IBM

Series/1

SL23-0104-2 + 2.1-P02

EDX Communications Facility Design and Installation Guide

Version 2

Introduction GL23-0103 2	Design and Installation Guide SL23-0104 2	Operator's Guide SL23-0105 2
Master Index and Glossary SL23-0121 2	Programmer's Guide SL23-0106 2	Messages and Codes SL23-0120 3
Debugging Guide LL23-0109 2	Programmer's Reference Summary SX23-0145 1	Operator's Reference Summary SX23-0108 1

IBM

Series/1

SL23-0104-2

EDX Communications Facility Design and Installation Guide

Version 2

Introduction GL23-0103	Design and Installation Guide SL23-0104	Operator's Guide SL23-0105
Master Index and Glossary SL23-0121	Programmer's Guide SL23-0106	Messages and Codes SL23-0120
Debugging Guide LL23-0109	Programmer's Reference Summary SX23-0145	Operator's Reference Summary SX23-0108

Third Edition (September 1986)

This edition applies to Version 2.1 PTF 1 of the Licensed Program IBM Series/1 Event Driven Executive Communications Facility, Program Number 5719-CF2, and to all subsequent versions and modifications of this program unless otherwise indicated in new editions or technical newsletters.

Information in this publication is subject to significant change. Any such changes will be reported in new editions or technical newsletters. Use this publication only for the purposes mentioned under "About This Book" on page v.

It is possible that this material may contain references 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.

Direct all requests for copies of IBM publications to your IBM representative or to the IBM branch office that serves your locality.

Any names of individuals, companies, brands, and products in the examples in this book are fictitious; any similarity to the name of an actual business enterprise is entirely coincidental.

This publication may contain technical or typographical errors. Forms for readers' comments are provided at the back of the book. If the forms have been removed, address your comments to IBM Corporation, System Products Division, 4J1/037, PAS5 MS33, PO Box 10500, Palo Alto, CA 94303. IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

About This Book

This book is intended for people who are planning to install the IBM Event Driven Executive Communications Facility for the Series/1. It assumes that you already understand the Communications Facility's functions and know how your installation is going to use the Communications Facility. If you need introductory information about the Communications Facility, refer to *Series/1 EDX Communications Facility Introduction*, GL23-0103. Some sections of the book assume you are familiar with systems network architecture (SNA); the International Telegraph and Telephone Consultative Committee (CCITT) Recommendations (X.25 in particular) for data communication networks; and the Series/1-PC Connect for EDX (5719-CN1). See the publications list for book titles on these subjects.

In this book:

Asynchronous adapter is used to mean feature programmable communications adapter, multifunction attachment, or 4950/5170-495 terminal/host adapter.

Series/1-PC Connect Attachment is used to mean a Series/1 attachment card (feature number 4000) and a PC extender card shipped with this feature.

3101 is used to mean 3101 terminals (models 12, 13, 22, and 23), 3161, 3163, or 3164 terminals in 3101 emulation mode, and IBM Personal Computers running in 3101 emulation mode as if they were 3277 model 2 terminals.

EDX Secondary SNA1 refers to the licensed program 5719-SX1.

EDX Secondary SNA2 refers to the licensed program 5719-XX9.

EDX-SNA refers to both the EDX Secondary SNA1 and SNA2. It is used to indicate functions supported by both versions of SNA.

This book describes the functions and components of the Communications Facility and gives you the information you need to define, install, and start using your communications configuration. To that end, it has these chapters:

- "Communications Facility Components" on page 1 introduces the programs and data sets that make up the Communications Facility. It explains Communications Facility concepts and facilities that you'll need to understand before you begin configuring and installing your system.

- "Defining Stations" on page 45 describes the various types of stations you may need to define and explains how to define each one.

- "Using the Supplied Input/Output Control Programs" on page 71 explains the requirements and restrictions you need to consider when using the input/output control programs supplied as part of the Communications Facility.

- "Communications Facility Data Sets" on page 179 describes the data sets that are used to hold Communications Facility information, and explains how to change their contents.

- "Creating $.SYSPD" on page 187 explains the structure of the program dispatcher data set, and gives the syntax of the statements that create entries in the data set.

- "Creating Disk-Queue Data Sets" on page 207 explains how to create the data sets that hold messages that are queued on disk rather than in processor storage.

- "Planning Storage Requirements" on page 213 shows how to determine how much processor storage the Communications Facility will require. It includes tables that list the sizes of the Communications Facility modules as distributed from ISD.

- "Coding Supervisor Definition Instructions" on page 235 shows how to code instructions to define elements of your Communications Facility configuration to the EDX supervisor.

- "Installing the Communications Facility" on page 251 explains how to install the Communications Facility.

- "Sample Configurations" on page 269 presents examples of Communications Facility configurations. Each example includes a diagram of the configuration; an explanation of the message flow between stations; station definitions; EDX definitions; and the EDX link control data set.

- "Maintaining the Communications Facility" on page 445 explains how to assemble and link-edit Communications Facility programs.

- The "Glossary" presents the definitions of technical Communications Facility terms and acronyms. For EDX definitions, see the appropriate book.

- The "Index" is a conventional index to the publication.

To use this book effectively, you will need a working knowledge of the Event Driven Executive (EDX) operating system. Pertinent publications include:

- *Series/1 Event Driven Executive Installation and System Generation Guide,* SC34-0646

- *Series/1 Event Driven Executive Operation Guide,* SC34-0642

- *Series/1 Event Driven Executive Language Programming Guide,* SC34-0637

- *Series/1 Event Driven Executive Communications Guide,* SC34-0638

- *Series/1 Event Driven Executive Language Reference,* SC34-0643

- *Series/1 Event Driven Executive Operator Commands and Utilities Reference,* SC34-0644
- *Series/1 Event Driven Executive Messages and Codes,* SC34-0636.

Other Communications Facility publications you might refer to are:

- *Series/1 EDX Communications Facility Operator's Guide,* SL23-0105 (referred to in this book as the *Operator's Guide*), which explains how to operate the Communications Facility using operator commands, IBM-supplied transactions, and utility programs.
- *Series/1 EDX Communications Facility Messages and Codes,* SL23-0120, which lists and explains the Communications Facility error and informational messages and the return codes from Communications Facility programming instructions.
- *Series/1 EDX Communications Facility Programmer's Guide,* SL23-0106 (referred to in this book as the *Programmer's Guide*), which explains how to code application programs, device-support programs, transaction-processing programs, and command-processing programs to run under the Communications Facility.
- *Series/1 EDX Communications Facility Debugging Guide,* LL23-0109 (referred to in this book as the *Debugging Guide*), which describes the Communications Facility modules, suggests steps to follow in debugging, and explains how to submit an authorized program analysis report (APAR) to IBM.

For information about the Series/1-PC Connect, see the *Series/1-PC Connect for the Event Driven Executive Communications Facility,* SC34-0698.

For information about the Communications Facility/Host licensed program, see *Series/1 Communications Facility/Host General Information,* GH20-2485.

For information about the EDX X.25/HDLC Communications Support licensed program, see *Series/1 Event Driven Executive X.25/HDLC Communications Support Programming and Operating Reference Manual,* SC09-1030, referred to in this book as the XHCS *Programming and Operating Reference.*

If you need information about 3101 devices, see the *3101 Display Terminal Description,* GA18-2033.

If you need information about 3161 or 3163 display stations, see the *3161/3163 Display Station Operator Reference and Problem Solving Guide,* GA18-2311.

If you need information about 7485 devices, see the *7485 Display Terminal Description,* GA18-2075.

If you need information about the 4978 or 4980 display terminal, see the:

- *Series/1 4980 Display Station Description and Reference,* GA21-9296
- *Series/1 4978-1 Display Station and Attachment General Information,* GA34-1550.

If you need information about 3270 devices, see the:

- *3270 Information Display System: 3271 Control Unit, 3272 Control Unit, 3275 Display Station Description and Programmer's Guide,* GA23-0060 (referred to in this book as the *3271 Guide*)
- *3270 Information Display System: 3274 Control Unit Description and Programmer's Guide,* GA23-0061 (referred to in this book as the *3274 Guide*)
- *3287 Printer Component Description,* GA27-3153.

