as the 5C_ENTRY segment.

This address allows the Local Area Network Support Program interrupt arbitrator to pass trace data to the registered CCB interface's trace routine for:

- NCBs passed to NETBIOS that is using the registered CCB interface.
- CCBs that are rejected by the Local Area Network Support Program interrupt arbitrator.

When the trace is active, the Local Area Network Support Program interrupt arbitrator will make a CALL FAR to the specified address with the registers set as follows:

- If the traced control block is an NCB, the register values are:

CH	Adapter number
CL	X'0F' (Trace ID)
DH	Return code
DL	Command code
SI	Number of bytes pushed on the stack since the INT X'5C' plus 4. (This includes the 4 bytes pushed on the stack during the Call Far.) This value is used to determine the location of the interrupt address on the stack.

- If the traced control block is a CCB, the register values are:

CH	X'00'
CL	Adapter number
DH	Return code
DL	Command code
SI	Number of bytes pushed on the stack since the INT X'5C' plus 4. (This includes the 4 bytes pushed on the stack during the Call Far.) This value is used to determine the location of the interrupt address on the stack.

NCB Handler Registration for NETBIOS using IEEE 802.2

NCB registration is done using a modified DIR.DEFINE.MIF.ENVIRONMENT command. The command is essentially the same as the DIR.DEFINE.MIF.ENVIRONMENT command with one exception:

The CCB_WORK field has been changed to the CCB_CONFIG field.

After the Local Area Network Support Program interrupt arbitrator has used the DEFINE.MIF command to register the interface with the Local Area Network Support Program interrupt arbitrator, the CCB is forwarded to the appropriate CCB interfaces for further processing.

NCB REGISTRATION

Return Codes (CCB_RETCODE):

X'00'	Operation was completed successfully
X'01'	Invalid command code
X'1B'	Invalid CCB_PARM_TAB pointer
X'1C'	Invalid pointer in the CCB parameter table
X'1D'	Invalid adapter number (no CCB interface registered)
X'82'	NCB interface already registered for requested adapter
X'84'	Duplicate CCB registration, PCNET is primary

Table D-3. NCB Registration Parameter Table

Offset	Field Name	Byte Length	8086 Type	Description
0	CCB_ADAPTER	1	DB	Adapter 0 or 1
1	CCB_COMMAND	1	DB	X'2B': DIR.DEFINE.MIF. ENVIRONMENT
2	CCB_RETCODE	1	DB	Command return (completion) code
3	CCB_CONFIG	1	DB	CCB interface configuration data *
4	CCB_POINTER	4	DD	Queue pointer and work area
8	CCB_CMD_CMPL	4	DD	Command completion user appendage
12	CCB_PARM_TAB	4	DD	Pointer to CCB parameter table

* Indicates a returned value (for example, a value set by the Local Area Network Support Program interrupt arbitrator).

CCB_CONFIG

Explanation: The Local Area Network Support Program interrupt arbitrator passes the configuration data from the registered CCB interface to the registering NCB interface in this field. The definition of the bits contained in this field are:

Bit **Meaning**

0 3270 Workstation Program supported

This bit indicates that the requesting interface supports the 3270 Workstation Program with the XMA adapter. When this bit is set, the Local Area Network Support Program interrupt arbitrator activates logic that provides the required environment data to the registered interface via the interface defined in “3270 Workstation Support” on page D-8.

1-7 Reserved

Table D-4. CCB Parameter Table Structure

Offset	Field Name	Byte Length	8086 Type	Description
0	NCB.INPUT@	4	DD	Address of NCB input module
4	NCB.OPEN@	4	DD	Address of the NCB open module
8	NCB.CLOSE@	4	DD	Address of the NCB Close module
12	NCB.ENABLE@	4	DD	Address of the interrupt enable module *

* Indicates a returned value (for example, a value set by the Local Area Network Support Program interrupt arbitrator).

NCB_INPUT@

Explanation: This is the address to which the Local Area Network Support Program interrupt arbitrator passes control when it detects an NCB for the specified adapter. The Local Area Network Support Program interrupt arbitrator pushes all registers (except for AX) onto the stack and then transfers control to the specified address by a simulated INT instruction. The registers (except for AX) will be restored after the registered interface executes the IRET instruction to return control to the application program that issued the NCB.

The registers are pushed in the following order: BX, CX, DX, DI, SI, BP, ES, DS.

NCB_ENABLE@

Explanation: It should be noted that the value returned in this field is from the CCB interface registered for adapter 1 if multiple CCB interfaces are registered with the Local Area Network Support Program interrupt arbitrator.

Additional Functions

Monitored Control Blocks: There are certain CCB_COMMAND values that are adapter independent. These commands are intended for all currently active CCB interfaces. These CCB_COMMAND values include:

X'2B' DIR.DEFINE.MIF.ENVIRONMENT

X'24' PDT.TRACE.ON

X'25' PDT.TRACE.OFF

The Local Area Network Support Program Interrupt Arbitrator monitors the CCB interface for the occurrence of one of these CCB_COMMANDS. When one of these commands is detected, the Local Area Network Support Program interrupt arbitrator propagates the CCB to all currently registered CCB interfaces. If the CCB_CMD_CMPL field is not zero, then the Local Area Network Support Program interrupt arbitrator gives control to the defined appendage after the CCB has been passed to all of the registered CCB interfaces.

3270 Workstation Support: The Local Area Network Support Program interrupt arbitrator supports the 3270 Workstation Program including bank swapping with the XMA adapter via the following interface.

When a CCB or NCB interface registers with the 3270 Workstation Program supported bit set (bit 0 in the CCB_IG field), the Local Area Network Support Program interrupt arbitrator will activate the following interfaces:

For CCB interfaces:

- The AX register will be used to indicate the currently active bank to the registered CCB interface.
  - AX = X'FFFF' indicates that the 3270 Workstation Program is not loaded, or bank swapping is not active so all programs are in common memory.
  - AX = X'00FF' indicates that the 3270 Workstation Program is loaded and bank swapping is active. However the "CCB in common memory" bit (bit 3 in the CCB_ADAPTER field) is set indicating that the control block is in common memory.
  - AX <> X'FFFF' or <> X'00FF' indicates that the 3270 Workstation Program is loaded and bank swapping is active. AX contains the current environment and bank ID. This information is in the same format that would be returned via an interrupt X'7A' function X'9D' request to the 3270 Workstation Program.

For NETBIOS interfaces:

- The first 2 bytes of the NCB reserved space (offset X'32') will be used to indicate the currently active bank to the registered NETBIOS interface.
  - NCB offset X'32' = X'FFFF' indicates that the 3270 Workstation Program is not loaded, or bank swapping is not active so all programs are in common memory.
  - NCB offset X'32' <> X'FFFF' indicates that the 3270 Workstation Program is loaded and bank swapping is active. Offset X'32' in the NCB/NCB contains the current environment and bank ID. This information is in the same format that would be returned via an interrupt X'7A' function X'9D' request to the control program.

Appendix E. Operating System/2 Extended Edition Information

About This Appendix E-2
CONFIG.SYS Commands E-2
  Adapter Parameters E-3
  DLC Parameters E-7
OS/2 EE NETBIOS Parameters E-11
OS/2 EE Trace Facility E-11
Trace Code Definition E-12
  Trace Entry Format E-12
  Trace Code Formats E-14
    CCB Trace Entry (Trace Codes X'00' through X'07') E-14
    CCB Completion Trace Entry (Trace Code X'08') E-15
    Receive Completion Trace Entry (Trace Code X'09') E-15
    Status/Exception Completion Trace Entry (Trace Code X'0A') E-16
    Interrupt Received Trace Entry (Trace Codes X'B' through X'0C') . . E-17
    Interrupt Error Trace Entry (Trace Code X'0D') E-18
OS/2 EE NETBIOS Trace Facility E-18

About This Appendix

This appendix contains information specific to OS/2 EE Version 1.3.

CONFIG.SYS Commands

The OS/2 System Initialization facility processes the configuration commands in the CONFIG.SYS file, establishing the final operating environment.

The CONFIG.SYS file contains commands used to configure the workstation. During workstation initialization, OS/2 opens and reads the CONFIG.SYS file and interprets the commands within the file. The adapter support software relies on the DEVICE command to install its device drivers (LANDD.SYS/TRNETDD.SYS/PCNETDD.SYS) and NETBIOS (NETBDD.SYS).

The DEVICE command is used to specify the path and file names of the adapter support software's device drivers so that they may be installed.

The keyword "cfg=" is provided on the DEVICE command line which is used to specify the path and file name of a configuration file. The configuration file is created by the configuration aid in OS/2 EE 1.3. If the parameter is not provided or the file is not found, a set of defaults will be used as described in the following section.

There is also a trace level parameter passed to the adapter support software with the DEVICE command to enable OS/2 workstation tracing for the adapter support software (see "OS/2 EE Trace Facility" on page E-11 for more information about the OS/2 trace). The keyword t=n (where n can be 2, 3, or 4) is provided on the DEVICE command line.

These CONFIG.SYS commands might appear as follows in the CONFIG.SYS file:

```
DEVICE=LANDD.SYS (Using NETBIOS or adapter support software Device Driver Interfa
DEVICE=TRNETDD.SYS CFG=pathname\filename.CFG t=n  (Token-Ring Network)
DEVICE=PCNETDD.SYS CFG=pathname\filename.CFG t=n  (PC Network)
DEVICE=ETHERDD.SYS CFG=pathname-filename.CFG t=n  (Ethernet)
DEVICE=NETBDD.SYS CFG=pathname\filename.CFG t=n
```

Notes:

1. These command statements are only examples; refer to *IBM Operating System/2 Extended Edition Version 1.1 Command Reference* for details on how these commands can be used. (Refer to *IBM Operating System/2 Extended Edition Version 1.2 Command Reference* if you are using Ethernet.)
2. The DEVICE=LANDD.SYS command must appear first in the CONFIG.SYS file.

   You need to add new commands to CONFIG.SYS for Ethernet support. Refer to *IBM Operating System/2 Extended Edition 1.1 System Administrator's Guide for Communications* for information about CONFIG.SYS commands.
3. The PC Network configuration and the Ethernet configuration are mutually exclusive. Only one of these configurations can be used at the same time.

The following list summarizes the global parameters that define the user's environment for each adapter.

- OPEN_OPTIONS (not used for PC Network and Ethernet support)

  Note: Some of these options can only be set by the application program.
- NODE_ADDRESS
- GROUP_ADDRESS (set only by the application program)
- FUNCTIONAL_ADDRESS (set only by the application program)
- NUMBER_RCV_BUFFERS
- RCV_BUFFER_LENGTH
- DHB_BUFFER_LENGTH
- DATA_HOLD_BUFFERS (not used for PC Network or Ethernet support)
- PRODUCT_ID
- SRAM_ADDRESS (Not used for PC Network or Ethernet support)
- DLC_MAX_SAP
- DLC_MAX_STATIONS
- DLC_MAX_GSAP
- DLC_MAX_GSAP_MEM
- DLC_T1_TICK_ONE
- DLC_T2_TICK_ONE
- DLC_Ti_TICK_ONE
- DLC_T1_TICK_TWO
- DLC_T2_TICK_TWO
- DLC_Ti_TICK_TWO
- SYSTEM_KEY
- MAX_USERS
- ADAPTER_WORK_SIZE (used only for PC Network and Ethernet adapters)
- QSIZE
- GDTSIZE
- NETWORK_FLAGS (Used only for Ethernet adapter)

None of the parameters are required, because defaults are used if the configuration file does not exist.

Adapter Parameters

OPEN_OPTIONS (applies to IBM Token-Ring Network Adapters)

Various options used when the adapter is physically opened. PC Network and Ethernet adapters ignore these parameters.

- Argument specified as a 2-byte character string hexadecimal number, for example X'2000'
- Character set supported (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F)
- Optional - Default is X'2000'
- Range - See the following option settings.

Various options, each represented by a bit where the bit set on (1) indicates that the option is to be taken. Bit 15 is the leftmost (high order) bit:

OPEN OPTION - Bit 15: WRAP INTERFACE
The adapter will not insert into the ring. Instead, it will cause all user transmit data to appear as received data.

OPEN OPTION - Bit 14: DISABLE HARD ERROR[*]
If this option is taken, the Network Status HARD ERROR and TRANSMIT BEACON interrupts will not occur.

OPEN OPTION - Bit 13: DISABLE SOFT ERROR[*]
If this option is taken, the Network Status SOFT ERROR interrupt will not occur.

OPEN OPTION - Bit 12: PASS ADAPTER MAC FRAMES[*]
Pass Adapter Class MAC frames which are received but not supported by the adapter as direct interface data to the workstation.

If this option is not taken, frames in this class will be rejected.

OPEN OPTION - Bit 11: PASS ATTENTION MAC FRAMES[*]
All Attention MAC frames that are not the same as the last received Attention MAC frame will be passed as direct interface data to the workstation.

If this option is not taken, these frames will not be passed to the workstation.

OPEN OPTION - Bits 9-10: Reserved
Should be set to 0, but are not checked.

OPEN OPTION - Bit 8: CONTENDER
The adapter will participate in Monitor Contention, if applicable.

OPEN OPTION - Bit 7: PASS BEACON FRAMES[*]
The adapter will pass beacon frames up to the attached processor.

OPEN OPTION - Bits 5-6: Reserved
Should be set to 0, but are not checked.

OPEN OPTION - Bit 4: TOKEN RELEASE
Setting this bit selects the non-default state of Token Release.

Table E-1. Token Release Table

Adapter Data Rate	Default Token Release	Non-Default Token Release
16 Mbps	Early Token Release	Regular Token Release

[*] These open options are set by the application program, not by the OS/2 configuration.

OPEN OPTION - Bits 0-3: Reserved
Should be set to 0, but are not checked.

NODE_ADDRESS
The 12-hexadecimal-digit specific node address of this station on the LAN. The 2 high-order bits of this value must be B'01'. If this value is coded as zero, then the permanently encoded address is used. If this value is coded as a non-zero value, then the coded value is used.

- Argument specified as a 6-byte character string hexadecimal number
- Optional - Default is permanently encoded address
- Recommended Range - X'400000000000' to X'400079999999'.