If you're using EDX Secondary SNA1, see the *Event Driven Executive: Systems Network Architecture and Remote Job Entry Guide,* SC34-0402 (referred to in this book as the *EDX Secondary SNA1 Guide*).

If you're using EDX Secondary SNA2, see the *Event Driven Executive: Systems Network Architecture and Remote Job Entry Guide,* SC34-0773 (referred to in this book as the *EDX Secondary SNA2 Guide*).

If you need more information about binary synchronous communication (BSC) protocol, see *General Information—Binary Synchronous Communications,* GA27-3004, and *Series/1 Binary Synchronous Features Description,* GA34-0244.

If you're using the multifunction attachment feature, see the *Multi-function Attachment Feature Diskette Initialization User's Guide,* GA34-0147, and the *Multi-function Attachment Feature and 4975 Printer Description,* GA34-0144.

For information about the channel attachment feature, refer to the *Series/1 System/370 Channel Attachment Feature and 4993 Model 1—Series/1 System/370 Termination Enclosure Description,* GA34-0057.

If you're using the Local Communications Controller, see the *Series/1 Local Communications Controller Feature Description,* GA34-0142.

If you're using the Series/1-PC Connect Attachment, see the *Series/1-Personal Computer Channel Attachment Description,* GA34-0287.

If your system is going to communicate with the Realtime Programming System's Communications Manager, you may want to refer to the Series/1 Realtime Programming System *Communications Manager Assembler Language Programmer's Guide,* SL23-0123, which explains Communications Manager message formats; and the *Communications Manager Operator's Guide,* SL23-0128, which explains Communications Manager commands.

Summary of Amendments

Only changes affecting the content of this book are listed in the Summary of Amendments.

Version 2.1 PTF 2

The following changes were made between the Communications Facility Version 2.1 PTF 1 and Version 2.1 PTF 2:

EDX SYSTEM Statement Support: Communications Facility now supports the COMBASE parameter of the EDX SYSTEM statement. This support allows you to use a partition, other than partition 1, as the Communications Facility common area. The following changes were made:

- A description of the new storage pool, S$CFDCBS, was added to "Communications Facility Components" on page 1.
- The descriptions of DEFINE BRB and DEFINE BUFFERPOOL in "Coding Supervisor Definition Instructions" on page 235 were changed.
- The storage requirements for the supervisor partition, $.IO06F0, and $.IO0680 in "Planning Storage Requirements" on page 213 were changed.
- Installation requirements were added to "Installing the Communications Facility" on page 251.

Remote TCB Support for $.IO0670: The description of "4978/4980 Data Stream IOCP, $.IO0670" on page 137 was changed to reflect the new support. $.IO0670 now places its device TCBs in the dynamic storage area instead of the station control block in S$POOL. "Planning Storage Requirements" on page 213 was updated to reflect this change.

Retry Support: The description of "3270 Control IOCP, $.IO0AC0" on page 97 was changed to add the new support. $.IO0AC0 now inactivates a control unit station if the maximum number of time-outs allowed are exceeded. The device station is not polled until the control unit station is started again by the operator.

Other Changes

- The dynamic storage requirements for $.IO0674 have been changed in "Planning Storage Requirements" on page 213.

- Appendix A, "Configuring and Connecting 3101 Display Terminals" on page 452.1 has been added to explain the hardware configurations required to connect a 3101 device to a Series/1 using the 3101 IOCP, $.IO06F0.

Version 2.1 PTF 1

The following changes were made between the Communications Facility Version 2.1 and Version 2.1 PTF 1:

EDX Secondary SNA2 Support: Communications Facility now supports EDX Secondary SNA2, 5719-XX9. This support allows you to run Communications Facility with EDX Secondary SNA1 or EDX Secondary SNA2. The following changes were made:

- $.IO14E8, the SNA IOCP, was modified to include the new SNA2 support.

- Installation requirements were added to "Installing the Communications Facility" on page 251.

- A new sample configuration, "Example 4: EDX Secondary SNA2 Connection to a Host" on page 296, was added.

Version 2.1

The following changes were made between the Communications Facility Version 2.0 and Version 2.1:

Series/1-PC Connect Support: A new IOCP, $.IO0AA8, was added to provide support for communication between a Series/1 and Personal Computers in a local area network (LAN). The following changes were made:

- A description of the new IOCP was added to "Using the Supplied Input/Output Control Programs" on page 71.

- The IOCP's storage requirements are described in "Planning Storage Requirements" on page 213.

- A new instruction, PCC, was added to "Coding Supervisor Definition Instructions" on page 235.

- Installation requirements were added to "Installing the Communications Facility" on page 251.

- Two new sample configurations were added: "Example 14: X.25 Connection between a Series/1 and DTEs" on page 406, and "Example 15: LAN with

Series/1-PC Connect Attachment and EDX Secondary SNA1 Connection" on page 418.

- "Defining Node Stations" on page 61 was modified to include the PC node.

Disk-server Support: A new IOCP, $.IO1CC8, was added to provide support for emulated PC disks. A description of the IOCP was added to "Using the Supplied Input/Output Control Programs" on page 71. The IOCP's storage requirements are described in "Planning Storage Requirements" on page 213.

$.DSFORM: A new utility, $.DSFORM, which manages emulated PC disks, is described in "$.DSFORM, Emulated PC Disk Management Utility" on page 41. A description of the new control data set, $.DSPROF, has been added to "Communications Facility Data Sets" on page 179.

Volume Station: A description of the volume station was added to "Defining Stations" on page 45. This station represents an EDX volume that contains data sets representing emulated PC disks.

CFBUF Usage: The Communications Facility now monitors the use of space in the message buffer pool (CFBUF). This new support is described in "Message Buffer Pool (CFBUF) Usage" on page 17.

3101 Keyboard Change: Figure 25 on page 128 has been changed to show the new 3101 keyboard.

Other Changes:

- The non-RPQ 4978 IOCP, $.IO0670, is no longer supported. Its description was removed from "Using the Supplied Input/Output Control Programs."

- 4978/4980 data stream IOCP is now called $.IO0670 and is described in "4978/4980 Data Stream IOCP, $.IO0670" on page 137.

- The teletypewriter connected 3101 IOCP, $.IO0630, is no longer supported. Its description was removed from "Using the Supplied Input/Output Control Programs."

Version 2.0

The following changes were made between the Communications Facility Version 1.2 and Version 2.0:

X.25 Support: A new IOCP, $.IO0AB8, was added to provide support for communication using Recommendation X.25 protocol. The following changes were made:

- A description of the new IOCP was added to "Using the Supplied Input/Output Control Programs" on page 71.

- The new data set, $.SYSX25, is described in "X.25 Call ID Data Set ($.SYSX25)" on page 184.

- Two new modes, STOP and RETRY, are described in the section "STOP and RETRY Modes" on page 74.

- The IOCP's storage requirements are described in "$.IO0AB8, X.25" on page 225.

- A new sample configuration worksheet was added—Figure 18 on page 55; the line/terminal/ node worksheet—Figure 16 on page 53—was changed to include DTE and DCE lines.

- Installation requirements were added to "Installing the Communications Facility" on page 251.

- Three new sample configurations were added: "Example 11: Multiple Local Communications Controller and BSC Connections" on page 374, "Example 12: X.25 Connection between Series/1s" on page 382, and "Example 13: X.25 Connection Between Series/1s through an X.25 Network" on page 392.

4980 Terminal Support: The description of "4978/4980 Data Stream IOCP, $.IO0670" on page 137 was changed to reflect the addition of the 4980 terminal. The keyboard is shown in Figure 29 on page 144. The Communications Facility data sets required for 4980 operation are described in "4980 Data Stream Data Sets ($4980CSA, $4980ISA, and $4980R0A)" on page 182. "Example 1: BSC Connection to a Host" on page 272 shows a 4980 as part of a Communications Facility configuration.

LU-1 Support: The Communications Facility now supports the Series/1 printers as LU-1s. A description of the new support is in "Printer IOCP, $.IO0674" on page 149. Two new modes, SCS and 3270 are explained in "3270 and SCS Modes" on page 75. Installation procedures are described in "Installing the Communications Facility" on page 251. "Example 3: EDX Secondary SNA1 Connection to a Host" on page 284, was changed to show LU-1s as part of a Communications Facility configuration.

Model 30D and 60D Integrated Disk Storage Support: "Remote IPL" on page 37 was changed to include the 30 megabyte and 60 megabyte integrated disks as remote IPL devices. (Communications Facility books call these disks DDSK-30 and DDSK-60 for consistency with the Event Driven Executive books.)

$.PNLUT1: A new utility $.PNLUT1, which prints panels defined by $.PANEL, is described in "$.PNLUT1, Panel Utility" on page 41.

Module Names and Sizes: Figure 39 on page 216 and "Message Flow within a Node" on page 6 were updated to reflect all new and changed modules.

3101F Device Station Use of Unmapped Storage: "3101 IOCP, $.IO06F0" on page 121 and "7485 IOCP, $.IO0680" on page 155 include changes to $.IO06F0 and $.IO0680 related to use of unmapped storage. "Planning Storage Requirements" on page 213 was updated to reflect these changes. A step for converting your existing 3101F station definitions to new ones that use unmapped storage was added to "Installing the Communications Facility" on page 251.

Publications Changes: Step 6 of the installation procedures, "Edit the Supervisor Link Control Data Set," was expanded. Because more information was added to this section, the example link control data sets were deleted from the sample configurations.

Contents

Figures

Communications Facility Components

The Communications Facility manages communication among programs and various input/output devices, such as terminals, printers, and communication lines. It supports communication within a Series/1, between Series/1s, and between a Series/1 and other computers. It consists of programs that run under control of the Event Driven Executive (EDX) operating system.

This chapter introduces the programs and data sets that make up the Communications Facility. It explains the Communications Facility concepts and facilities that you'll need to understand before you can begin the tasks of configuring and installing your Communications Facility system.

Figure 1 on page 2 shows the components of the Communications Facility—programs, data sets, and pools of processor storage. The portions included in the EDX supervisor and the Communications Facility control program are required in every Communications Facility system. Each of the other programs is loaded into processor storage only when its functions are needed. The figure shows Communications Facility data sets on the left and right, with lines connecting each data set to the primary programs that use it.

A Communications Facility system can also include user-written programs that process message data or perform functions unique to your installation.

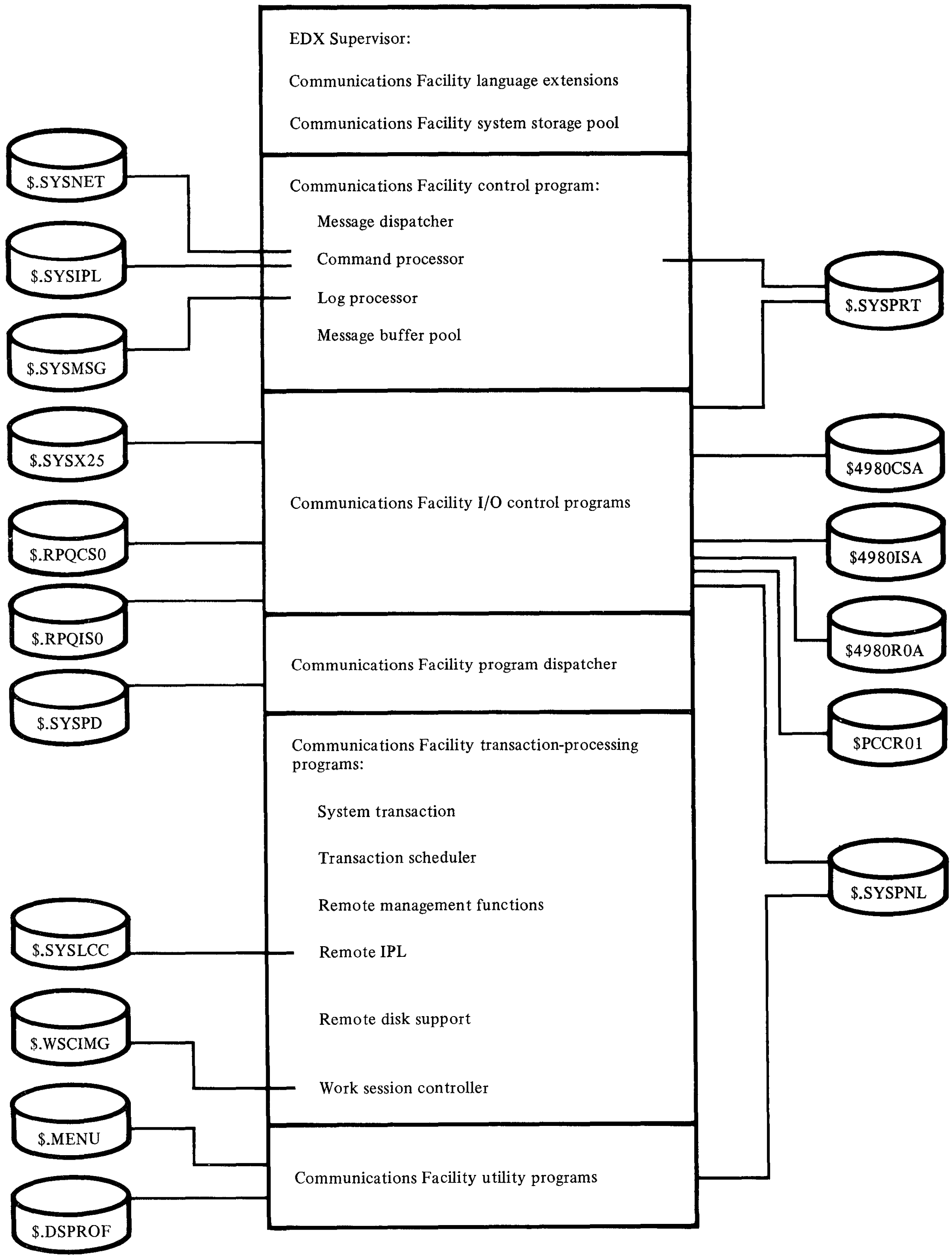


Figure 1. Summary of Communications Facility Components

Data Sets

The Communications Facility includes the data sets summarized in Figure 2.

Name	Contents
$4980CSA	4980 control store
$4980ISA	4980 image store
$4980R0A	4980 microcode patch data set
$PCCR0*n*	PC Connect microcode patch data set. *n* is the EC level.
$.DSPROF	Control information for data sets used as emulated PC disks
$.MENU	Work session controller sample program menu
$.RPQCS0	4978 data stream (RPQ D02428) control store
$.RPQIS0	4978 data stream (RPQ D02428) image store
$.SYSIPL	Communications Facility startup commands
$.SYSLCC	Local Communications Controller remote IPL transactions
$.SYSMSG	Text of error and informational messages
$.SYSNET	Station definitions
$.SYSPD	Program dispatcher startup commands, path definitions, transaction definitions, and transactions
$.SYSPNL	System panels (3270 screen images)
$.SYSPRT	4978 control and image stores, buffered printer data
$.SYSX25	X.25 network addresses

Figure 2. Summary of Communications Facility Data Sets

The chapter "Communications Facility Data Sets" on page 179 describes each of the data sets in detail and explains how you can modify them to suit your requirements.

The data set of particular importance to this discussion of system components is $.SYSNET, the network configuration data set. $.SYSNET contains the definitions of *stations.* A station is a named unit of hardware or software managed by the Communications Facility. It represents a program or an I/O device, such as a terminal or a communication line, that sends and receives messages. There is one $.SYSNET data set for each *node*—each Series/1 in a Communications Facility configuration. The $.SYSNET data set at a node contains the definitions of local stations (stations in that node) and of remote stations (stations in other nodes) to which local stations will send messages. The chapter "Defining Stations" on page 45 describes the various types of stations and explains how to define them.

System Storage Pool (S$POOL)

The *system storage pool (S$POOL)* is an area of processor storage that contains information about stations that have been started. When a station is started, a control block, called a *station block*, is created in S$POOL, and remains there until the station is halted. The station block contains all the information the Communications Facility needs to manage the station. The information in the station block comes from the station's definition in $.SYSNET.