GROUP_ADDRESS (Set only by the application program)
This is the group address.

- Argument specified as a 4-byte character string hexadecimal number
- Optional - Default is X'00000000'
- Range - X'00000000' to X'FFFFFFFF'.

Group address for which the adapter will receive messages.

FUNCTIONAL_ADDRESS (Set only by the application program)
Functional address for which the adapter will receive messages.

Note that bits 0 and 31 are ignored.

- Argument specified as a 4-byte character string hexadecimal number
- Optional - Default is X'00000000'
- Range - X'00000000' to X'7FFFFFFE'.

NUMBER_RCV_BUFFERS
The number of receive buffers required for the adapter support software to initialize. The adapter will configure all RAM remaining after other storage requirements have been met as receive buffers.

If the number of receive buffers available is less than the number specified, a message is displayed to the user stating that there are inadequate receive buffers available for the adapter to open. Initialization will fail, and all commands will be rejected with CCB_RETCODE set to X'30'.

If the number of receive buffers available is greater than the number requested, no action is taken.

- Argument specified as a character string decimal number
- Optional - Default is 8 if value is less than 2 or not specified
- Range - 2 to upper boundary, determined by the amount of memory available in shared RAM.

RCV_BUFFER_LENGTH
The length of the adapter's receive buffers, which reside in the adapter's shared RAM. A receive buffer has 8 bytes of overhead. For example, when a buffer is 112 bytes, it can hold only 104 bytes of received data.

Receive buffers are allocated from the remaining shared RAM after all other requirements are satisfied.

- Argument specified as a character string decimal number

- Optional - Default is 112 if value is 0 or not specified
- Range - 96 to 2048 bytes in multiples of 8 bytes.

DHB_BUFFER_LENGTH

The length of the adapter's transmit data hold buffers (DHBs). With Token-Ring Network adapters, the DHBs reside in the adapter's shared RAM. With PC Network or Ethernet adapters, the DHB is located in workstation memory. A transmit buffer has 6 bytes of overhead. For example, when a buffer is 600 bytes, it can hold only 594 bytes of transmit data.

- Argument specified as a character string decimal number
- Optional - Default is 600 if value is 0 or not specified
- Range - 96 to 17960 bytes in multiples of 8 bytes.

  Note: It is the responsibility of the sending node to ensure that the size of the transmitted frame does not exceed the receive buffer capacity of the receiving node.

DATA_HOLD_BUFFERS (ignored by PC Network and Ethernet adapters)

This defines the number of adapter hold buffers, which contain transmit data passed from the workstation to the adapter. Although the adapter will accept any value from 1 to 255, the integrity of adapter operation cannot be guaranteed if the value is greater than 2.

The control information for the first two DATA_HOLD_BUFFERS is located in read-only memory, and is isolated from the application program's write operations. However, when more than 2 buffers are specified, the control information for the remaining buffers is located in read/write memory and is no longer isolated from the application program's write operations.

Requesting two DHBs may improve adapter performance by allowing a frame to be moved into the second DHB while the adapter is transmitting from the first. However, this reduces the storage available for receive buffers.

- Argument specified as a character string decimal number
- Optional - Default is 1 if value is not specified
- Range - 1 to 255.

PRODUCT_ID

This is the 18-byte product ID.

- Argument specified as a character string

  If less than 18 characters are specified, the product ID is padded with blanks on the right. This character string is provided for problem determination purposes and should describe the workstation attached to the network.

- Character set supported - All uppercase letters (A-Z) and decimal digits (0-9)

- Optional - Default is 18 blanks if a value is not specified.

SRAM_ADDRESS (Not used for PC Network or Ethernet support)

Defines the segment in workstation storage that contains the adapter's shared RAM. This value should be on a boundary that is equal to the size of shared RAM. For example, if shared RAM is 8 KB, the SRAM_ADDRESS should be on an 8 KB boundary, or if shared RAM is

16 KB, the SRAM_ADDRESS should be on a 16 KB boundary. When using 16/4 Token-Ring Network adapter shared RAM, paging can be used, which would allow the use of a 16 KB boundary. The SRAM_ADDRESS parameter is only used for a PC System with ISA Bus since there is no SETUP facility for these workstations.

- Argument specified as a 2-byte character string hexadecimal number, for example, X'D000'
- Optional - If value is 0 or not specified the defaults are
  - Adapter 0 (primary) - Segment value X'D800'
  - Adapter 1 (secondary) - Segment value X'D400'
- Range - Starting Address Range X'C00' to X'DE00' on the appropriate boundaries.

DLC Parameters

The parameter ranges that represent quantities (ranges for DLC_MAX_SAP, DLC_MAX_STATIONS, DLC_MAX_GSAP and DLC_MAX_GSAP_MEM parameters) are defined to ensure that values specified can be stored in the allotted memory sizes and do not represent the capabilities of the supported network adapters.

DLC_MAX_SAP

Indicates the maximum number of SAP stations that can be opened.

The maximum value of this parameter is 126. However, the maximum value allowed may be reduced if the amount of the available internal RAM work space or the amount of available adapter shared RAM is less than the value specified.

- Argument specified as a character string decimal number, for example, 6
- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 2 if value is not specified
- Range - 0 to 126 (upper limit is 125 for PC Network support).

DLC_MAX_STATIONS

Indicates the maximum number of link stations that can be opened.

The maximum value allowed may be reduced if the amount of available internal RAM work area or the amount of available Adapter Shared RAM is less than the value specified.

- Argument specified as a character string decimal number, for example, 6
- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 6 if value is not specified
- Range - 0 to 255.

DLC_MAX_GSAP

Indicates the maximum number of group SAPs that can be opened at one time.

If the value is 0, no group SAPs are allowed.

- Argument specified as a character string decimal number, for example, 2
- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 0 if value is not specified
- Range - 0 to 125 (upper limit is 124 for PC Network support).

DLC_MAX_GSAP_MEM
Indicates the maximum number of SAPs that can be members in any given Group.

- Argument specified as a character string decimal number, for example, 2
- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 0 if value is not specified
- Range - 0 to 126 (upper limit is 125 for PC Network support).

DLC_T1_TICK_ONE
Indicates the number of 40-ms intervals between timer ticks for DLC timer T1, timer values 1-5.

- Argument specified as a character string decimal number, for example, 2
- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 5 (200-400 ms) if value is 0 or not specified
- Range - 0 to 255.

DLC_T2_TICK_ONE
Indicates the number of 40-ms intervals between timer ticks for DLC timer T2, timer values 1-5.

- Argument specified as a character string decimal number, for example, 2
- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 1 (40-80 ms) if value is 0 or not specified
- Range - 0 to 255.

DLC_Ti_TICK_ONE
Indicates the number of 40-ms intervals between timer ticks for DLC timer Ti, timer values 1-5.

- Argument specified as a character string decimal number, for example, 20
- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 25 (1-2 seconds) if value is 0 or not specified
- Range - 0 to 255.

DLC_T1_TICK_TWO
Indicates the number of 40-ms intervals between timer ticks for DLC timer T1, timer values 6-10.

- Argument specified as a character string decimal number, for example, 20

- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 25 (1-2 seconds) if value is 0 or not specified
- Range - 0 to 255.

DLC_T2_TICK_TWO

Indicates the number of 40-ms intervals between timer ticks for DLC timer T2, timer values 6-10.

- Argument specified as a character string decimal number, for example, 100
- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 10 (400-800 ms) if value is 0 or not specified
- Range - 0 to 255.

DLC_Ti_TICK_TWO

Indicates the number of 40-ms intervals between timer ticks for DLC timer Ti, timer values 6-10.

- Argument specified as a character string decimal number, for example, 100
- Character set supported (0,1,2,3,4,5,6,7,8,9)
- Optional - Default is 125 (5-10 seconds) if value is 0 or not specified
- Range - 0 to 255.

SYSTEM_KEY

Is the system key code. It is used to enable only a system administrator to perform operations that could stop ring communication for application programs.

This key code restricts the following operations:

- Change functional address
- Change group addresses
- Reset selected or all SAPs and all stations
- Relinquish ownership of direct stations
- Force a physical close for an adapter
- Force the adapter to initialize
- Read and reset adapter error and direct interface logs.

The SYSTEM_KEY is not typically used by application programs. It is most often used for maintenance and problem determination operations.

The SYSTEM_KEY should be defined as follows:

- Argument specified as a 2-byte character string hexadecimal number
- Range - X'0001' to X'FFFF'.

MAX_USERS

Indicates the maximum number of users that can concurrently use the Token-Ring Network adapter. This parameter provides a mechanism that enables the adapter support software to determine the size of memory needed for its work area. If requests issued to adapter support software continue to fail with CCB_RETCODE set to X'59', either the number of MAX_USERS or the QSIZE parameter should be increased and the adapter support software restarted by re-IPLing the system.

- Argument specified as a character string decimal number
- Optional - Default is 3 if value is 0 or not specified
- Range - 0 to 5.

ADAPTER_WORK_SIZE (Used only for PC Network and Ethernet adapters)

Defines the size of memory allocated by the adapter support software during the initialization process. This memory is used to contain Link Station Control Blocks, SAP Control Blocks, Group SAP Control Blocks, and receive buffers.

- Argument specified as a character string decimal number.
- Optional - Default is X'2000'
- Range - X'2000' to X'FFFF'.

QSIZE

Specifies the number of internal queue elements that the adapter support software can allocate for all application programs. If no parameter is provided, the number of queue elements is calculated using the number of users (MAX_USER value). The number of queue elements allocated per user is 200. For example, if 5 users are defined to be the maximum number of users at one time, 1000 queue elements will be allocated.

- Argument specified as a character string decimal number
- Optional - Default is calculated using MAX_USER
- Range - 200 to 1400. The default is 200.

GDTSIZE

Specifies the number of Global Descriptor Table (GDT) selectors that the adapter support software can allocate for all application programs. If no parameter is specified, the number of GDT selectors is calculated using the number of users (MAX_USER), but cannot exceed 30. The number of GDT selectors allocated per user is 10. For example, if 2 users are defined to be the maximum number of users at one time, 20 GDT selectors will be allocated.

- Argument specified as a character string decimal number
- Optional - Default is calculated using MAX_USER
- Range - 10 to 30. The default is 20.

NETWORK_FLAGS (Used only for Ethernet adapter)

Defines the Ethernet protocol used for the adapter on Ethernet.

- Argument specified as a 1-byte character string hexadecimal number
- Optional - Default is X'00'
- The only possible values are X'00' for the DIX Version 2.0 protocol, or X'80' for the IEEE 802.3 protocol.

OS/2 EE NETBIOS Parameters

NETBIOS version 3.0 is the NETBIOS interface provided by OS/2 EE.

Table E-2. NETBIOS 3.0 System Default Values

Load Parameters	Abbr.	Valid Values	Default
STATIONS	ST	0 - 254	32 **
SESSIONS	S	0 - 254	32 **
COMMANDS	C	0 - 255	32
NAMES	N	0 - 254	17
TRANSMIT.TIMEOUT	TT	0 - 20	1
TRANSMIT.COUNT	TC	0 - 10	6
DLC.MAXOUT	MO	0 - 9	2
DLC.MAXIN	MI	0 - 9	1
DLC.RETRY.COUNT	RC	0 - 255	*
DLC.T1	T1	0 - 10	5
DLC.T2	T2	0 - 11	2
DLC.TI	TI	0 - 10	3
RING.ACCESS	RA	0 - 3	0
DATAGRAM.MAX	DG	Y(yes)/N(no)	NO
REMOTE.NAME.DIRECTORY	RND	0 - 255	0
REMOTE.DATAGRAM.CONTROL	RDC	Y(yes)/N(no)	NO
ADAP.ADDR.NOT.REVERSED	ANR	Y(yes)/N(no)	NO

* The default values are determined by the adapter and the adapter interface code.

** These values vary for OS/2 EE 1.3.

OS/2 EE Trace Facility

The adapter support software utilizes the OS/2 System Trace facilities. In order for the adapter support software to use the OS/2 System Trace facilities, the application program must provide a keyword "t=" on the "DEVICE=" command line of the CONFIG.SYS file when defining the adapter support software device drivers. The keyword is defined below.

The trace keyword "t=" must appear on the "DEVICE=" command line of the CONFIG.SYS file when defining the adapter support software's device drivers.

```
DEVICE=TRNETDD.SYS CFG=pathname filename.CFG t=n
```

The value of `n` can be 2, 3, or 4.

Value	Meaning
t=2	A value of 2 indicates the adapter support software will trace the following codes. See "Trace Code Definition" on page E-12 for a definition of what each trace code means. X'00' X'02' X'08' X'09' X'0A' X'0D'
t=3	A value of 3 indicates the adapter support software will trace the following codes. See "Trace Code Definition" for a definition of what each trace code means. X'00' X'02' X'08' X'09' X'0A' X'0B' X'0C' X'0D'
t=4	A value of 4 indicates the adapter support software will trace the following codes. See "Trace Code Definition" for a definition of what each trace code means. X'00' X'01' X'02' X'03' X'04' X'05' X'06' X'07' X'08' X'09' X'0A' X'0B' X'0C' X'0D'

Trace Code Definition

Trace Entry Format

Each trace entry is 16 bytes in length and each entry varies slightly depending upon the type of trace data (determined by the trace code).

The trace code in each trace entry identifies the point at which the trace entry was created and the format of the remainder of the trace entry. The trace codes are:

X'00' CCB command received.

X'01'	CCB command verified, queue element created.
X'02'	CCB command error.
X'03'	CCB command issued to the adapter (SRB) (not used by Token-Ring Network).
X'04'	CCB queue element added to receive queue.
X'05'	CCB queue element added to read queue.
X'06'	CCB queue element added to SRB wait queue.
X'07'	CCB queue element added to SSB wait queue.
X'08'	Completion of a CCB command or transmit command event occurred.
X'09'	Completion of a received frame event occurred.
X'0A'	Completion of a network status, DLC status, workstation action, or exception event occurred.
X'0B'	Adapter interrupt received.
X'0C'	Adapter timer expired.
X'0D'	Adapter interrupt error detected.