S$POOL may also contain other information: a program loader, which the program dispatcher uses, and work areas for some of the I/O control programs (the programs that manage devices).

S$POOL is part of the EDX supervisor, and is mapped into all partitions in which Communications Facility programs run. The chapter "Installing the Communications Facility" on page 251 explains how to define S$POOL and include it in the supervisor. The chapter "Planning Storage Requirements" on page 213 explains how to calculate S$POOL's size for your installation.

DCB Pool (S$CFDCBS)

The *device control block (DCB)* pool is an area of processor storage that contains DCBs and interrupt handler routine calls for certain types of started device stations. Any hardware device managed by the Communications Facility without using EDX-provided I/O facilities requires a DCB in the DCB pool.

The DCB pool is part of the EDX supervisor, and must be included in partition 1. The chapter "Installing the Communications Facility" on page 251 explains how to define S$CFDCBS and include it in the supervisor. The chapter "Planning Storage Requirements" on page 213 explains how to calculate the size of S$CFDCBS for your installation.

Message Buffer Pool (CFBUF)

The *message buffer pool (CFBUF)* is an area of processor storage that contains messages that have been sent but not yet received. CFBUF contains queues of messages—one queue for each destination station. Each station block in S$POOL contains a pointer to the station's message queue in CFBUF.

At your option, you may queue low-priority messages on disk. If you take this option, there is another storage pool, used for *file control blocks*. A file control block contains information about a station that has a message queue on disk.

Both of these pools are in the dynamic storage of the Communications Facility control program. The chapter "Planning Storage Requirements" on page 213 explains how to estimate how big the dynamic storage should be for your installation and how to alter the distributed size.

The Communications Facility monitors the use of space in the message buffer pool. The section "Message Buffer Pool (CFBUF) Usage" on page 17 explains how you can display information on the usage of the message buffer pool and how you can regulate the flow of messages depending on the usage levels of CFBUF.

Language Extensions

The Communications Facility includes a set of instructions that extend the Event Driven Executive Language (EDL). These instructions are called *language extensions*. When you install the Communications Facility, you include the support for the language extensions in the EDX supervisor, as explained in the chapter "Installing the Communications Facility" on page 251. The supplied Communications Facility programs use the language extension instructions, and you can use them in your application programs.

The instructions fall into these categories:

Message management instructions, used to send and receive messages.

Storage management instructions, used to get and free blocks of processor storage.

Station management instructions, used to create and delete station blocks and to control stations.

3270 data stream instructions, used to create 3270 data streams and to extract data from them.

A *data move* instruction, used to perform indexed and indirect moves.

A *system facilities* instruction, used to gain access to various Communications Facility control blocks.

A *task control* instruction, used by I/O control programs (IOCPs) to activate and deactivate tasks.

Queue management instructions, used to put elements on queues and to remove elements from queues.

Supervisor definition instructions, used to define elements of a Communications Facility configuration to the EDX supervisor. These instructions are described in this book, in the chapter "Coding Supervisor Definition Instructions" on page 235.

All the instructions are described in the *Programmer's Guide*.

Communications Facility Control Program

The control program is that part of the Communications Facility that provides basic, required services. It contains message dispatcher, command processor, and log processor functions—and the dynamic storage used for the message buffer pool and the file control block pool.

There are two versions of the control program. One, $.CFS, supports queuing of messages only in processor storage. The other, $.CFD, supports queuing of messages in processor storage and on disk. You start the version you want by issuing an EDX $L command to load it. Once either version is loaded, it sets the name in its program header to $.CF.

Message Dispatcher

When a message is sent, the message dispatcher determines its destination and places it on the destination station's message queue, either in processor storage or on disk. Message dispatching is performed partly by the SEND instruction and partly by the control program. The SEND instruction dispatches messages that are queued in processor storage. The SEND instruction and the control program together dispatch messages that are queued on disk.

A message can be as large as 32K bytes and must be contained in a single storage area. The Communications Facility does not support segmented messages.

Message Flow within a Node

Figure 3 on page 7 is an example of message flow within a node. It shows how messages flow between an application program and terminals. An IOCP manages the terminals. Station blocks TERM1, TERM2, and TERM3 represent the terminals, and station block APROG represents the application program. The numbered lines show the flow of a message from one of the terminals to the application program and the flow of the response from the application program back to the terminal.

1 The IOCP reads data from the terminal represented by station TERM1 and builds a message in its working storage area.

2 The IOCP issues a SEND instruction specifying TERM1 as the origin station and a null destination station. IOCPs know the names of stations they manage, but they are not responsible for knowing the destinations of the messages they send.

The SEND instruction builds a message header that includes the message's type, priority, origin, and destination. The SEND instruction acquires space in the message buffer pool and moves the header and message to the pool.

3 The message dispatcher examines the header. The destination is null, so the message dispatcher examines the origin station's station block to find the default destination. When you define a station, you can specify a default destination for messages it sends. This connection from one station to another is called a *direct link vector*, or just a *direct link*. As

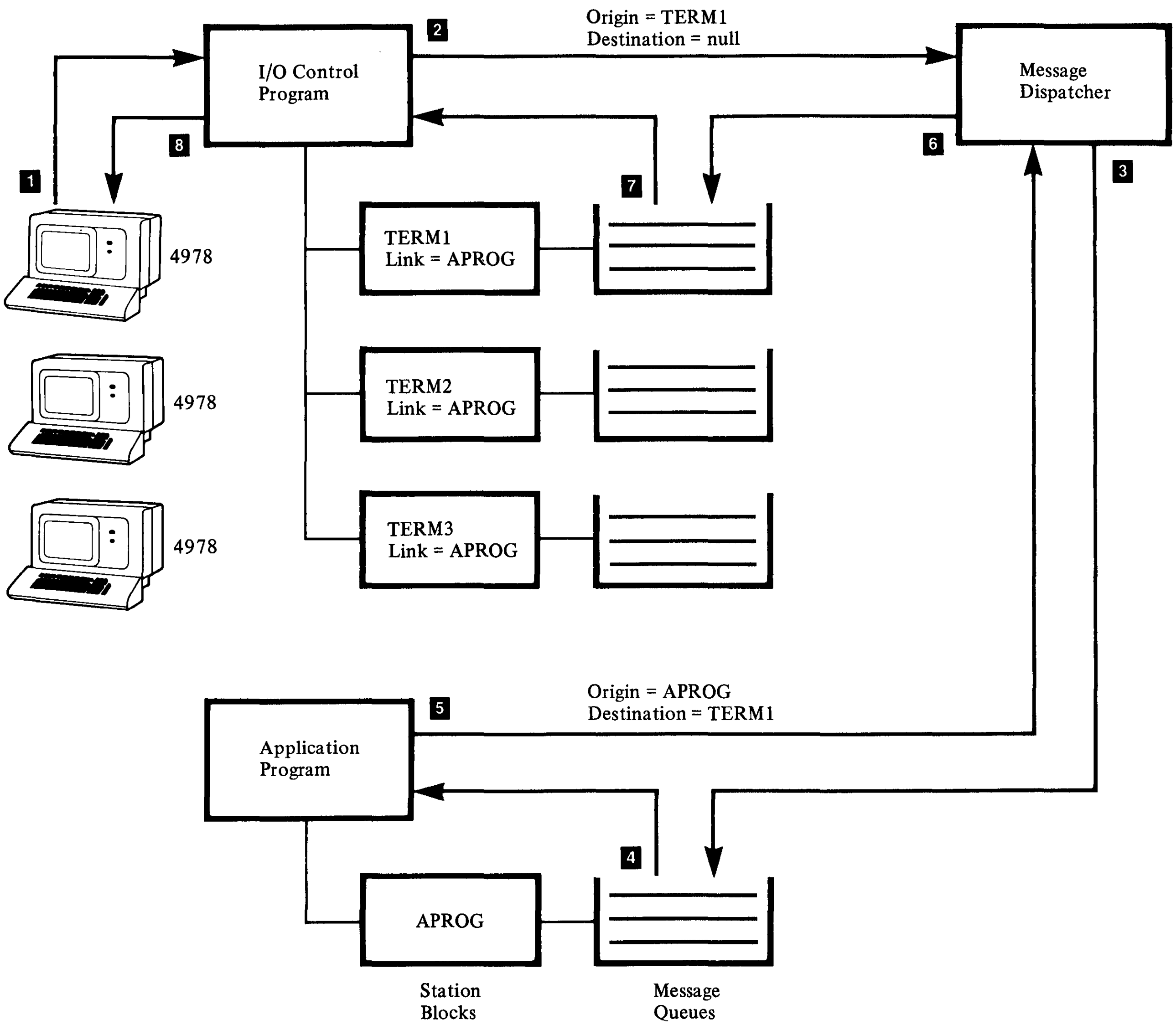


Figure 3. Message Flow within a Node

shown in Figure 3, station TERM1 is linked to APROG, so the message dispatcher places the message on APROG's message queue. If the origin station (TERM1) had no direct link, the message would be undeliverable.

4 The application program issues a RECEIVE instruction to get a message from station APROG's message queue. The RECEIVE instruction moves the message to the receiving program's working storage area, adjusts queue pointers, and frees the space in the message buffer pool. The receiving program always receives the message data. It can optionally receive either the entire message header or the name of the origin station.

Application programs that communicate with multiple users don't usually care which particular station they receive a message from. They just save the origin of each message and direct the response back to that station.

5 The application program processes the message, builds a response, and issues a SEND instruction specifying itself as the origin station and TERM1 as the destination station.

6 The message dispatcher examines the message header. The destination is specified, so it places the message on station TERM1's message queue.

7 The IOCP issues a RECEIVE instruction to get a message from station TERM1's message queue.

8 The IOCP writes the message data to the terminal represented by station TERM1.

Communications Facility programs that send messages usually don't wait for the destination station to receive a message. Instead, they wait only until the message is placed on the destination station's message queue. In the example, the IOCP resumes processing input or output for its stations at the end of step **3**. The application program receives its next message (or waits for one to be placed on its message queue) at the end of step **6**. You can write application programs that wait until a message has been received (or received and processed) at its destination, but none of the IOCPs or other Communications Facility programs work that way.

Message Flow between Nodes

Figure 4 on page 9 is an example of message flow between nodes. It shows how messages flow between an application program in one node and terminals in another node. In each node, there is an IOCP that manages the communication line between nodes; a station that represents the remote node; and stations that represent the remote stations to which local stations will send messages. Each station definition includes the identification of the node in which the hardware or program represented by the station exists. In node 1, the stations that represent the terminals (only one of which is shown in the figure) have node ID 1; stations NODE2 and APROG have node ID 2. In node 2, the station that represents the application program has node ID 2; stations NODE1, TERM1, TERM2, and TERM3 have node ID 1.

1 The IOCP that manages the terminals reads data from the terminal represented by station TERM1 and builds a message in its working storage area.

2 The IOCP issues a SEND instruction specifying TERM1 as the origin station and a null destination station.

The SEND instruction builds a message header that includes the message's type, priority, origin, and destination. The SEND instruction acquires space in the message buffer pool and moves the header and message to the pool.

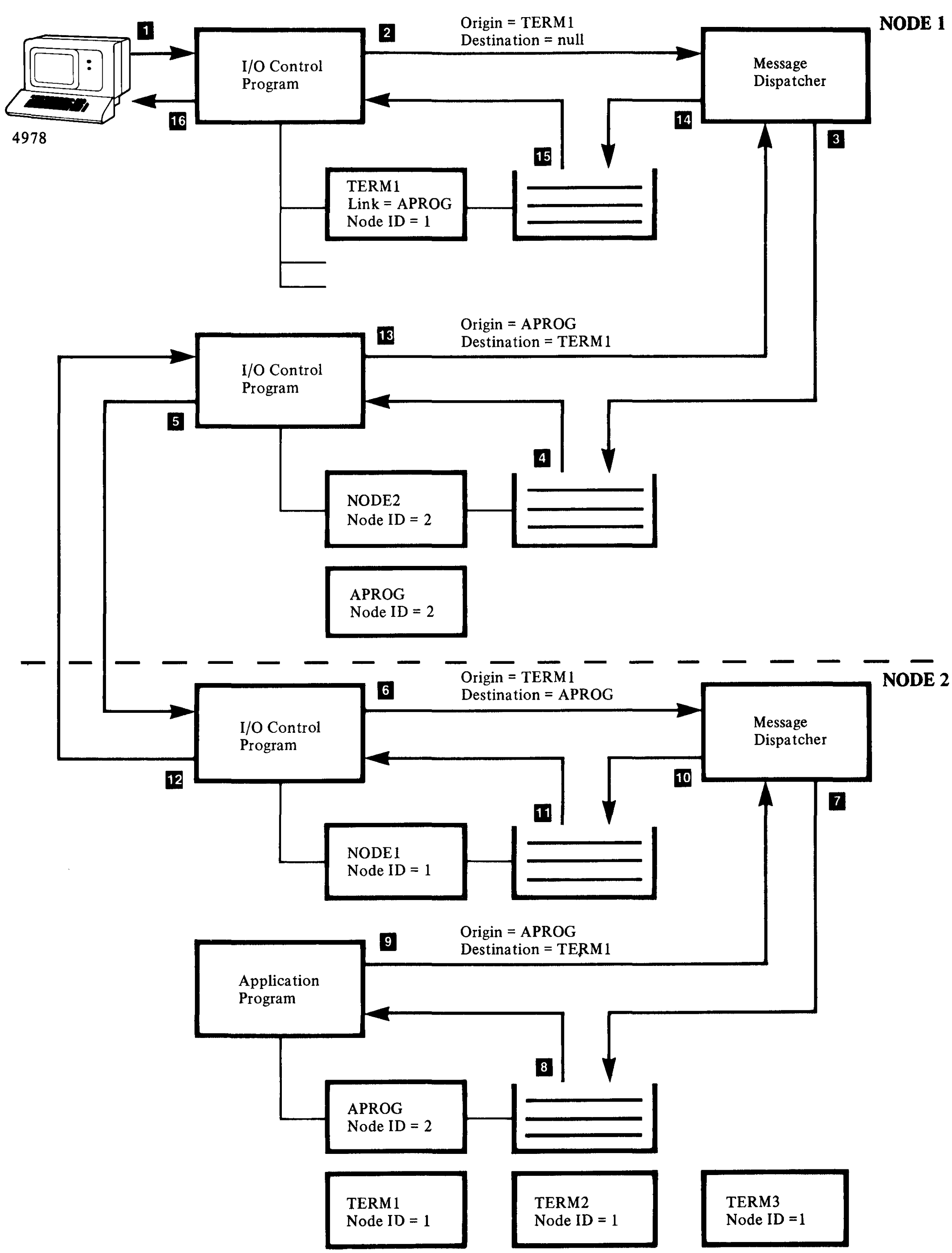


Figure 4. Message Flow between Nodes

3 The message dispatcher in node 1 examines the header. The destination is null, so the message dispatcher examines the origin station's station block to find the default destination. As shown in Figure 4 on page 9, station TERM1 is linked to APROG, so the message dispatcher places that information in the message header.

Because station APROG is in node 2, the message dispatcher places the message on the queue of the station that represents that node (NODE2).

4 The IOCP in node 1 that manages the communication line issues a RECEIVE instruction to get a message from station NODE2's queue. The RECEIVE instruction moves the message to the receiving program's working storage area, adjusts queue pointers, and frees the space in the message buffer pool.

5 The IOCP writes the message header and data to the communication line; the IOCP in node 2 reads it.

6 The IOCP in node 2 sends the message with its header, which specifies that the origin station is TERM1 and the destination station is APROG.

7 The message dispatcher in node 2 examines the header. The destination is APROG, so the message dispatcher places the message on APROG's message queue.

8 The application program issues a RECEIVE instruction to get a message from station APROG's message queue.

9 The application program processes the message, builds a response, and issues a SEND instruction specifying itself as the origin station and TERM1 as the destination station.

10 The message dispatcher in node 2 examines the message header. The destination, TERM1, is in node 1, so the message dispatcher places the message on the queue of the station that represents that node, NODE1.