Trace Code Formats

CCB Trace Entry (Trace Codes X'00' through X'07')

Byte:

0	1	2	3	4	5	6	7	8-9	10-13	14-15
Trace Code	Adapter Number (0/1)	Adapter Work Flag 76543210	---	Command Code	RC/ Correlator	Element Flag	Appl ID	Process ID	CCB Virt Addr	Timer Ticks Lo Word

Byte 2: Adapter Work Flags:
- Bit 7: Adapter initialized successfully
- Bit 6: Adapter initialization in progress
- Bit 5: Adapter initialization failure
- Bit 4: Adapter opened successfully
- Bit 3: Adapter open in progress
- Bit 2: Adapter open failure
- Bit 1: Adapter reset in progress
- Bit 0: Adapter interrupt in progress

Byte 6: Element Flag:
- Bit 7: Chain transmits/receives by SAP
- Bit 6: Chain transmits/receives by link station
- Bit 5: Purge the link station
- Bit 4: Purge the SAP
- Bit 3: Purge the application program
- Bit 2: Not used
- Bit 1: No active CCB corresponding to this CCB queue element
- Bit 0: Completion flag set for this completion event

Bytes 10-13: CCB Virtual Address:
    Points to the corresponding application program CCB

Figure E-1. CCB Trace Entry

CCB Completion Trace Entry (Trace Code X'08')

Byte:

0	1	2	3	4	5	6	7	8-9	10-13	14-15
Trace Code	Adapter Number (0/1)	Adapter Work Flag 76543210	Event Code	Command Code	RC/ Correlator	Element Flag	Appl ID	Process ID	CCB Virt Addr	Timer Ticks Lo Word

Byte 3: Event Code:
- Bit 7: Not used
- Bit 6: Workstation action
- Bit 5: Network status
- Bit 4: Critical exception
- Bit 3: DLC status
- Bit 2: Receive
- Bit 1: Transmit
- Bit 0: CCB command complete

Figure E-2. CCB Completion Trace Entry

Receive Completion Trace Entry (Trace Code X'09')

Byte:

0	1	2	3	4-5	6	7	8-9	10-13	14-15
Trace Code	Adapter Number (0/1)	Adapter Work Flag 76543210	Event Code	---	Element Flag	Appl ID	Process ID	SAP Virt Addr	Timer Ticks Lo Word

Bytes 10-14: SAP Virtual Address:
Points to the corresponding received frame's first SAP buffer

Figure E-3. Receive Completion Trace Entry

Status/Exception Completion Trace Entry (Trace Code X'0A')

Byte:

0	1	2	3	4	5	6	7	8-9	10-13	14-15
Trace Code	Adapter Number (0/1)	Adapter Work Flag 76543210	Event Code	Event SubCode	---	Element Flag	Appl ID	Status	Status Data	Timer Ticks Lo Word

Byte 4: Event Subcode:

For workstation action event code:

X'01': DIR.CLOSE.ADAPTER
X'02': DIR.INITIALIZE
X'03': DLC.RESET for a single SAP station
X'04': DLC.RESET for all SAP stations

For critical exception event code:

X'01': Network status
X'02': Adapter check
X'03': PC error
X'04': Workstation action

Bytes 8-9: Status:

The 2-byte status code corresponding to the type of event
(network status code, DLC status code, adapter check code, workstation error code)

Bytes 10-13: Status Data:

The first 4 bytes of the data corresponding to the type of event

Figure E-4. Status Exception Completion Trace Entry

Interrupt Received Trace Entry (Trace Codes X'B' through X'0C')

Adapter Support Software Trace Entry:

Byte:

0	1	2	3	4	5	6-7	8-13	14-15
Trace Code	Adapter Number (0/1)	Adapter Work Flag 76543210	---	ISRP High	ISRP Low	Timer Count	---	Timer Ticks Lo Word

Byte 4: ISRP High:
- Bit 7: NMI disabled (always on)
- Bit 6: Interrupt enable (always on)
- Bit 5: Parity error in shared RAM
- Bit 4: Timer expired (100-ms programmable timer)
- Bit 3: Adapter check
- Bit 2: Shared RAM access violation
- Bit 1: Always on
- Bit 0: Adapter number (0 or 1)

Byte 5: ISRP Low:
- Bit 7: IMPL received
- Bit 6: Adapter check
- Bit 5: SRB response
- Bit 4: ASB free
- Bit 3: ARB command
- Bit 2: SSB response
- Bit 1: Always 0
- Bit 0: Always 0

Bytes 6-7: Timer Count:
Number of timer interrupts from adapters 0 and 1

Adapter Support Software Trace Entry:

Byte:

0	1	2	3	4	5	6-9	10-13	14-15
Trace Code	Adapter Number (0/1)	Adapter Work Flag 76543210	Activation Record Reason code	Command Code	Return Code	Command specific	SS:SP	Timer Ticks Lo Word

Figure E-5. Interrupt Received Trace Entry

Interrupt Error Trace Entry (Trace Code X'0D')

Byte:

0	1	2	3-13*	14-15
Trace Code	Adapter Number (0/1)	Adapter Work Flag 76543210	SRB/SSB/ ARB Work Area	Timer Ticks Lo Word

Bytes 3-13: SRB/SSB/ARB Work Area
First 11 bytes of the SRB/SSB/ARB that corresponds to the interrupt
(The format of the 11 bytes depends upon the command, interrupt, ...)

* These 11 bytes of work area are not used by the adapter support software.

Figure E-6. Interrupt Error Trace Entry

OS/2 EE NETBIOS Trace Facility

The NETBIOS major trace code is X'A4'. There are two minor trace codes: X'20' indicates an OS/2 DD interface level trace entry and X'21' indicates an OS/2 DLR interface level trace entry.

Further, receipt of NETBIOS protocol message "remote trace off" is not supported by OS/2 EE NETBIOS (the message is ignored).

The specific trace data in the system is shown in Table E-3.

Notes:

1. The basic information is similar to trace data in previous levels of NETBIOS. See the NCB.TRACE command in Chapter 4, "NETBIOS."
2. There is a new entry type: SYSTEM ACTION. This is indicated in the trace with the entry type and modifier X'EE04'.

Table E-3. NETBIOS Trace Table

Byte Offset	Usage
0	Adapter number indicator **X'AA'** Adapter 0 **X'BB'** Adapter 1
1	X'00' or an environment ID assigned by NETBIOS
2	Entry type
3	Entry modifier
4-7	Reserved
8-11	Type-dependent data
12-15	Address of data in bytes 16 through 31
16-31	Type-dependent data

Appendix F. NDIS Overview

About This Appendix

This appendix briefly describes the Network Driver Interface Specification (NDIS).

Description of NDIS

NDIS, developed jointly by 3Com and Microsoft Corporation, is a standardized interface for network adapter drivers and protocol drivers. NDIS separates protocol handling from hardware manipulation by defining functions that protocol drivers and network adapter drivers must provide to each other. It defines a configuration and binding process, which is handled by the protocol manager. NDIS has become a pseudo industry standard, providing a common, open interface that enables different manufacturers of network adapters and LAN software to communicate. Figure F-1 illustrates the major software components in the NDIS environment.

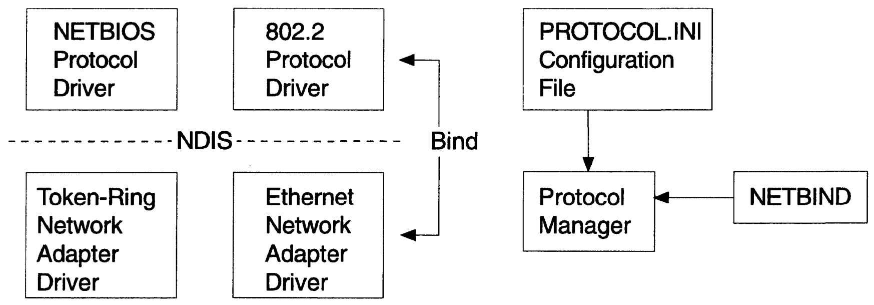

NETBIOS Protocol Driver
802.2 Protocol Driver
PROTOCOL.INI Configuration File
NDIS
Bind
Token-Ring Network Adapter Driver
Ethernet Network Adapter Driver
Protocol Manager
NETBIND

Figure F-1. Major Components in the NDIS Environment

A network adapter driver provides the communication between a network adapter and a protocol. The main function of the network adapter driver is to support network packet reception and transmission. A protocol driver provides the communication between an application and a network adapter driver. The main function of the protocol manager is to read configuration information from the configuration file, PROTOCOL.INI, and make it available to the network adapter drivers and protocol drivers. It also coordinates connections between protocol drivers and network adapter drivers. Before the protocol driver and the network adapter driver can communicate, they must be bound together (entry points are exchanged). NETBIND initiates the binding process based on information in PROTOCOL.INI.

List of Abbreviations

A. Address recognized bit in the frame status field of a Token-Ring frame.

AC. Access control.

A/C. Address recognized/frame copied.

ACDI. Asynchronous Communications Device Interface.

AIP. Adapter identification PROM.

API. Application program interface.

APPC/PC. Advanced Program-to-Program Communications/Personal Computers.

ARB. Adapter request block.

ASB. Adapter status block.

ASCII. American National Standard Code for Information Interchange.

BIOS. Basic Input/Output System.

CCB. Command Control Block.

CRC. Cyclic redundancy check.

CSMA/CD. Carrier sense multiple access with collision detection.

DB. Define byte.

DD. (1) Direct device interface. (2) Define doubleword.

DHB. Data holding buffer.

DISC. Disconnect character.

DIX. Digital Intel Xerox.

DLC. Data link control.

DLR. Dynamic Link Routine.

DM. Disconnect mode.

DMA. Direct memory access.

DOS. Disk Operating System.

DMA. Direct memory access.

DSAP. Destination service access point.

DW. Define word.

EBCDIC. Extended binary-coded decimal interchange code.

ED. Ending delimiter.

EFS. End frame sequence.

EHLLAPI. Emulator High-Level Language Application Programming.

EPROM. Erasable programmable read-only memory.

ES. Extended services.

ETR. Early Token Release.

FC. Frame control.

FCS. Frame check sequence.

FDDI. Fiber Distributed Data Interface

FIFO. First-in first-out.

FRMR. Frame reject.

FS. Frame status.

GDT. Global descriptor table.

GSAP. Group service access point.

I. Information (frame).

IEEE. Institute of Electrical and Electronics Engineers.

I/O. Input/output.

ISO. International Organization for Standardization.

ISRA. Interrupt status register, adapter.

ISRP. Interrupt status register, PC.

Kb. Kilobit.

KB. Kilobyte.

LAN. Local area network.

LAPS. Lan Adapter and Protocol Support.

LDT. Local descriptor table.

LLC. Logical link control.

LPDU. Logical link control protocol data unit.

LS. LAN Server.

LSAP. Local service access point.

LSN. Least significant nibble.

LSP. LAN support program.

LU. Logical unit.

MAC. Medium access control.

MB. Megabyte.

Mbps. Megabits per second.

MBps. Megabytes per second.

MGA. Multiple group addresses.

MMIO. Memory mapped input/output.

ms. Microsecond.

NAUN. Nearest active upstream neighbor.

NCB. Network Control Block.

NDIS. Network Driver Interface Specification.

NETBIOS. Network Basic Input/Output System.

ns. Nanosecond.

NTS/2. Network Transport Services/2.

ODI. Open data-link interface.

OEM. (1) Other equipment manufacturer. (2) Original equipment manufacturer.

OS/2. Operating System/2.

OS/2 EE. Operating System/2 Extended Edition.

PA. Preamble field of a Token-Ring frame.

PC. Personal computer.

PDU. Protocol data unit.

PIO. Programmed input/output.

POST. Power-On Self Test.

PROM. Programmable read only memory.

PS/2. Personal System/2.

PT. Physical trailer.

PU. Protocol unit.

RAM. Random access memory.

RI. Routing information field of the Token-Ring frame.

ROM. Read-only memory.

RPL. Remote program load.

RRR. RAM relocation register.

RR/RNR. Receiver ready/receiver not ready.

RSAP. Remote service access point.

SABME. Set asynchronous balanced mode extended.

SAP. Service access point.

SD. Starting delimiter field of a Token-Ring frame.

SDLC. Synchronous data link control.

SFS. Start Frame Sequence.

SNA. Systems Network Architecture.

SRAM. Shared random access memory.

SRB. System request block.

SRPI. Server-Requestor Programming Interface.

SRPR. Shared RAM paging register.

SSAP. Source service access point.

SSB. System status block.

TCR. Timer control register.

TVR. Timer value register.

UA. Unnumbered acknowledgment.

UI. Unnumbered information.

UPS. Uninterruptible power supply.

WRBR. Write region base management register.

WWCR. Write window close management register.

WWOR. Write window open management register.

X.25. Packet-switched networks.

XID. Exchange identification.

Glossary

This Glossary defines local area network terms and abbreviations. It includes terms and definitions from the *IBM Dictionary of Computing (Information Processing, Personal Computing, Telecommunications, Office Systems, IBM-Specific Terms)*, SC20-1699.

- The symbol (A) identifies definitions from the *American National Dictionary for Information Processing Systems*, copyright 1982 by the Computer and Business Equipment Manufacturers Association (CBEMA).
- The symbol (I) identifies definitions from the *ISO Vocabulary-Information Processing* and *ISO Vocabulary-Office Machines*, developed by the International Organization for Standardization, Technical Committee 97, Subcommittee 1.
- The symbol (T) identifies definitions from draft international standards, draft proposals, and working papers in development by the International Organization for Standardization, Technical Committee 97, Subcommittee 1.

This Glossary uses standard reference words for entries. They are:

Reference	Meaning
Deprecated term for	Indicates that the term should not be used (because it is obsolete, misleading, ambiguous, or jargonistic) and refers to the preferred term. For a deprecated term, the commentary contains only this reference; the deprecated term is not defined.
Synonymous with	Appears in the commentary of a preferred term and identifies less desirable or less specific terms that have the same meaning.
Synonym for	Appears in the commentary of a less desirable or less specific term and identifies the preferred term that has the same meaning. The less desired or less specific term is not defined.
Contrast with	Refers to a term that has an opposite or substantially different meaning.
See	Refers to terms in which this term appears.
See also	Refers to related terms that have similar (but not synonymous) meanings.