11 The IOCP in node 2 issues a RECEIVE instruction to get a message from station NODE1's message queue.

12 The IOCP writes the message header and data to the communication line; the IOCP in node 1 reads it.

13 The IOCP in node 1 sends the message with its header, which specifies that the origin station is APROG and the destination station is TERM1.

14 The message dispatcher in node 1 examines the header. The destination is TERM1, so it places the message on TERM1's message queue.

15 The IOCP that manages the terminals issues a RECEIVE instruction to get a message from station TERM1's message queue.

16 The IOCP writes the message data to the terminal represented by station TERM1.

Communication between nodes can be over a point-to-point BSC line, a multipoint BSC line, a Local Communications Controller ring, or an HDLC line. The flow of messages is somewhat different for each type of connection. The chapter "Sample Configurations" on page 269 contains an example of each type of connection. Each example explains what station definitions are required and gives a detailed description of the message flow.

Message Priority

The sender specifies the priority of a message as an operand of the SEND instruction. Priorities range from 1 (highest priority) to 127 (lowest priority). The default priority, which all the IOCPs and other Communications Facility programs use, is 127.

When the message dispatcher places a message on a queue, it does so according to the message's priority. Suppose that a station's message queue contains messages of priorities 50 and 100. A message with priority less than 50 will be placed at the top of the queue, and a message with priority greater than 100 will be placed at the end of the queue. A message with priority 50 will be placed after the older priority 50 messages.

The RECEIVE instruction always transfers the first message on a queue. Because messages are queued by priority, they are received by priority.

Disk-Queued Messages

When you define a station, you can specify that the lowest-priority (priority 127) messages sent to it are to be queued on disk. Each station that is to have a disk queue requires a separate data set. The chapter "Creating Disk-Queue Data Sets" on page 207 explains how to create them.

To use disk queues, you must start the version of the control program that supports disk queuing, $.CFD. Starting a station that has a disk queue also opens its disk-queue data set. When a priority 127 message is sent to the station, the message dispatcher writes the message to the disk-queue data set. All messages of higher priority are queued in processor storage.

Stopping or halting a station with a disk queue also closes its disk-queue data set. If the disk queue contains messages that have not yet been received, those messages are retained and are available when the station is next started. If the Communications Facility terminates abnormally, storage-queued messages are lost, but disk-queued messages are not; they are available when the system is restarted even though the data sets were not closed.

If you don't want a station to receive disk-queued messages from a previous session, use the *cold start* option. This option causes any disk-queued messages from a prior session for the station to be discarded before the station can receive messages.

Controlling Message Flow

You control message flow by issuing commands. The start command enables the flow of messages to and from a station. The stop and halt commands disable the flow of messages to and from a station. The link command defines or changes the destination of messages sent by a station. The modify command changes the attributes of a station, some of which affect message flow.

Starting a station creates its station block in S$POOL and opens its disk-queue data set (if it has one). The station can send messages, and messages can be sent to it.

Stopping a station flags its station block as inactive and closes its disk-queue data set (if it has one). Stopped stations can send messages, but those that are managed by IOCPs or other Communications Facility programs don't. Messages can't be sent to a stopped station. There are two exceptions to this rule: status messages can be sent to a stopped station; any type of message can be sent to a stopped message station.

A station may have messages on its queue when it is stopped. Those queued in processor storage can be received. If they aren't received, they're available when the station is restarted. Pending disk-queued messages can't be received.

Halting a station closes its disk-queue data set (if it has one), discards any pending storage-queued messages, and deletes the station's control block from S$POOL. The station is in the same state as if it had never been started.

You can use the link command to change the direct link vector of a station. You can change the link at any time, whether the station is active or not.

You can use the modify command to set a station on input hold or output hold. When a station is on input hold, all messages it sends are discarded. The messages are not undeliverable, as discussed in the next section; they are thrown away. When a station is on output hold, messages can be sent to it, but it can't receive any messages from its queue.

Undeliverable Messages

The message dispatcher may be unable to deliver messages, for various reasons:

- The destination station does not exist (it has no station block).
- The destination station is stopped.
- The destination is not known; the message was sent with a null destination, and the origin station doesn't have a direct link vector.
- The destination is a remote station, and the station that represents the remote node does not exist or is stopped.

When the message dispatcher can't deliver a message, it puts the message on the queue of the station named $.WASTE. If there is no such station, the message dispatcher discards undeliverable messages. If station $.WASTE has a disk queue, all messages to $.WASTE are queued on disk, regardless of their priority.

The use of $.WASTE is not restricted to the message dispatcher. Any program can send messages to $.WASTE. The program dispatcher sends messages to $.WASTE for the reasons described in the section "Undeliverable Transactions" on page 33. No other Communications Facility programs use $.WASTE. They discard invalid data and issue a log message to report the error.

The easiest way for you to handle undeliverable messages is to define $.WASTE as a message type station with a disk queue, and use utility program $.UT2 to process the undeliverable messages. $.UT2 lets you obtain a listing of messages and purge them or send them to other destinations. If you use $.WASTE, you should assign it a disk queue and use the disk-queuing version of the control program. If you don't, the message buffer pool may fill up with undeliverable messages.

You can handle undeliverable messages in other ways by writing a program to receive and process undeliverable messages. The *Programmer's Guide* describes the format of the messages sent to $.WASTE.

Don't define $.WASTE as a remote station. If you do, it won't be used; undeliverable messages will be discarded.

Message Types

There are five types of Communications Facility messages: data, transaction, command, status, and log.

A *data message* contains data sent from one station to another. The data is meaningful only to its destination. Routing of data messages is described in "Message Flow within a Node" on page 6 and "Message Flow between Nodes" on page 8.

A *transaction message* contains information that causes execution of a program (called a *transaction-processing program*) somewhere in the configuration. This information is in the first part of the message, which is a fixed-format area called the *transaction header*. A transaction message may also contain data to be passed to the transaction-processing program. The initial destination of all transaction messages is the program dispatcher, the component of the Communications Facility that manages transaction-processing programs. Transaction messages are routed to it, just like data messages. It receives transaction messages and handles them as explained in the section "Program Dispatcher" on page 25.

A *command message* contains a command, and is sent to the command processor. A command message can be routed between nodes, just like a data message. When it reaches the node where it is to be processed, it is routed to the command processor task of the control program.

A *status message* contains control information, such as an X.25 control message or a command, that tells a program to stop or halt. A status message is routed like a data message. If the status message is an X.25 control message, it can also be routed to an alternate link as described in "Linking Stations" on page 65. When a program receives a status message, the RECEIVE instruction completes with a unique status condition (+6).

A *log message* is an error or informational message to be sent to the system log. The message dispatcher doesn't route log messages. They are routed directly to the log processor task of the control program.

Command Processor

The command processor processes the commands, called CP commands, summarized in Figure 5. These commands allow you to control your Communications Facility system and display information about it.

Command	Activity
DEF	Define a new station
F	Modify attributes of an existing station
FILE	Display or modify the disk-queuing parameters of a station
H	Halt a station
HELP	List CP commands and their functions
LINK	Define a connection between two stations so that messages may be routed between them
P	Stop a station
Q	Display information about stations, BSC lines, EDX terminals, X.25 connections, and EDX-SNA PUs and LUs
READ	Execute commands contained in a data set
S	Start a station
SET	Assign a log device for system log messages, set the node address of a Series/1, or set levels in CFBUF.
ST	Display message activity statistics and Local Communications Controller hardware statistics
V	Remove a node from the Local Communications Controller ring

Figure 5. Summary of Command Processor (CP) Commands

The command processor and the program dispatcher together process the commands, called PD commands, summarized in Figure 6 on page 15. These commands allow you to control the actions of the program dispatcher and display information about it.