A

abort. To terminate, in a controlled manner, a processing activity in a computer system because it is impossible or undesirable for the activity to proceed. (T)

access priority. The maximum priority that a token can have for the adapter to use it for transmission.

access unit. A unit that allows multiple attaching devices access to a token-ring network at a central point such as a wiring closet or in an open work area.

active. (1) Able to communicate on the network. A token-ring network adapter is active if it is able to transmit and receive on the network. (2) Operational. (3) Pertaining to a node or device that is connected or is available for connection to another node or device. (4) Currently transmitting or receiving.

active monitor. A function in a single adapter on a token-ring network that initiates the transmission of tokens and provides token error recovery facilities. Any active adapter on the ring has the ability to provide the active monitor function if the current active monitor fails.

adapter. In a LAN, within a communicating device, a circuit card that, with its associated software and/or microcode, enables the device to communicate over the network.

adapter address. Twelve hexadecimal digits that identify a LAN adapter.

address. (1) In data communication, the IEEE-assigned unique code or the unique locally administered code assigned to each device or workstation connected to a network. (2) A character, group of characters, or a value that identifies a register, a particular part of storage, a data source, or a data sink. The value is represented by one or more characters. (T) (3) To refer to a device or an item of data by its address. (A) (4) The location in the storage of a computer where data is stored. (5) In word processing, the location, identified by the address code, of a specific section of the recording medium or storage. (T)

address segment. A section of IBM personal computer memory that can be up to 64 KB.

alert. (1) For IBM LAN management products, a notification indicating a possible security violation, a persistent error condition, or an interruption or potential interruption in the flow of data around the network. See also *network management vector transport*. (2) In SNA, a record sent to a system problem management focal point to communicate the existence of an alert condition. (3) In the NetView program, a high-priority event that warrants immediate attention. This data base record is generated for certain event types that are defined by user-constructed filters.

alternate adapter. In a personal computer that is used on a LAN and that supports installation of two network adapters, the adapter that uses alternate (not standard or default) mapping between adapter-shared RAM, adapter ROM, and designated computer memory segments. The alternate adapter is usually designated as adapter 1 in configuration parameters. Contrast with *primary adapter*.

analog. Pertaining to data consisting of continuously variable physical quantities. (A) Contrast with *digital*.

appendage. An application program routine that assists in handling the occurrence of specific events.

application program. (1) A program written for or by a user that applies to the user's work. Some application programs receive support and services from a special kind of application program called a network application program. (2) A program used to connect and communicate with stations in a network, enabling users to perform application-oriented activities.

application program interface (API). A functional interface supplied by the operating system or by a separately orderable licensed program that allows an application program written in a high-level language to use specific data or functions of the operating system or the licensed program.

architecture. A logical structure that encompasses operating principles including services, functions, and protocols. See *computer architecture, network architecture, Systems Application Architecture (SAA), Systems Network Architecture (SNA)*.

asynchronous. (1) Pertaining to two or more processes that do not depend upon the occurrence of a specific event such as a common timing signal. (T)
(2) In Fiber Distributed Data Interface (FDDI) rings, a type of data traffic that does not need bounded access delay to the medium and guaranteed throughput.

attach. To make a device a part of a network logically.

attaching device. Any device that is physically connected to a network and can communicate over the network.

automatic single-route broadcast. A function used by some IBM bridge programs to determine the correct settings for, and set the bridge single-route broadcast configuration parameters dynamically, without operator intervention. As bridges enter and leave the network, the parameter settings may need to change to maintain a single path between any two LAN segments for single-route broadcast messages. See also *single-route broadcast*.

auto-removal. The removal of a device from data-passing activity without human intervention. This action is accomplished by the adapter in the device, and can be initiated by a network management program.

available memory. In a personal computer, the number of bytes of memory that can be used after memory requirements for the operating system, device drivers, and other application programs have been satisfied.

B

baseband. (1) A frequency band that uses the complete bandwidth of a transmission medium. Contrast with *broadband, carrierband*. (2) A method of data transmission that encodes, modulates, and impresses information on the transmission medium without shifting or altering the frequency of the information signal.

baseband local area network. A local area network in which information is encoded, multiplexed, and transmitted without modulation of a carrier. (T)

Basic Input/Output System (BIOS). In IBM personal computers with PC I/O channel architecture, microcode that controls basic hardware operations such as interactions with diskette drives, fixed disk drives, and the keyboard.

beacon. (1) A frame sent by an adapter on a ring network indicating a serious ring problem, such as a broken cable. It contains the addresses of the beaconing station and its nearest active upstream neighbor (NAUN). (2) To send beacon frames continuously. An adapter is *beaconing* if it is sending such a frame.

beaconing. An error-indicating function of token-ring adapters that assists in locating a problem causing a hard error on a token-ring network.

binary. (1) Pertaining to a system of numbers to the base two; the binary digits are 0 and 1. (A)
(2) Pertaining to a selection, choice, or condition that has two possible different values or states. (I) (A)

bit. Either of the binary digits: a 0 or 1.

bit error rate (BER). The ratio of the number of bits experiencing error on a telecommunications link divided by the number of bits sent over the link.

bootstrap. (1) A sequence of instructions whose execution causes additional instructions to be loaded and executed until the complete computer program is in storage. (T) (2) A technique or device designed to bring itself into a desired state by means of its own action, for example, a machine routine whose first few instructions are sufficient to bring the rest of itself into the computer from an input device. (A)

bridge. (1) An attaching device that connects two LAN segments to allow the transfer of information from one LAN segment to the other. A bridge may connect the LAN segments directly by network adapters and software in a single device, or may connect network adapters in two separate devices through software and use of a telecommunications link between the two adapters. (2) A functional unit that connects two LANs that use the same logical link control (LLC) procedures but may use the same or different medium access control (MAC) procedures. (T) Contrast with *gateway* and *router*.

bridge ID. The bridge label combined with the adapter address of the adapter connecting the bridge to the LAN segment with the lowest LAN segment number; it is used by the automatic single-route broadcast function in IBM bridge programs.

broadband. (1) A frequency band between any two non-zone frequencies. (2) A frequency band divisible into several narrower bands so that different kinds of transmissions such as voice, video, and data transmission can occur at the same time. Synonymous with *wideband*. Contrast with *baseband*, *carrierband*.

broadband local area network (LAN). A local area network (LAN) in which information is encoded, multiplexed, and transmitted through modulation of a carrier. (T)

broadcast. Simultaneous transmission of data to more than one destination.

broadcast frame. A frame that is simultaneously transmitted to more than one destination. A broadcast frame is forwarded by all bridges, unless otherwise restricted.

buffer. (1) A portion of storage used to hold input or output data temporarily. (2) A routine or storage used to compensate for a difference in data rate or time of occurrence of events, when transferring data from one device to another. (A)

bus. (1) In a processor, a physical facility on which data is transferred to all destinations, but from which only addressed destinations may read in accordance with appropriate conventions. (I) (2) A network configuration in which nodes are interconnected through a bidirectional transmission medium. (3) One or more conductors used for transmitting signals or power. (A)

byte. (1) A string that consists of a number of bits, treated as a unit, and representing a character. (T) (2) A binary character operated upon as a unit and usually shorter than a computer word. (A) (3) A string that consists of a particular number of bits, usually 8, that is treated as a unit, and that represents a character. (4) A group of 8 adjacent binary digits that represent one extended binary-coded decimal interchange code (EBCDIC) character. See *n-bit byte*.

C

cable segment. A section of cable between components or devices on a network. A segment may consist of a single patch cable, multiple patch cables connected together, or a combination of building cable and patch cables connected together. See *LAN segment*, *ring segment*.

carrier. A wave or pulse train that may be varied by a signal bearing information to be transmitted over a communication system.

carrierband. A frequency band in which the modulated signal is superimposed on a carrier signal (as differentiated from baseband), but only one channel is present on the medium (as differentiated from broadband). Contrast with *baseband*, *broadband*.

carrier sense. In a local area network, an ongoing activity of a data station to detect whether another station is transmitting. (T)

channel. (1) A functional unit, controlled by a host computer, that handles the transfer of data between processor storage and local peripheral equipment. (2) A path along which signals can be sent. (3) The portion of a storage medium that is accessible to a given reading or writing station. (4) In broadband transmission, a designation of a frequency band 6 MHz wide.

circuit. (1) A logic device. (2) One or more conductors through which an electric current can flow.

collision. (1) An unwanted condition that results from concurrent transmissions on a channel. (T) (2) When a frame from a transmitting adapter encounters any other signal in its path (frame, noise, or another type of signal), the adapter stops transmitting and a collision is registered.

collision detection. In carrier sense multiple access with collision detection (CSMA/CD), a signal indicating that two or more stations are transmitting simultaneously.

command. (1) A request for performance of an operation or execution of a program. (2) A character string from a source external to a system that represents a request for system action.

completion code. The final return code provided by a program or adapter, as a result of an issued command, to indicate that an operation has ended.

component. (1) Any part of a network other than an attaching device, such as an IBM 8228 Multistation Access Unit. (2) Hardware or software that is part of a functional unit.

computer architecture. The organizational structure of a computer system, including hardware and software. (A)

configuration. (1) The arrangement of a computer system or network as defined by the nature, number, and chief characteristics of its functional units. More specifically, the term may refer to a hardware configuration or a software configuration. (I) (A) (2) The devices and programs that make up a system, subsystem, or network. See also *system configuration.*

configuration file. The collective set of definitions that describes a configuration.

configuration parameters. Variables in a configuration definition, the values of which characterize the relationship of a product, such as a bridge, to other products in the same network.

connect. In a LAN, to physically join a cable from a station to an access unit or network connection point. Contrast with *attach.*

contention. In a LAN, a situation in which two or more data stations are allowed by the protocol to start transmitting concurrently and thus risk collision. (T)

continuous carrier. On broadband networks, a condition in which a carrier signal is being constantly broadcast on a given frequency. No further information can be modulated on that frequency.

control block. (1) A storage area used by a computer program to hold control information. (I) (2) In the IBM Token-Ring Network, a specifically formatted block of information provided from the application program to the Adapter Support Interface to request an operation.

controller. A unit that controls input/output operations for one or more devices.

converter. In an IBM Token-Ring Network, a device that converts electronic signals to light pulses or vice versa for use in an optical fiber subsystem.

cyclic redundancy check (CRC). Synonym for *frame check sequence (FCS).*

D

data. (1) A representation of facts, concepts, or instructions in a formalized manner suitable for communication, interpretation, or processing by human or automatic means. (I) (A) (2) Any representations such as characters or analog quantities to which meaning is or might be assigned. (A)

datagram. A particular type of information encapsulation at the network layer of the adapter protocol. No explicit acknowledgment for the information is sent by the receiver. Instead, transmission relies on the *best effort* of the link layer.

data integrity. (1) The condition that exists as long as accidental or intentional destruction, alteration, or loss of data does not occur. (T) (2) Preservation of data for its intended use.

data link. (1) Any physical link, such as a wire or a telephone circuit, that connects one or more remote terminals to a communication control unit, or connects one communication control unit with another. (2) The assembly of parts of two data terminal equipment (DTE) devices that are controlled by a link protocol, and the interconnecting data circuit, that enable data to be transferred from a data source to a data sink. (I) (3) In SNA, see also *link.*
Note: A telecommunication line is only the physical medium of transmission. A data link includes the physical medium of transmission, the protocol, and associated devices and programs; it is both physical and logical.

data link control (DLC) protocol. The LAN protocol used to attach a device to and remove a device from the network. The DLC protocol is also used to send information onto and receive information from the network, exchange data, and control information with network higher level protocols and interfaces.

data network. An arrangement of data circuits and switching facilities for establishing connections between data terminal equipment. (I)

data packet. (1) At the interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE), a data unit used to transmit user data over a virtual circuit. (2) In an Open Systems Interconnection (OSI) network, a data unit passed between transport layer entities.

data rate. See *data transfer rate, line data rate.*

data transfer. (1) The result of the transmission of data signals from any data source to a data receiver. (2) The movement, or copying, of data from one location and the storage of the data at another location.

data transfer mode. The method of data transfer between a host computer and a channel-attached device. Data Channel Interlock (DCI) and data streaming are two data transfer modes.

data transfer rate. The average number of bits, characters, or blocks per unit of time passing between equipment in a data-transmission session. (I) The rate is expressed in bits, characters, or blocks per second, minute, or hour.

data transmission. The conveying of data from one place for reception elsewhere by means of telecommunications. (I)

default. Pertaining to an attribute, value, or option that is assumed when none is explicitly specified.

default value. A value assumed when no value has been specified.

delimiter. (1) A character used to indicate the beginning or end of a character string. (T) (2) A bit pattern that defines the beginning or end of a frame or token on a LAN.

destination. Any point or location, such as a node, station, or particular terminal, to which information is to be sent.

destination address. A field in the medium access control (MAC) frame that identifies the physical location to which information is to be sent. Contrast with *source address.*

device. (1) A mechanical, electrical, or electronic contrivance with a specific purpose. (2) An input/output unit such as a terminal, display, or printer. See also *attaching device.*

device driver. The code needed to attach and use a device on a computer or a network.

diagnostics. Modules or tests used by computer users and service personnel to diagnose hardware problems.

digital. (1) Pertaining to data in the form of digits. (A) Contrast with *analog.* (2) Pertaining to data consisting of numerical values or discrete units.

direct memory access (DMA). The transfer of data between memory and I/O units without processor intervention.

disabled. (1) Pertaining to a state of a processing unit that prevents the occurrence of certain types of interruptions. (2) Pertaining to the state in which a transmission control unit or audio response unit cannot accept incoming calls on a line.

disconnected mode. (1) In synchronous data link control (SDLC), a response from a secondary station indicating that it is disconnected and wants to be online. Synonym for *disconnected phase.*

disconnected phase. A phase entered by data circuit-terminating equipment (DCE) when it detects error conditions, recovers from a temporary internal malfunction, or receives a disconnect (DISC) command from data terminal equipment (DTE). In the disconnected phase, the DCE may initiate link setup but can transmit only disconnected-mode responses to received frames. Synonymous with *disconnected mode (2).*

Disk Operating System. An operating system for computer systems that use disks and diskettes for auxiliary storage of programs and data.

downstream. (1) On an IBM Token-Ring Network, the direction of data flow. (2) In the direction of data flow or toward the destination of transmission. Contrast with *upstream.*

E

Early Token Release (ETR). In token-ring and Fiber Distributed Data Interface (FDDI) networks, a function that allows a transmitting adapter to release a new token as soon as it has completed frame transmission, whether or not the frame header has returned to that adapter.