Command	Activity
C	Change the cell identifier
CP	Send a command to a cell
F	Modify a path or transaction table entry
H	List PD commands and their functions
I	Insert a path or transaction table entry
ID	Check whether a cell is active
M	Send a message to an EDX terminal
P	Stop a path or transaction
Q	Display the transaction and path tables, remote disk definitions, and scheduler entries
R	Remove a path or transaction table entry
RC	Set retry counts for program load
S	Start a path or transaction
T	Send the system time and date to a cell
TRAC	Start or stop a trace of transactions
TRAN	Send a transaction
UP	Set user program ($.UP*xxxx*) load mode

Figure 6. Summary of Program Dispatcher (PD) Commands

There are several ways of entering commands. You can enter them from a terminal by pressing ATTN and entering CP or PD followed by the function and parameters you want. For example, to start a station named APROG:

> CP S APROG

Commands can also be read from a data set, entered as transactions, issued by programs, or entered with the diagnostic aid utility, $.UT1.

A few Communications Facility commands can be entered only from a terminal. They are summarized in Figure 7.

Command	Activity
CPRSTART	Restart the Communications Facility
GOTEST	Begin execution of a transaction program in test mode
TRAN	Issue a transaction
WSC	Start work session controller terminal

Figure 7. Summary of Other Commands

The *Operator's Guide* gives the syntax and an example of each command and more information about the different ways of entering commands.

The command processor also processes commands sent from a Communications Manager system over a Local Communications Controller ring. These commands allow a Communications Manager operator to start, stop, and obtain information about Communications Facility stations. The commands are described in the *Communications Manager Operator's Guide.*

The command processor consists of a task in the control program and a set of transient programs, one for each command. The task receives control when a command is issued. It translates the command to uppercase, extracts the verb (*xxxx*) from the command, and loads the transient program named $.CP*xxxx*. The program processes the command and terminates, returning control to the initial task, which can then receive another command.

You can add your own commands to the Communications Facility. The *Programmer's Guide* explains how to write a command-processing program and includes an example program.

Log Processor

The log processor processes log messages issued by Communications Facility programs. It builds a formatted message and sends it to the system log. The message contains the time of day, a message ID, the name of the issuing program, optional variable information passed by the issuing program, and fixed text taken from the message data set $.SYSMSG.

The *Messages and Codes* gives a detailed description of the format of log messages. It also lists and explains all the log messages issued by Communications Facility programs.

There is one system log in each node. The default is the terminal from which the control program is loaded. You can use the CP SET LOG command to assign the system log to an EDX device or to a Communications Facility station, or to turn off logging.

When the system log is an EDX device, the log processor shares the device with other EDX tasks. If the device is busy, the log processor and the task that issued the log message wait until the device is available. When you assign the system log to an EDX device, the device should be one that is used only briefly (or not at all) by other EDX tasks.

Log messages issued by the command processor in response to a command entered at a terminal are displayed at that terminal and sent to the system log. All log messages issued by the program dispatcher are displayed at the terminal where the program dispatcher was started and sent to the system log.

You can process log messages however you want by writing a program that assigns the system log to its station and then receives and processes all log messages. The *Programmer's Guide* explains how you can use this technique to provide a central system log for a multinode configuration.

When you assign the system log to a Communications Facility station, you should assign a disk queue to that station. If the log station doesn't have a disk queue, the system message pool can be flooded with log messages, and a deadlock can occur. For example, if the log device is one of several 3270 terminals, the system message pool may fill up with log messages that aren't being received. This happens because the program that manages the terminal is waiting for space to send a message to some other terminal. The disk queue that you assign to a log station should be large enough to avoid losing log messages during periods of high activity.

Message Buffer Pool (CFBUF) Usage

The Communications Facility monitors the use of storage in the message buffer pool (CFBUF). It allows you to regulate the flow of messages from a station, depending on how much storage in CFBUF is in use.

CFBUF Usage Classes

When you define a Communications Facility station, the station is automatically assigned to one of two CFBUF usage level classes, class 1 or class 2. Class 1 stations can use CFBUF up to level 1 and class 2 stations can use CFBUF up to level 2. Class 1 is the default class for all Communications Facility stations. Messages can be sent from the station as long as usage of CFBUF has not exceeded the level allowed for the station's usage class. Messages with unknown origins are treated as messages coming from usage class 1 stations.

You can use the CP Q NET command to display a station's usage class. You can use the CP F CLASS command to change the station's usage class. These commands are described in the *Operator's Guide*.

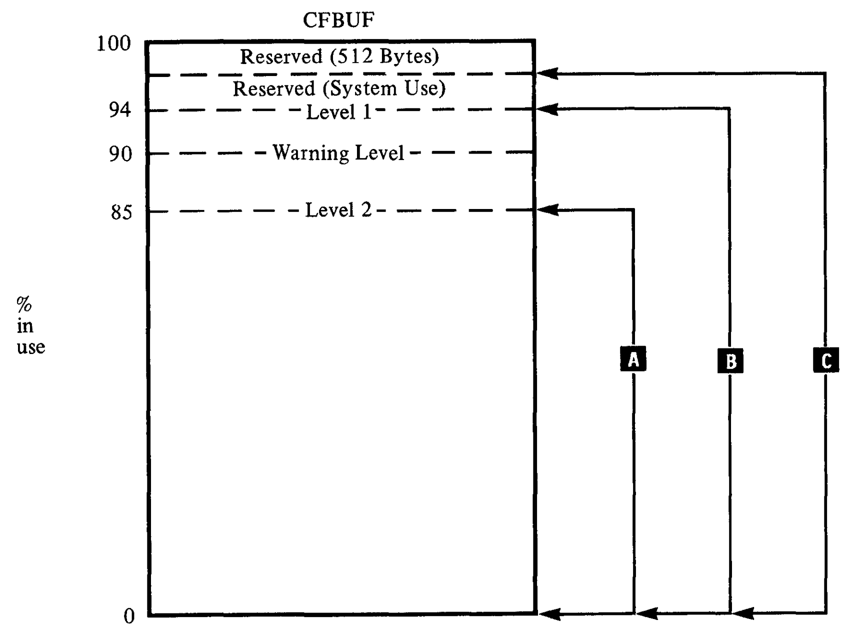
CFBUF Usage Levels

The total amount of storage available in the message buffer pool (CFBUF) is divided into three levels of usage, as shown in Figure 8 on page 18. When you first start the Communications Facility, CFBUF is below level 2. A small amount of storage in CFBUF is used for two system buffers and a control block, but most of the storage is available. As messages are sent and more storage is used but no storage is freed, CFBUF usage moves above level 2. If storage is still not freed, CFBUF usage level moves above level 1.

When CFBUF usage is below level 2, messages can be sent from stations with usage class 2 or class 1. When CFBUF usage exceeds level 2 but is below level 1, only messages from usage class 1 stations can be sent. When usage exceeds level 1, only CP commands can be issued. You should issue commands to free message storage when usage exceeds level 1. For example, you can stop stations that have accumulated messages on their queue.

The default value for level 1 is set by the system to prevent the last 512 bytes of CFBUF storage from being used up. You can not modify level 1.

The default value for level 2 is 85% of CFBUF in use. You can use the CP SET CFBUF command to set the value for level 2. You can not set level 2 to allow more CFBUF to be in use than allowed by level 1. The CP SET CFBUF command is described in the *Operator's Guide*.



A Available for data, transaction, or status messages when origin station has usage class 2.

B Available for data, transaction, or status messages when origin station has usage class 1, or when origin station is not known.

C Available for command messages, regardless of origin.

Figure 8. CFBUF Usage Levels

Warning Level

The Communications Facility can log an error message as a warning when more than a certain percentage of the CFBUF is full. The warning processor of $.CF issues an error message when the usage of CFBUF exceeds the warning level. The default warning level at which a system log message is issued is set at 90%. You can use the CP SET CFBUF command to specify a warning level. After a warning message is issued, another warning message is not issued until CFBUF usage level falls 10% below the warning level and then exceeds it again.

Displaying Current Usage

You can use the CP Q CFBUF command to display the current usage level thresholds and the current usage of CFBUF. This command also displays the highest percentage of CFBUF in use since the Communications Facility was loaded. The command is described in the *Operator's Guide*.

I/O Control Programs

The Communications Facility includes several device-support programs, called I/O control programs or IOCPs.

Functions of an I/O Control Program

An IOCP manages devices and the Communications Facility stations that represent those devices. Each IOCP manages devices of a particular type, such as a terminal, printer, or communication line. An IOCP can handle multiple devices. It consists of a main task and one or more subtasks for each individual device.