EBCDIC. Extended binary-coded decimal interchange code. A coded character set consisting of 8-bit coded characters. (A)

element. (1) In a set, an object, entity, or concept having the properties that define a set. (I) (A) (2) A parameter value in a list of parameter values.

emulation. (1) The imitation of all or part of one computer system by another, primarily by hardware, so that the imitating system accepts the same data, executes the same programs, and achieves the same results as the imitated computer system. (I) (A) (2) The use of programming techniques and special machine features to permit a computing system to execute programs written for another system.

enabled. (1) On a LAN, pertaining to an adapter or device that is active, operational, and able to receive frames from the network. (2) Pertaining to a state of a processing unit that allows the occurrence of certain types of interruptions. (3) Pertaining to the state in which a transmission control unit or an audio response unit can accept incoming calls on a line.

end delimiter. The last byte of a token or frame, consisting of a special, recognizable bit pattern.

enterprise. A business or organization that consists of two or more sites separated by a public right-of-way or a geographical distance. Contrast with *establishment*.

establishment. A user's premises that do not extend across public rights of way (for example, a single office building, warehouse, or campus). Contrast with *enterprise.*

Ethernet network. A baseband LAN with a bus topology in which messages are broadcast on a coaxial cable using a carrier sense multiple access/collision detection (CSMA/CD) transmission method. the completion of an asynchronous operation, such as an I/O operation.

exception. An abnormal condition such as an I/O error encountered in processing a data set or a file. See also *overflow exception* and *underflow exception.*

execute. To perform the actions specified by a program or a portion of a program. (T)

execution. The process of carrying out an instruction or instructions of a computer program by a computer. (I) (A)

exit. To execute an instruction or statement within a portion of a program in order to terminate the execution of that portion. (T)
Note: Such portions of programs include loops, routines, subroutines, and modules.

extended binary-coded decimal interchange code (EBCDIC). A coded character set consisting of 8-bit coded characters.

F

fault. An accidental condition that causes a functional unit to fail to perform its required function. (I) (A)

feature. A part of an IBM product that may be ordered separately by the customer.

Fiber Distributed Data Interface (FDDI). A high-performance, general-purpose, multi-station network designed for efficient operation with a peak data transfer rate of 100 Mbps. It uses token-passing protocol, similar in concept to token-ring architecture. It may use optical fiber as the transmission medium over distances of several kilometers, or it may use copper cable as the transmission medium over shorter distances.

field. On a data medium or a storage medium, a specified area used for a particular category of data; for example, a group of character positions used to enter or display wage rates on a panel. (T)

file. A named set of records stored or processed as a unit. (T)

file name. (1) A name assigned or declared for a file. (2) The name used by a program to identify a file.

first-in first-out (FIFO). A queuing technique in which the next item to be retrieved is the item that has been in the queue for the longest time. (A)

flag. A character or indicator that signals the occurrence of some condition, such as the setting of a switch, or the end of a word. (A)

frame. (1) The unit of transmission in some LANs, including the IBM Token-Ring Network and the IBM PC Network. It includes delimiters, control characters, information, and checking characters. On a token-ring network, a frame is created from a token when the token has data appended to it. On a token bus network (IBM PC Network), all frames including the token frame contain a preamble, start delimiter, control address, optional data and checking characters, end delimiter, and are followed by a minimum silence period. (2) A housing for machine elements. (3) In synchronous data link control (SDLC), the vehicle for every command, every response, and all information that is transmitted using SDLC procedures. Each frame begins and ends with a flag.

frame check sequence (FCS). (1) A system of error checking performed at both the sending and receiving station after a block check character has been accumulated. (2) A numeric value derived from the bits in a message that is used to check for any bit errors in transmission. (3) A redundancy check in which the check key is generated by a cyclic algorithm. (T) Synonymous with *cyclic redundancy check (CRC).*

frequency. The rate of signal oscillation, expressed in hertz (cycles per second).

function. (1) A specific purpose of an entity, or its characteristic action. (A) (2) In data communications, a machine action such as carriage return or line feed.

functional address. In IBM network adapters, a special kind of group address in which the address is bit-significant, each **On** bit representing a function performed by the station (such as *Active Monitor, Ring Error Monitor, LAN Error Monitor,* or *Configuration Report Server*).

G

gateway. A device and its associated software that interconnect networks or systems of different architectures. The connection is usually made above the reference model network layer. For example, a gateway allows LANs access to System/370 host computers. Contrast with *bridge* and *router.*

group. (1) A set of related records that have the same value for a particular field in all records. (2) A collection of users who can share access authorities for protected resources. (3) A list of names that are known together by a single name.

group address. In a LAN, a locally administered address assigned to two or more adapters to allow the adapters to copy the same frame. Contrast *locally administered address* with *universally administered address.*

group SAP. A single address assigned to a group of service access points (SAPs). See also *group address.*

H

hard error. An error condition on a network that requires that the source of the error be removed or that the network be reconfigured before the network can resume reliable operation. See also *beaconing.* Contrast with *soft error.*

hardware. Physical equipment as opposed to programs, procedures, rules, and associated documentation. (I) (A)

header. The portion of a message that contains control information for the message such as one or more destination fields, name of the originating station, input sequence number, character string indicating the type of message, and priority level for the message.

I

IBM Personal Computer Disk Operating System (DOS). A disk operating system based on MS-DOS.

inactive. (1) Not operational. (2) Pertaining to a node or device not connected or not available for connection to another node or device. (3) Pertaining to a station that is only repeating frames or tokens, or both.

individual address. An address that identifies a particular network adapter on a LAN. See also *locally administered address* and *universally administered address.*

initialize. In a LAN, to prepare the adapter (and adapter support code, if used) for use by an application program.

input/output (I/O). (1) Pertaining to a device whose parts can perform an input process and an output process at the same time. (I) (2) Pertaining to a functional unit or channel involved in an input process, output process, or both, concurrently or not, and to the data involved in such a process.

insert. To make an attaching device an active part of a LAN.

interface. (1) A shared boundary between two functional units, defined by functional characteristics, common physical interconnection characteristics, signal characteristics, and other characteristics as appropriate. (I) (2) A shared boundary. An interface may be a hardware component to link two devices or a portion of storage or registers accessed by two or more computer programs. (A) (3) Hardware, software, or both, that links systems, programs, or devices.

interrupt. (1) A suspension of a process, such as execution of a computer program, caused by an external event and performed in such a way that the process can be resumed. (A) (2) To stop a process in such a way that it can be resumed. (3) In data communication, to take an action at a receiving station that causes the sending station to end a transmission. (4) A means of passing processing control from one software or microcode module or routine to another, or of requesting a particular software, microcode, or hardware function.

interrupt level. The means of identifying the source of an interrupt, the function requested by an interrupt, or the code or feature that provides a function or service.

J

jumper. A connector between two pins on a network adapter that enables or disables an adapter option, feature, or parameter value.

L

LAN adapter. The circuit card within a communicating device (such as a personal computer) that, together with its associated software, enables the device to be attached to a LAN.

LAN multicast. The sending of a transmission frame intended to be accepted by a group of selected data stations on the same LAN.

LAN segment. (1) Any portion of a LAN (for example, a single bus or ring) that can operate independently but is connected to other parts of the establishment network via bridges. (2) An entire ring or bus network without bridges. See *cable segment*, *ring segment*.

latency. The time interval between the instant at which an instruction control unit initiates a call for data and the instant at which the actual transfer of data begins. Synonymous with *waiting time*. See also *ring latency*.

layer. (1) One of the seven levels of the Open Systems Interconnection reference model. (2) In open systems architecture, a collection of related functions that comprise one level of hierarchy of functions. Each layer specifies its own functions and assumes that lower level functions are provided. (3) In SNA, a grouping of related functions that are logically separate from the functions of other layers. Implementation of the functions in one layer can be changed without affecting functions in other layers.

limited broadcast. Synonym for *single-route broadcast*.

line data rate. The rate of data transmission over a telecommunications link.

link. (1) The logical connection between nodes including the end-to-end link control procedures.
(2) The combination of physical media, protocols, and programming that connects devices on a network.
(3) In computer programming, the part of a program, in some cases a single instruction or an address, that passes control and parameters between separate portions of the computer program. (I) (A) (4) To interconnect items of data or portions of one or more computer programs. (5) In SNA, the combination of the link connection and link stations joining network nodes.

link connection. (1) All physical components and protocol machines that lie between the communicating link stations of a link. The link connection may include a switched or leased physical data circuit, a LAN, or an X.25 virtual circuit. In SNA, the physical equipment providing two-way communication and error correction and detection between one link station and one or more other link stations.

link station. (1) A specific place in a service access point (SAP) that enables an adapter to communicate with another adapter. (2) A protocol machine in a node that manages the elements of procedure required for the exchange of data traffic with another communicating link station. (3) A logical point within a SAP that enables an adapter to establish connection-oriented communication with another adapter. (4) In SNA, the combination of hardware and software that allows a node to attach to and provide control for a link.

lobe. In the IBM Token-Ring Network, the section of cable (which may consist of several cable segments) that connects an attaching device to an access unit.

local area network (LAN). A computer network located on a user's premises within a limited geographical area.
Note: Communication within a local area network is not subject to external regulations; however, communication across the LAN boundary may be subject to some form of regulation. (T)

local busy. A state that may occur on a network for a given link station during which information-frame reception is suspended. This condition may occur because of an application program request, or a lack of buffers in the service access point (SAP) buffer pool.

locally administered address. An adapter address that the user can assign to override the universally administered address. Contrast with *universally administered address*.

local session number. The number assigned to each session established by an adapter. Each session receives a unique number that distinguishes it from any other active sessions.

logical connection. In a network, devices that can communicate or work with one another because they share the same protocol. See also *physical connection*.

logical link control protocol (LLC protocol). In a local area network, the protocol that governs the exchange of frames between data stations independently of how the transmission medium is shared. (T)

logical unit (LU). In SNA, a port through which an end user accesses the SNA network in order to communicate with another end user and through which the end user accesses the functions provided by system services control points (SSCPs). An LU can support at least two sessions, one with an SSCP and one with another LU, and may be capable of supporting many sessions with other logical units.

loop. A closed unidirectional signal path connecting input/output devices to a network.

M

MAC frame. Frames used to carry information to maintain the ring protocol and for exchange of management information.

MAC protocol. (1) In a local area network, the protocol that governs communication on the transmission medium without concern for the physical characteristics of the medium, but taking into account the topological aspects of the network, in order to enable the exchange of data between data stations. (T) See also *logical link control protocol (LLC protocol).* (2) The LAN protocol sublayer of data link control (DLC) protocol that includes functions for adapter address recognition, copying of message units from the physical network, and message unit format recognition, error detection, and routing within the processor.

macro. An instruction that causes the execution of a predefined sequence of instructions in the same source language.

medium. A physical carrier of electrical or optical energy.

medium access control (MAC) protocol. In a local area network, the part of the protocol that governs communication on the transmission medium without concern for the physical characteristics of the medium, but taking into account the topological aspects of the network, in order to enable the exchange of data between data stations. (T)

message. (1) A logical partition of the user device's data stream to and from the adapter. (2) A group of characters and control bits transferred as an entity.

Micro Channel. The architecture used by IBM Personal System/2 computers, Models 50 and above. This term is used to distinguish these computers from personal computers using a PC I/O channel, such as an IBM PC, XT, or an IBM Personal System/2 computer, Model 25 or 30.

microcode. (1) One or more microinstructions. (2) A code, representing the instructions of an instruction set, that is implemented in a part of storage that is not program-addressable. (3) To design, write, and also test one or more microinstructions.

monitor. (1) A functional unit that observes and records selected activities for analysis within a data processing system. Possible uses are to show significant departures from the norm, or to determine levels of utilization of particular functional units. (I) (A) (2) Software or hardware that observes, supervises, controls, or verifies operations of a system. (A)

Multicast address. See *LAN multicast.*

multitasking. (1) Pertaining to the concurrent execution of two or more tasks by a computer. (2) Multiprogramming that provides for the concurrent performance, or interleaved execution, of two or more tasks.

N

name. An alphanumeric term that identifies a data set, statement, program, or cataloged procedure.

n-bit byte. A string that consists of n bits. (T)

NDIS adapter. Any adapter that is supported by an NDIS MAC driver in addition to the LAN Support Program or the LAN Adapter and Protocol Support software. Ethernet adapters are examples. This term refers only to adapters using IBM adapter support software. See also *NDIS MAC driver* and *non-NDIS* adapter.

NDIS MAC driver. A device driver designed in accordance with NDIS that provides low-level access to network adapters.

network. (1) A configuration of data processing devices and software connected for information interchange. (2) An arrangement of nodes and connecting branches. Connections are made between data stations. (T)

network administrator. A person who manages the use and maintenance of a network.

network application program. A program used to connect and communicate with adapters on a network, enabling users to perform application-oriented activities and to run other application programs.

network architecture. The logical structure and operating principles of a computer network. (T) See also *systems network architecture (SNA)* and *Open Systems Interconnection (OSI) architecture.*
Note: The operating principles of a network include those of services, functions, and protocols.

Network Basic Input/Output System (NETBIOS). A message interface used on LANs to provide message, print server, and file server functions. The IBM NETBIOS application program interface (API) provides a programming interface to the LAN so that an application program can have LAN communication without knowledge and responsibility of the data link control (DLC) interface.

network management. The conceptual control element of a station that interfaces with all of the architectural layers of that station and is responsible for the resetting and setting of control parameters, obtaining reports of error conditions, and determining if the station should be connected to or disconnected from the network.

network management vector transport. The portion of an alert transport frame that contains the alert message.

network status. The condition of the network.

no carrier. On broadband networks, a condition in which a carrier signal is not being broadcast on a given frequency. In the absence of such a carrier, no information can be modulated on that frequency.

node. (1) Any device, attached to a network, that transmits and/or receives data. (2) An endpoint of a link, or a junction common to two or more links in a network. (3) In a network, a point where one or more functional units interconnect transmission lines.

node address. The address of an adapter on a LAN.

non-NDIS adapter. An adapter supported by IBM adapter support software only, without support from an NDIS MAC driver. An example is the IBM PC Network Adapter. All non-NDIS adapters are used with the CCB1 interface. See *NDIS adapter*.

O

office. See *work area*.

open. (1) To make an adapter ready for use. (2) A break in an electrical circuit. (3) To make a file ready for use.

Open Systems Interconnection (OSI). (1) The interconnection of open systems in accordance with specific ISO standards. (T) (2) The use of standardized procedures to enable the interconnection of data processing systems.
Note: OSI architecture establishes a framework for coordinating the development of current and future standards for the interconnection of computer systems. Network functions are divided into seven layers. Each layer represents a group of related data processing and communication functions that can be carried out in a standard way to support different applications.

Open Systems Interconnection (OSI) architecture. Network architecture that adheres to a particular set of ISO standards that relates to Open Systems Interconnection. (T)

Open Systems Interconnection (OSI) reference model. A model that represents the hierarchical arrangement of the seven layers described by the Open Systems Interconnection architecture.

operating system. Software that controls the execution of programs. An operating system may provide services such as resource allocation, scheduling, input/output control, and data management. (A) Examples are IBM PC DOS and IBM OS/2.

Operating System/2 (OS/2). A set of programs that control the operation of high-speed large-memory IBM personal computers (such as the IBM Personal System/2 computer, Models 50 and above), providing multitasking and the ability to address up to 16 MB of memory. Contrast with *Disk Operating System (DOS)*.

operation. (1) A defined action, namely, the act of obtaining a result from one or more operands in accordance with a rule that completely specifies the result for any permissible combination of operands. (A) (2) A program step undertaken or executed by a computer. (3) An action performed on one or more data items, such as adding, multiplying, comparing, or moving.

option. (1) A specification in a statement, a selection from a menu, or a setting of a switch, that may be used to influence the execution of a program. (2) A hardware or software function that may be selected or enabled as part of a configuration process. (3) A piece of hardware (such as a network adapter) that can be installed in a device to modify or enhance device function.

output device. A device in a data processing system by which data can be received from the system. (I) (A) Synonymous with *output unit*.

output unit. Synonym for *output device*.

overflow exception. A condition caused by the result of an arithmetic operation having a magnitude that exceeds the largest possible number. See also *underflow exception*.

P

packet. (1) In data communication, a sequence of binary digits, including data and control signals, that is transmitted and switched as a composite whole. (I) Synonymous with *data frame*. Contrast with *frame*.

page. (1) The portion of a panel that is shown on a display surface at one time. (2) To move back and forth among the pages of a multiple-page panel. See also *scroll*. (3) In a virtual storage system, a fixed-length block that has a virtual address and is transferred as a unit between real storage and virtual storage.

parameter. (1) A variable that is given a constant value for a specified application and that may denote the application. (I) (A) (2) An item in a menu or for which the user specifies a value or for which the system provides a value when the menu is interpreted.
(3) Data passed between programs or procedures.

parity (even). A condition when the sum of all of the digits in an array of binary digits is even.

parity (odd). A condition when the sum of all of the digits in an array of binary digits is odd.

path. (1) In a network, any route between any two nodes. (T) (2) The route traversed by the information exchanged between two attaching devices in a network. (3) A command in IBM Personal Computer Disk Operating System (PC DOS) and IBM Operating System/2 (OS/2) that specifies directories to be searched for commands or batch files that are not found by a search of the current directory.

PC Network. An IBM broadband or baseband LAN with a bus topology in which messages are broadcast from PC Network adapter to PC Network adapter.

personal computer (PC). A desk-top, free-standing, or portable microcomputer that usually consists of a system unit, a display, a monitor, a keyboard, one or more diskette drives, internal fixed-disk storage, and an optional printer. PCs are designed primarily to give independent computing power to a single user and are inexpensively priced for purchase by individuals or small businesses. Examples include the various models of the IBM Personal Computers, and the IBM Personal System/2 computer.

phase. The relative timing (position) of periodic electrical signals.

physical connection. The ability of two connectors to mate and make electrical contact. In a network, devices that are physically connected can communicate only if they share the same protocol. See also *logical connection*.

physical layer. In the Open Systems Interconnection reference model, the layer that provides the mechanical, electrical, functional, and procedural means to establish, maintain, and release physical connections over the transmission medium. (T)

pointer. (1) An identifier that indicates the location of an item of data. (A) (2) A data element that indicates the location of another data element. (T) (3) A physical or symbolic identifier of a unique target.

port. (1) An access point for data entry or exit. (2) A connector on a device to which cables for other devices such as display stations and printers are attached. Synonymous with *socket*.

post. (1) To affix to a usual place. (2) To provide items such as return code at the end of a command or function. (3) To define an appendage routine. (4) To note the occurrence of an event.

Power-On Self Test (POST). A series of diagnostic tests that are run automatically each time the computer's power is turned on.

primary adapter. In a personal computer that is used on a LAN and that supports installation of two network adapters, the adapter that uses standard (or default) mapping between adapter-shared RAM, adapter ROM, and designated computer memory segments. The primary adapter is usually designated as adapter 0 in configuration parameters. Contrast with *alternate adapter*.

PROCEDURE. A set of instructions that gives a service representative a step-by-step procedure for tracing a symptom to the cause of failure.

processor. In a computer, a functional unit that interprets and executes instructions. (I) (A)

protocol. (1) A set of semantic and syntactic rules that determines the behavior of functional units in achieving communication. (I) (2) In SNA, the meanings of and the sequencing rules for requests and responses used for managing the network, transferring data, and synchronizing the states of network components. (3) A specification for the format and relative timing of information exchanged between communicating parties.

protocol driver. A device driver that provides a higher-level communication interface between an application program and the interface defined by the adapter. For example, an 802.2 protocol driver provides the 802.2 interface at the top and the interface to the NDIS MAC driver at the bottom. See also *NDIS MAC driver*.

protocol handler. Hardware or microcode in an adapter that encodes and decodes the protocol used to format signals sent along a network.

R

RAM Paging. RAM Paging is a technique that allows the computer software to access all of the RAM on adapters that contain 64 KB of RAM, without having to map the entire shared RAM into the computer's memory map. The shared RAM on the adapter is paged into the computer's memory map one 16 KB page at a time.

RAM size. The amount of RAM that is directly mapped into the computer's memory map.

random access memory (RAM). A computer's or adapter's volatile storage area into which data may be entered and retrieved in a nonsequential manner.

read-only memory (ROM). A computer's or adapter's storage area whose contents cannot be modified by the user except under special circumstances.

receive. To obtain and store information transmitted from a device.

Reference Diskette. A diskette shipped with the IBM Personal System/2 computers with Micro Channel architecture. The diskette contains code and files used for configuration of options and for hardware diagnostic testing.

register. (1) A storage area in a computer's memory where specific data is stored. (2) A storage device having a specified storage capacity such as bit, byte, or computer word, and usually intended for a special purpose. (I)

remodulator. In broadband networks, an active device that demodulates inbound information and remodulates it on the higher frequency outbound channel. A remodulator may or may not provide frame error detection and does not amplify inbound noise distortion. It provides network clocking by broadcasting continuous idle when the inbound channel is not transmitting information. Contrast with *translator*.

remote program load. A function provided by adapter hardware components and software that enables one computer to load programs and operating systems into the memory of another computer, without requiring the use of a diskette or fixed disk at the receiving computer.

remove. (1) To take an attaching device off a network. (2) To stop an adapter from participating in data passing on a network.

return code. (1) A value (usually hexadecimal) provided by an adapter or a program to indicate the result of an action, command, or operation. (2) A code used to influence the execution of succeeding instructions. (A)

ring in (RI). In an IBM Token-Ring Network, the receive or input receptacle on an access unit or repeater.

ring latency. In an IBM Token-Ring Network, the time, measured in bit times at the data transmission rate, required for a signal to propagate once around the ring. Ring latency includes the signal propagation delay through the ring medium, including drop cables, plus the sum of propagation delays through each data station connected to the Token-Ring Network. (T)

ring network. A network configuration in which a series of attaching devices is connected by unidirectional transmission links to form a closed path. A ring of an IBM Token-Ring Network is referred to as a LAN segment or as a Token-Ring Network segment.

ring segment. A ring segment is any section of a ring that can be isolated (by unplugging connectors) from the rest of the ring. A segment can consist of a single lobe, the cable between access units, or a combination of cables, lobes, and/or access units. See *cable segment*, *LAN segment*.

ring station. A station that supports the functions necessary for connecting to the LAN and for operating with the token-ring protocols. These include token handling, transferring copied frames from the ring to the using node's storage, maintaining error counters, observing medium access control (MAC) sublayer protocols (for address acquisition, error reporting, or other duties), and (in the full-function native mode) directing frames to the correct data link control (DLC) link station.

ring status. The condition of the ring.

router. An attaching device that connects two LAN segments, which use similar or different architectures, at the reference model network layer. Contrast with *bridge* and *gateway*.

routine. Part of a program, or a sequence of instructions called by a program, that may have some general or frequent use.

routing. (1) The assignment of the path by which a message will reach its destination. (2) The forwarding of a message unit along a particular path through a network, as determined by the parameters carried in the message unit, such as the destination network address in a transmission header.

S

scroll. To move all or part of the display image vertically or horizontally to display data that cannot be observed within a single display image. See also *page (2)*.

segment. See *cable segment*, *LAN segment*, *ring segment*.

server. (1) A device, program, or code module on a network dedicated to providing a specific service to a network. (2) On a LAN, a data station that provides facilities to other data stations. Examples are a file server, print server, and mail server.

service access point (SAP). (1) A logical point made available by an adapter where information can be received and transmitted. A single SAP can have many links terminating in it. (2) In Open Systems Interconnection (OSI) architecture, the logical point at which an n + 1-layer entity acquires the services of the n-layer. For LANs, the n-layer is assumed to be data link control (DLC). A single SAP can have many links terminating in it. These link *end-points* are represented in DLC by link stations.

session. (1) A connection between two application programs that allows them to communicate. (2) In SNA, a logical connection between two network addressable units that can be activated, tailored to provide various protocols, and deactivated as requested. (3) The data transport connection resulting from a call or link between two devices. (4) The period of time during which a user of a node can communicate with an interactive system, usually the elapsed time between log on and log off. (5) In network architecture, an association of facilities necessary for establishing, maintaining, and releasing connections for communication between stations. (T)

session layer. In the Open Systems Interconnection reference model, the layer that provides the means necessary for two end users to organize and synchronize their dialogue and to manage their data exchange. (T)

shared RAM. Random access memory (RAM) on an adapter that is shared by the computer in which the adapter is installed.

signal. (1) A time-dependent value attached to a physical phenomenon for conveying data. (2) A variation of a physical quantity, used to convey data.

single-route broadcast. The forwarding of specially designated broadcast frames only by bridges which have single-route broadcast enabled. If the network is configured correctly, a single-route broadcast frame will have exactly one copy delivered to every LAN segment in the network. Synonymous with *limited broadcast.* See also *automatic single-route broadcast.*

socket. Synonym for *port (2).*

soft error. An intermittent error on a network that causes data to have to be transmitted more than once to be received. A soft error affects the network's performance but does not, by itself, affect the network's overall reliability. If the number of soft errors becomes excessive, reliability is affected. Contrast with *hard error.*

source address. A field in the medium access control (MAC) frame that identifies the location from which information is sent. Contrast with *destination address.*

start delimiter. The first byte of a token or frame, consisting of a special, recognizable bit pattern.

static RAM. In the IBM PC Network, random access memory on the adapter that is shared by the computer in which the adapter is installed.

station. (1) A communication device attached to a network. The term used most often in LANs is an *attaching device* or *workstation.* (2) An input or output point of a system that uses telecommunication facilities; for example, one or more systems, computers, terminals, devices, and associated programs at a particular location that can send or receive data over a telecommunication line. See also *attaching device, workstation.*

subvector. A subcomponent of the medium access control (MAC) major vector.

switch. On an adapter, a mechanism used to select a value for, enable, or disable a configurable option or feature.

synchronous. (1) Pertaining to two or more processes that depend on the occurrences of a specific event such as common timing signal. (I) (A)
(2) Occurring with a regular or predictable timing relationship.

synchronous data link control (SDLC). A discipline conforming to subsets of the Advanced Data Communication Control Procedures (ADCCP) of the American National Standards Institute (ANSI) and High-level Data Link Control (HDLC) of the International Organization for Standardization, for managing synchronous, code-transparent, serial-by-bit information transfer over a link connection. Transmission exchanges may be duplex or half-duplex over switched or nonswitched links. The configuration of the link connection may be point-to-point, multipoint, or loop. (I)

system. In data processing, a collection of people, machines, and methods organized to accomplish a set of specific functions. (I) (A)

system configuration. A process that specifies the devices and programs that form a particular data processing system.

Systems Application Architecture (SAA). An architecture developed by IBM that consists of a set of selected software interfaces, conventions, and protocols, and that serves as a common framework for application development, portability, and use across different IBM hardware systems.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks.
Note: The layered structure of SNA allows the ultimate origins and destinations of information, that is, the end users, to be independent of and unaffected by the specific SNA network services and facilities used for information exchange.

system unit. (1) A part of a computer that contains the processing unit, and may contain devices such as disk and diskette drives. (2) In an IBM personal computer, the unit that contains the processor circuitry, ROM, RAM, and the I/O channel. It may have one or more disk or diskette drives.

T

terminal. In data communication, a device, usually equipped with a keyboard and display device, capable of sending and receiving information.

threshold. (1) A level, point, or value above which something is true or will take place and below which it is not true or will not take place. (2) In IBM bridge programs, a value set for the maximum number of frames that are not forwarded across a bridge due to errors, before a *threshold exceeded* occurrence is counted and indicated to network management programs. (3) An initial value from which a counter is decremented to zero, or a value to which a counter is incremented or decremented from an initial value. When the counter reaches zero or the threshold value, a decision is made and/or an event occurs.

time-out. (1) A parameter related to an enforced event designed to occur at the conclusion of a predetermined elapsed time. A time-out condition can be canceled by the receipt of an appropriate time-out cancellation signal. (2) A time interval allotted for certain operations to occur; for example, response to polling or addressing before system operation is interrupted and must be restarted.

token. A sequence of bits passed from one device to another on the token-ring network that signifies permission to transmit over the network. It consists of a starting delimiter, an access control field, and an end delimiter. The access control field contains a bit that indicates to a receiving device that the token is ready to accept information. If a device has data to send along the network, it appends the data to the token. When data is appended, the token then becomes a frame. See *frame*.

token ring. A network with a ring topology that passes tokens from one attaching device (node) to another. A node that is ready to send can capture a token and insert data for transmission.

token-ring network. (1) A ring network that allows unidirectional data transmission between data stations by a token-passing procedure over one transmission medium so that the transmitted data returns to and is removed by the transmitting station. (T) The IBM Token-Ring Network is a baseband LAN with a star-wired ring topology that passes tokens from network adapter to network adapter. (2) A network that uses a ring topology, in which tokens are passed in a sequence from node to node. A node that is ready to send can capture the token and insert data for transmission. (3) A group of interconnected token rings.

trace. (1) A record of the execution of a computer program. It exhibits the sequences in which the instructions were executed. (2) A record of the frames and bytes transmitted on a network.

transaction. In an SNA network, an exchange between two programs that usually involves a specific set of initial input data that causes the execution of a specific task or job. Examples of transactions include the entry of a customer's deposit that results in the updating of the customer's balance, and the transfer of a message to one or more destination points.

transceiver. Any device that can transmit and receive traffic.

translator. In broadband networks, an active device for converting an inbound channel to a higher frequency outbound channel. The conversion is done by removing the inbound carrier, adding the outbound carrier, and amplifying the signal. (A translator amplifies inbound errors and noise distortion.) Contrast with *remodulator*.

transmission frame. See *frame*.

transmit. To send information from one place for reception elsewhere.

transmitter. (1) A circuit used in data communication applications to send information from one place for reception elsewhere. (2) The device in which the transmission circuits are housed.

U

underflow exception. A condition caused by the result of an arithmetic operation having a magnitude less than the smallest possible nonzero number.

universally administered address. The address permanently encoded in an adapter at the time of manufacture. All universally administered addresses are unique. Contrast with *locally administered address*.

unnumbered acknowledgment. A data link control (DLC) command used in establishing a link and in answering receipt of logical link control (LLC) frames.

upstream. On an IBM Token-Ring Network, the direction opposite that of data flow. Contrast with *downstream*.

usability. The quality of a system, program, or device that enables it to be easily understood and conveniently employed by a user.

V

variable. (1) In computer programming, a character or group of characters that refers to a value and, in the execution of a computer program, corresponds to an address. (2) A quantity that can assume any of a given set of values. (A)

vector. One or more related fields of data, in a specified format. A quantity usually characterized by an ordered set of numbers. (I) (A)

version. A separate IBM-licensed program, based on an existing IBM-licensed program, that usually has significant new code or new function.

W

waiting time. Synonym for *latency*.

wideband. Synonym for *broadband*.

wire fault. An error condition caused by a break in the wires or a short between the wires (or shield) in a segment of cable.

work area. An area in which terminal devices (such as displays, keyboards, and printers) are located. Access units may also be located in work areas.

working diskette. A diskette to which files are copied from an original diskette for use in everyday operation to protect the original diskette from damage.

workstation. (1) An I/O device that allows either transmission of data or the reception of data (or both) from a host system, as needed to perform a job: for example, a display station or printer. (2) A configuration of I/O equipment at which an operator works. (T) (3) A terminal or microcomputer, usually one connected to a mainframe or network, at which a user can perform tasks.

Index

Numerics

3270 support, Workstation Program D-8

A

adapter 6-6
  address 2-26
  application program interfaces 1-2
  check
    CCB1 B-46
    CCB2 B-47
    CCB3 B-49
    reason codes, IBM Token-Ring Network B-51
    reason codes, PC network and Ethernet network B-52
  log formats 3-36
  NDIS 1-2
  non-NDIS 1-2
  open errors
    PC and Ethernet B-63
    Token-Ring network B-58
  protocol drivers 1-2
  support programs 1-4
Adapter Check
  0x46 EXCEPTION EVENT 6-14
  in NCB_RESERVE field 6-5, 6-7
  minor code for NETBIOS Trace 6-18
  on NCB.STATUS command 6-5, 6-7
  OS2TRACEMASK PARAMETER 6-19
  TRACE Parameter 6-16
Adapter Status 6-8
adapter type identification in NETBIOS 4-38
ADAPTER_WORK_SIZE E-10
adapters supported by LAPS 6-3
address
  adapter node 2-26
  service access point (SAP) 2-27
  station ID 2-27
appendages, adapter support software 2-4

B

bridge
  parallel 1-16
  setting length bits 1-13
BRING_UPS parameter 3-13, B-57
broadcast
  broadcast indicators 1-12
broadcast address 3-42
broadcast indicators 1-12
buffer
  pool 2-40

buffer *(continued)*
  receive
    field explanations 2-45
    formats 2-42
  transmit
    formats 2-47
    overview 2-46
BUFFER.FREE 3-4
BUFFER.GET 3-5
Busmaster Server Adapter 6-8

C

calling conventions
  DD interface (CCB3) 2-13
  DLR interface (CCB2) 2-8
  LAN Support Program (CCB1) 2-3
  NETBIOS 4-14
canonical bit ordering 2-26
CCB (command control block)
  CCB1 2-6
  CCB2 (DLR) 2-12
  CCB3 (DD) 2-18
  command tables
    DLC and Direct Interface A-1
    NETBIOS A-3
  commands
    See commands, adapter support
  field explanations 2-19
  return codes
    DLC and direct interface B-2
    DLC and direct interface by command B-5
    DLC and direct interface cause and action B-14
    NCB (NETBIOS) B-32
    NETBIOS by command B-34
    NETBIOS cause and action B-35
CCB interface
  DIR.RESET.MULT.GROUP.ADDRESS 3-37, 3-41
CCB1
  adapter check B-46
  command return codes B-5
  network status B-52
  PC system detected errors B-63
CCB2 6-13—6-15
  adapter check B-47
  command return codes B-8
  network status B-52
  PC system detected errors B-64
  system action exceptions B-69
CCB3 6-13—6-15
  adapter check B-49
  command return codes B-11

CCB3 *(continued)*
  network status B-54
  PC system detected errors B-67
  system action exceptions B-72
command completion
  DD interface (CCB3) 2-14
  DLR interface (CCB2) 2-9
  LAN Support Program (CCB1) 2-3
command control block (CCB)
  CCB1 2-6
  CCB2 (DLR) 2-12
  CCB3 (DD) 2-18
  command tables
    DLC and Direct Interface A-1
    NETBIOS A-3
  commands
    See commands, adapter support
  field explanations 2-19
  return codes
    DLC and direct interface B-2
    DLC and direct interface by command B-5
    DLC and direct interface cause and action B-14
    NCB (NETBIOS) B-32
    NETBIOS by command B-34
    NETBIOS cause and action B-35
commands, adapter support
  BUFFER.FREE 3-4
  BUFFER.GET 3-5
  DIR.CLOSE.ADAPTER 3-6
  DIR.CLOSE.DIRECT 3-8
  DIR.DEFINE.MIF.ENVIRONMENT 3-9
  DIR.INITIALIZE 3-11
  DIR.INTERRUPT 3-16
  DIR.MODIFY.OPEN.PARMS 3-16
  DIR.OPEN.ADAPTER 3-17
  DIR.OPEN.DIRECT 3-33
  DIR.READ.LOG 3-34
  DIR.RESTORE.OPEN.PARMS 3-37
  DIR.SET.EXCEPTION.FLAGS 3-38
  DIR.SET.FUNCTIONAL.ADDRESS 3-39
  DIR.SET.GROUP.ADDRESS 3-41
  DIR.SET.USER.APPENDAGE 3-43
  DIR.STATUS 3-44
  DIR.TIMER.CANCEL 3-51
  DIR.TIMER.CANCEL.GROUP 3-51
  DIR.TIMER.SET 3-52
  DLC.CLOSE.SAP 3-53
  DLC.CLOSE.STATION 3-54
  DLC.CONNECT.STATION 3-55
  DLC.FLOW.CONTROL 3-56
  DLC.MODIFY 3-57
  DLC.OPEN.SAP 3-60
  DLC.OPEN.STATION 3-66
  DLC.REALLOCATE 3-70
  DLC.RESET 3-71
  DLC.SET.THRESHOLD 3-73
commands, adapter support *(continued)*
  DLC.STATISTICS 3-74
  PDT.TRACE.OFF 3-76
  PDT.TRACE.ON 3-77
  PURGE.RESOURCES 3-85
  READ 3-86
  READ.CANCEL 3-94
  RECEIVE 3-95
  RECEIVE.CANCEL 3-102
  RECEIVE.MODIFY 3-103
  TRANSMIT.DIR.FRAME 3-107
  TRANSMIT.I.FRAME 3-108
  TRANSMIT.TEST.CMD 3-109
  TRANSMIT.UI.FRAME 3-109
  TRANSMIT.XID.CMD 3-109
  TRANSMIT.XID.RESP.FINAL 3-110
  TRANSMIT.XID.RESP.NOT.FINAL 3-110
commands, NETBIOS
  NCB.ADD.GROUP.NAME 4-15
  NCB.ADD.NAME 4-16
  NCB.CALL 4-16
  NCB.CANCEL 4-17
  NCB.CHAIN.SEND 4-18
  NCB.CHAIN.SEND.NO.ACK 4-19
  NCB.DELETE.NAME 4-20
  NCB.FIND.NAME 4-21
  NCB.HANG.UP 4-22
  NCB.LAN.STATUS.ALERT 4-23
  NCB.LISTEN 4-24
  NCB.RECEIVE 4-25
  NCB.RECEIVE.ANY 4-26
  NCB.RECEIVE.BROADCAST.DATAGRAM 4-27
  NCB.RECEIVE.DATAGRAM 4-27
  NCB.RESET 4-28
  NCB.SEND 4-32
  NCB.SEND.BROADCAST.DATAGRAM 4-33
  NCB.SEND.DATAGRAM 4-34
  NCB.SEND.NO.ACK 4-34
  NCB.SESSION.STATUS 4-35
  NCB.STATUS 4-37
  NCB.TRACE 4-41
  NCB.UNLINK 4-45
Communications Manager, O/S 2
  device driver (DD) interface 1-5
  dynamic link routine (DLR) interface 1-5
  interface comparison 1-6
config.sys 6-10
CONFIG.SYS commands (OS/2 EE) E-2
  ADAPTER_WORK_SIZE E-10
  DATA_HOLD_BUFFERS E-6
  DHB_BUFFER_LENGTH E-6
  DLC_MAX_GSAP E-7
  DLC_MAX_GSAP_MEM E-8
  DLC_MAX_SAP E-7
  DLC_MAX_STATIONS E-7
  DLC_T1_TICK_ONE E-8

CONFIG.SYS commands (OS/2 EE) *(continued)*
  DLC_T1_TICK_TWO E-8
  DLC_T2_TICK_ONE E-8
  DLC_T2_TICK_TWO E-9
  DLC_Ti_TICK_ONE E-8
  DLC_Ti_TICK_TWO E-9
  FUNCTIONAL_ADDRESS E-5
  GDTSIZE E-10
  GROUP_ADDRESS E-5
  MAX_USERS E-9
  NETWORK_FLAGS E-10
  NODE_ADDRESS E-5
  NUMBER_RCV_BUFFERS E-5
  OPEN_OPTIONS E-3
  PRODUCT_ID E-6
  QSIZE E-10
  RCV_BUFFER_LENGTH E-5
  SRAM_ADDRESS E-7
  SYSTEM_KEY E-9
configuration 6-2, F-1

D

data link control interface
  command sequence 2-31
  link station states 2-33
  network adapter parameters 2-36
  status change events 3-94
  timers 2-35
  types of service 2-31
data segment guidelines
  DD interface (CCB3) 2-13
  DLR interface (CCB2) 2-8
data segment restrictions
  DD interface (CCB3) 2-12
  DLR interface (CCB2) 2-7
data transmission
  between application programs 2-39
  Ethernet (DIX Version 2.0)
  IBM PC networks
  IBM Token-Ring network
  IEEE 802.3 networks
DATA_HOLD_BUFFERS E-6
DEF file 6-3, 6-4
definition of the IEEE 802.2 API for OS/2 6-3
definition of the NETBIOS API for OS/2 6-4
device driver (DD) interface 1-5
device driver interface 6-3, 6-5
device drivers
  See drivers
DHB_BUFFER_LENGTH E-6
DIR.CLOSE.ADAPTER 3-6
DIR.CLOSE.DIRECT 3-8
DIR.DEFINE.MIF.ENVIRONMENT 3-9
DIR.INITIALIZE 3-11, 6-6
DIR.INTERRUPT 3-16
DIR.MODIFY.OPEN.PARMS 3-16
DIR.OPEN.ADAPTER 3-17, 6-7
DIR.OPEN.DIRECT 3-33
DIR.READ.LOG 3-34, 6-7
DIR.RESET.MULT.GROUP.ADDRESS 3-37
DIR.RESTORE.OPEN.PARMS 3-37
DIR.SET.EXCEPTION.FLAGS 3-38
DIR.SET.FUNCTIONAL.ADDRESS 3-39, 6-7
DIR.SET.GROUP ADDRESS 3-41
DIR.SET.MULT.GROUP.ADDRESS 3-41
DIR.SET.USER.APPENDAGE 3-43
DIR.STATUS 3-44, 6-7, 6-8
DIR.TIMER.CANCEL 3-51
DIR.TIMER.CANCEL.GROUP 3-51
DIR.TIMER.SET 3-52
direct interface log format 3-36
direction bit 1-13
diskette, sample LAN program listing C-1
DIX Version 2.0 network
  frame field descriptions 1-17
  frames 1-19
DLC interface
  command sequence 2-31
  link station states 2-33
  network adapter parameters 2-36
  status change events 3-94
  timers 2-35
  types of service 2-31
DLC status 6-13—6-20
  codes B-30
  suggested actions B-31
  table B-29
DLC_MAX_GSAP E-7
DLC_MAX_GSAP_MEM E-8
DLC_MAX_SAP E-7
DLC_MAX_STATIONS E-7
DLC_T1_TICK_ONE E-8
DLC_T1_TICK_TWO E-8
DLC_T2_TICK_ONE E-8
DLC_T2_TICK_TWO E-9
DLC_Ti_TICK_ONE E-8
DLC_Ti_TICK_TWO E-9
DLC.CLOSE.SAP 3-53
DLC.CLOSE.STATION 3-54
DLC.CONNECT.STATION 3-55
DLC.FLOW.CONTROL 3-56
DLC.MODIFY 3-57
DLC.OPEN.SAP 3-60
DLC.OPEN.STATION 3-66
DLC.REALLOCATE 3-70
DLC.RESET 3-71
DLC.SET.THRESHOLD 3-73
DLC.STATISTICS 3-74
DOS NETBIOS 6-2, 6-8

drivers, NDIS MAC 1-2
drivers, protocol 1-2
dynamic link interface 6-3—6-4
dynamic link routine (DLR) interface 1-5

E

errors
  adapter open (PC network and Ethernet) B-63
  adapter open (Token-Ring network)
    codes B-58
    recommended action table B-63
    suggested action B-59
  bring-up (initialize) B-57
  PC system detected
    CCB1 B-63
    CCB2 B-64
    CCB3 B-67
Ethernet
  adapter check reason codes B-52
  calculating RAM (DOS)
  calculating RAM (OS/2 EE)
  status B-56
Ethernet (DIX Version 2.0)
  frame field descriptions 1-17
  frames 1-19
event posting
  DD interface 2-14
  DLC status change 3-94
  DLR interface 2-11

F

frame
  routing information field 1-12
frame, Token-Ring network
  elements 1-10
  MAC 1-11
  non-MAC 1-11
frames
  Ethernet (DIX Version 2.0) 1-19
  IEEE 802.3 1-17
  PC network 1-16
FUNCTIONAL_ADDRESS E-5

G

GDTSIZE E-10
GROUP_ADDRESS E-5

I

I frame 6-17
I-frame 6-18
IBM PC Network
  adapter check reason codes B-52
  frames 1-16
IBM PC Network *(continued)*
  status B-56
  trace table formats 3-81
IEEE 802.2 6-2
IEEE 802.3 network
  frame field descriptions 1-17
  frames 1-17
individual bridge portion 1-15
interface
  IEEE 802.2 1-7
  NETBIOS 1-7
    APPC/PC 1-9
    direct 1-9
    DLC 1-9
    NETBIOS 1-10
    overview 1-7
interface registration CCB D-4
interrupt arbitrator
  description D-1
  interface registration CCB D-4
  language request CCB D-3
  NCB handler registration (using IEEE 802.2) D-6
  registration process D-2

L

LAN Adapter and Protocol Support
  See OS/2 LAN Adapter and Protocol Support
LAN networks, introduction to 1-3
language request CCB D-3
LANTRAN.LOG 6-9
LAPs
  See OS/2 LAN Adapter and Protocol Support
largest frame bits 1-14
length bits 1-13
link station states 2-33
LLC 6-6
LLC Type 1 protocol 2-30
LLC Type 2 protocol 2-30
local area network
  Ethernet 1-4
  IBM PC Network (Baseband) 1-4
  IBM PC Network (Broadband) 1-4
  IBM Token-Ring 1-3
  NDIS 1.01 1-4
  sample program listing diskette C-1
Local Area Network Support Program
  3270 Workstation Program support D-8
  appendages 2-4
  CCB1 calling conventions 2-3
  CCB1 command completion 2-3
  CCB1 control blocks 2-6
  interrupt arbitrator D-1
  monitored control blocks D-8
  NETBIOS operational parameters
    new 4-13
    old 4-8

Local Area Network Support Program *(continued)*
  NETBIOS operational parameters *(continued)*
    OS/2 4-13
log format
  adapter 3-36
  direct interface 3-36

M

major code 6-10, 6-20
MAX_USERS E-9
memory 6-3, 6-5, 6-8
messages 6-9
minor code
  definitions for IEEE 802.2 6-13
  definitions for NETBIOS 6-17
  for IEEE 802.2 trace 6-10, 6-16
  for NETBIOS trace 6-10, 6-20
  for Virtual DOS LAN Support 6-21
multiple group addresses
  DIR.SET.MULT.GROUP.ADDRESS 3-41

N

NCB 6-18—6-21
NCB_RESERVE 6-5, 6-7
NCB.RECEIVE 6-6, 6-8
NCB.RECEIVE.ANY 6-6, 6-8
NCB.STATUS 6-5, 6-7, 6-8
  return of adapter type by 4-38
NDIS F-1
NDIS and non-NDIS adapters 1-2
NDIS MAC drivers 1-2
NETBIND F-1
NETBIOS 6-2
  3.0 (OS/2 EE) 4-13
  calling conventions
    using DD interface 4-14
    using DLR interface 4-14
    using LAN Support Program 4-14
  commands A-3
    See also NETBIOS commands
  completion
    with no-wait type command 4-15
    with wait type command 4-15
  DLC timers 4-14
  frame formats 5-11—5-39
  frame functions
    data transfer 5-9
    miscellaneous 5-9
    name management 5-9
    session establishment and termination 5-9
  frame header
    DLC I frame 5-6
    DLC UI frame 5-6
    fields 5-6
    format 5-5

NETBIOS *(continued)*
  frame summary 5-7
  frames by transmission type 5-10
  network control block (NCB)
    description 4-3
    field explanations 4-3
  operational parameters
    new LAN Support Program 4-13
    old LAN Support Program 4-8
  OS/2 EE trace facility E-18
  overview 4-2
  protocol examples
    See NETBIOS protocol examples with RND
  Remote Name Directory (RND) 5-4
NETBIOS 4.0 4-13
NETBIOS API 6-4
NETBIOS commands
  NCB.ADD.GROUP.NAME 4-15
  NCB.ADD.NAME 4-16
  NCB.CALL 4-16
  NCB.CANCEL 4-17
  NCB.CHAIN.SEND 4-18
  NCB.CHAIN.SEND.NO.ACK 4-19
  NCB.DELETE.NAME 4-20
  NCB.FIND.NAME 4-21
  NCB.HANG.UP 4-22
  NCB.LAN.STATUS.ALERT 4-23
  NCB.LISTEN 4-24
  NCB.RECEIVE 4-25
  NCB.RECEIVE.ANY 4-26
  NCB.RECEIVE.BROADCAST.DATAGRAM 4-27
  NCB.RECEIVE.DATAGRAM 4-27
  NCB.RESET 4-28
  NCB.SEND 4-32
  NCB.SEND.BROADCAST.DATAGRAM 4-33
  NCB.SEND.DATAGRAM 4-34
  NCB.SEND.NO.ACK 4-34
  NCB.SESSION.STATUS 4-35
  NCB.STATUS 4-37
  NCB.TRACE 4-41
  NCB.UNLINK 4-45
NETBIOS enhancements
  returning additional adapter types 4-38
NETBIOS protocol examples
  SEND.NO.ACK
    data transfer 5-61
    session establishment 5-60
  with RND
    remote adapter status 5-54
    session establishment 5-57
  without RND
    name management 5-40
    remote adapter status 5-43
    session data transfer 5-49
    session establishment 5-46

network
  adapter parameter values, selecting 2-36
  IBM Token-Ring
    See Token-Ring Network
  status
    CCB1 B-52
    CCB2 B-52
    CCB3 B-54
    Token-Ring B-55
network adapter 6-2
network adapter driver 6-2, F-1
Network Status
  0x46 EXCEPTION EVENT 6-14
  minor code for NETBIOS Trace 6-18
  OS2TRACEMASK PARAMETER 6-19
  TRACE Parameter 6-16
NETWORK_FLAGS E-10
networks, introduction to 1-3
NODE_ADDRESS E-5
notification flags 2-10
NUMBER_RCV_BUFFERS E-5

O

O/S 2 Communications Manager
  device driver (DD) interface 1-5
  dynamic link routine (DLR) interface 1-5
  interface comparison 1-6
Open Adapter 6-5, 6-7
open error, adapter
  PC networks and Ethernet B-63
  Token-ring network B-58
OPEN_OPTIONS E-3
OS/2 EE
  CONFIG.SYS commands
    adapter parameters E-3
    DLC parameters E-7
  NETBIOS parameters E-11
  NETBIOS trace facility E-18
  trace code format
    X'00' – '07' E-14
    X'08' E-15
    X'09' E-15
    X'0A' E-16
    X'0B' – '0C' E-17
    X'0D' E-18
  trace entry definition E-12
  trace facility E-11
OS/2 EE programming conventions
  DD interface
    calling conventions 2-13
    CCB2 control blocks 2-18
    command completion 2-14
    user's data segment guidelines 2-13
    user's data segment restrictions 2-12
  DLR interface
    calling conventions 2-8
OS/2 EE programming conventions *(continued)*
  DLR interface *(continued)*
    CCB2 control blocks 2-12
    command completion 2-9
    execute multiple commands 2-7
    user's data segment guidelines 2-8
    user's data segment restrictions 2-7
OS/2 LAN Adapter and Protocol Support 6-1
OS2TRACEMASK 6-10, 6-19, 6-20

P

parallel bridges 1-16
parameters
  TRACENAMES for LANDD.OS2 and NETBEUI.OS2 6-12
  TRACEOFF for LANDD.OS2 and NETBEUI.OS2 6-11
PC Network, IBM
  adapter check reason codes B-52
  frames 1-16
  status B-56
  trace table formats 3-81
PDT.TRACE.OFF 3-76
PDT.TRACE.ON 3-77
piggyback acknowledgment 6-6, 6-8
posting events
  DD interface 2-14
  DLC status change 3-94
  DLR interface 2-11
PRODUCT_ID E-6
programming conventions
  DOS 2-3
  OS/2 EE 2-7
protocol 6-2
protocol driver 6-2, F-1
protocol drivers 1-2
protocol examples, NETBIOS
  SEND.NO.ACK
    data transfer 5-61
    session establishment 5-60
  with RND
    remote adapter status 5-54
    session establishment 5-57
  without RND
    name management 5-40
    remote adapter status 5-43
    session data transfer 5-49
    session establishment 5-46
protocol manager F-1
PROTOCOL.INI
  configuration and binding information 6-2
  keyword separators 6-9
  OS2TRACEMASK parameter 6-10, 6-19
  relationship to protocol manager and NETBIND F-1
  TRACE parameter 6-10, 6-15

publications, related xviii
PURGE.RESOURCES 3-85

Q

QSIZE E-10

R

RCV_BUFFER_LENGTH E-5
READ 3-86
READ.CANCEL 3-94
reason codes, adapter check
  IBM Token-Ring network B-51
  PC network and Ethernet B-52
RECEIVE 3-95
RECEIVE.CANCEL 3-102
RECEIVE.MODIFY 3-103
related publications xviii
Remote Name Directory (RND) 5-4
return codes
  DLC and direct interface B-2
  DLC and direct interface by command
    CCB1 B-5
    CCB2 B-8
    CCB3 B-11
  DLC and direct interface cause and action B-14
  NCB (NETBIOS) B-32
  NETBIOS by command B-34
  NETBIOS cause and action B-35
ring number portion 1-15
Ring Status 6-5, 6-7
route designator field 1-15
  individual bridge portion 1-15
  ring number portion 1-15
routing control field 1-12
  broadcast indicators 1-12
  direction bit 1-13
  largest frame bits 1-14
  length bits 1-13
routing field for bridges 1-12
routing information field 1-12
  route designator field 1-15
  routing control field 1-12

S

sample LAN program listing diskette C-1
SAP assignments
  global 2-29
  null 2-29
software, IBM support
  Local Area Network Support Program 2-3
  O/S 2 Communications Manager 1-5, 2-7
  overview 1-4
source routing
  direction bit 1-13
  length bits 1-13
  route designator fields 1-15
  routing control field 1-12
SRAM_ADDRESS E-7
station ID 2-27
status, network
  CCB1 B-52
  CCB2 B-52
  CCB3 B-54
  PC and Ethernet B-56
  Token-Ring network B-55
system action exceptions (OS/2 EE)
  CCB2 B-69
  CCB3 B-72
system trace 6-10
SYSTEM_KEY E-9

T

timers, DLC 2-35, 4-14
Token-Ring Network Adapter
Token-Ring Network, IBM
  adapter check reason codes B-51
  adapter communication
  data transmission
  frame 1-10
  routing field 1-12
  status codes B-55
  trace table formats 3-79
TRACE 6-10, 6-15, 6-16
trace facility (OS/2 EE)
  codes
    X'00' – '07' E-14
    X'08' E-15
    X'09' E-15
    X'0A' E-16
    X'0B' – '0C' E-17
    X'0D' E-18
  entry format E-12
  NETBIOS E-18
  overview E-11
trace table formats
  PC network 3-81
  Token-Ring network 3-79
TRACENAMES parameter 6-12
TRACEOFF parameter 6-11
transmit buffers 6-6
TRANSMIT.DIR.FRAME 3-107
TRANSMIT.I.FRAME 3-108
TRANSMIT.TEST.CMD 3-109
TRANSMIT.UI.FRAME 3-109
TRANSMIT.XID.CMD 3-109
TRANSMIT.XID.RESP.FINAL 3-110

TRANSMIT.XID.RESP.NOT.FINAL 3-110

U

UI-frame 6-17, 6-18
user notification flags 2-10

V

Virtual LAN Support 6-2

Printed in U.S.A.

SC30-3587-00