The main task receives and processes commands to start, stop, and halt the stations that the IOCP manages. The main task attaches at least one subtask when a device is started. The subtask(s) manages all communication with the device. It reads data from the device and sends that data into the system in the form of Communications Facility messages. It receives messages from its station's queue and writes them to the device.

A subtask consists of reentrant code and a task control block. The task control block or a pointer to it is contained in the station block that represents the device.

Summary of Supplied I/O Control Programs

Figure 9 on page 20 shows the IOCPs that manage host system connections, terminals, printers, and emulated PC disks.

The IOCPs that manage host system connections are:

- $.IO0AD0 manages communication over a System/370 channel attachment. The Series/1 appears to the host as a locally attached 3272 control unit with attached display stations and printers. There is a station to represent each emulated display station or printer.

- $.IO0AE0 manages communication over a BSC multipoint line. The Series/1 appears to the host as a 3271 control unit with attached display stations and printers. There is a station to represent each emulated control unit, display station, or printer.

- $.IO14E8 manages a Systems Network Architecture (SNA) connection over an SDLC line. The Series/1 appears to the host as a 3274-SDLC control unit with attached display stations and printers. In SNA terminology, it appears as a physical unit (PU) with logical units (LUs): LU-2s (display stations) and LU-1s or LU-3s (printers). There is a station to represent the PU and each LU.

The IOCPs that manage terminals, printers, and emulated PC disks are:

- $.IO0AC0 manages communication with a 3270 system over a BSC multipoint line.

- $.IO0670 manages 4978 display stations with the data stream RPQ (D02428) and 4980 display stations so they appear as 3277 display stations. This IOCP

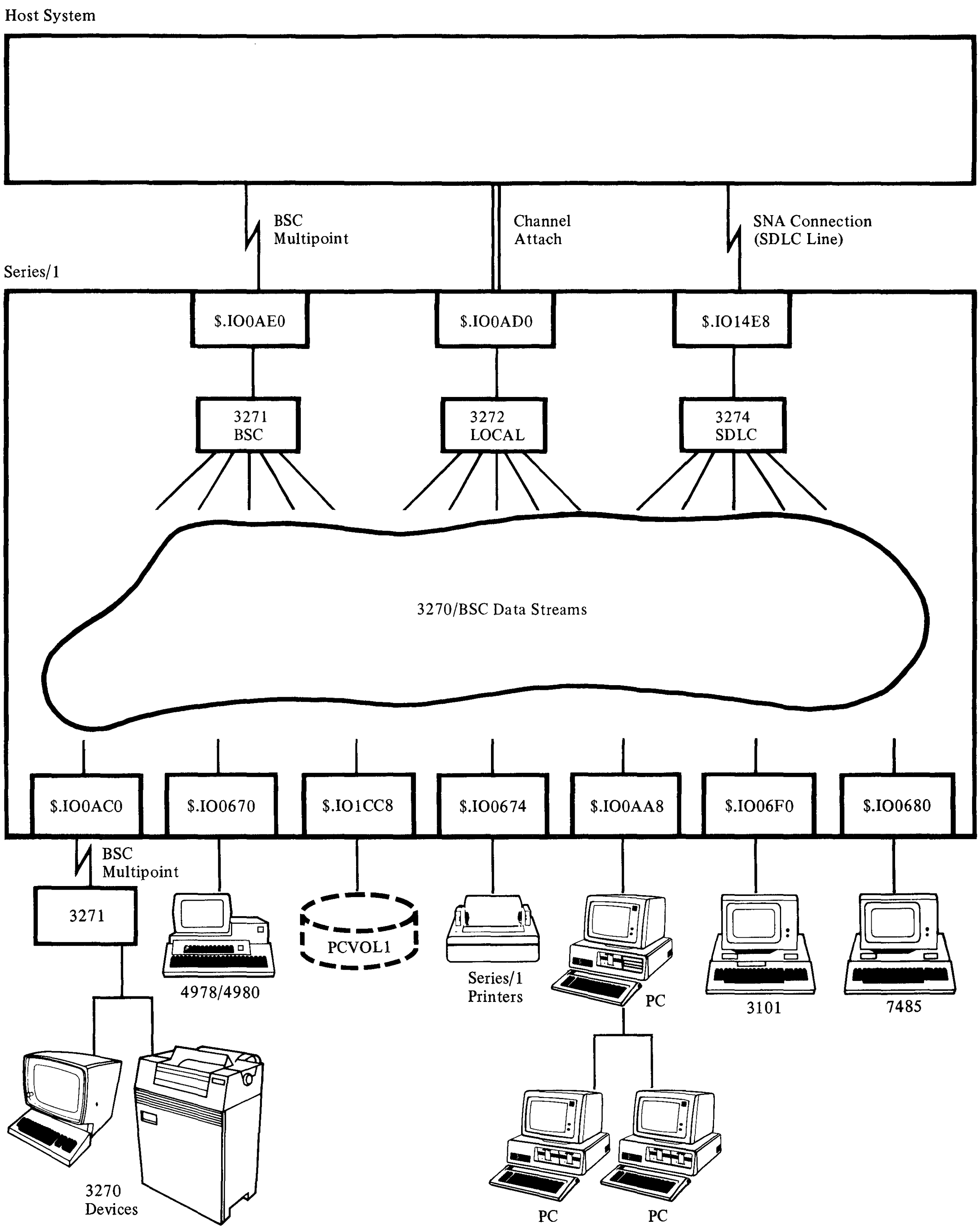


Figure 9. IOCPs that Manage Host System Connections, Terminals, Printers, and Emulated PC Disks

used to be called $.IO0678 in the previous edition (Version 2.0) of Communications Facility.

- $.IO06F0 manages 3101 terminals on a feature programmable communications adapter, a multifunction attachment, or a 4950/5170-495 terminal/host adapter so they appear as 3277 display stations.

- $.IO0674 manages 4973, 4974, 4975, 5219, 5224, 5225 , or 5262 printers so they appear as 3286 or 3287 printers.

- $.IO0680 manages 7485 terminals on a feature programmable communications adapter, a multifunction attachment, or a 4950/5170-495 terminal/host adapter so they appear as 3277 display stations.

- $.IO1CC8 allows you to use a Series/1 data set so that it appears to the *gateway* PC as a local disk device. The gateway PC is a Personal Computer connected to the Series/1 by Series/1-PC Connect Attachment and its software support. The IOCP can control multiple volumes for multiple LANs. It handles one volume for each gateway PC.

The messages sent to and received from all these IOCPs are 3270/BSC data streams. (The channel attach and SNA IOCPs remove BSC control characters before sending data to the host and append BSC control characters to data received from the host.) You can use the Communications Facility for communication between host application programs and any of the terminals or printers over any of the three types of connection. This type of communication, called *3270 pass-through*, requires no user programming. You need only define stations to represent the real devices, define stations to represent corresponding emulated devices, and link each pair of stations to each other. You can also write application programs that are independent of the type of terminal they communicate with or the type of connection to the host program they communicate with.

Figure 10 on page 22 shows the IOCPs that manage connections between Series/1s:

- $.IO0AB0 manages communication between Series/1s on a Local Communications Controller ring. The messages it transmits may or may not be 3270 data streams. It transmits entire messages, including their headers, whatever their content. Some of the Series/1s on the ring can be Communications Manager systems. The IOCP converts message headers from the format used by one system to that used by the other.

- $.IO0AB8 manages communication between Series/1s over an HDLC line, using CCITT Recommendation X.25 packet level procedures. The connection between the Series/1s may or may not have an an intervening X.25 packet-switching network. $.IO0AB8 transmits data and X.25 control information. The data may or may not include Communications Facility message headers, depending on how stations are defined. When communications is through an X.25 network, some of the Series/1s can be Communications Manager systems.

- $.IO0AC0 and $.IO0AE0 can be used for communication between Series/1s over a BSC multipoint line. $.IO0AC0 manages 3270 systems, and $.IO0AE0 makes the Series/1 appear to be a 3270 system.

- $.IO0A10 manages communication between two Series/1s over a BSC point-to-point line. It transmits entire messages, including their headers, whatever their content.

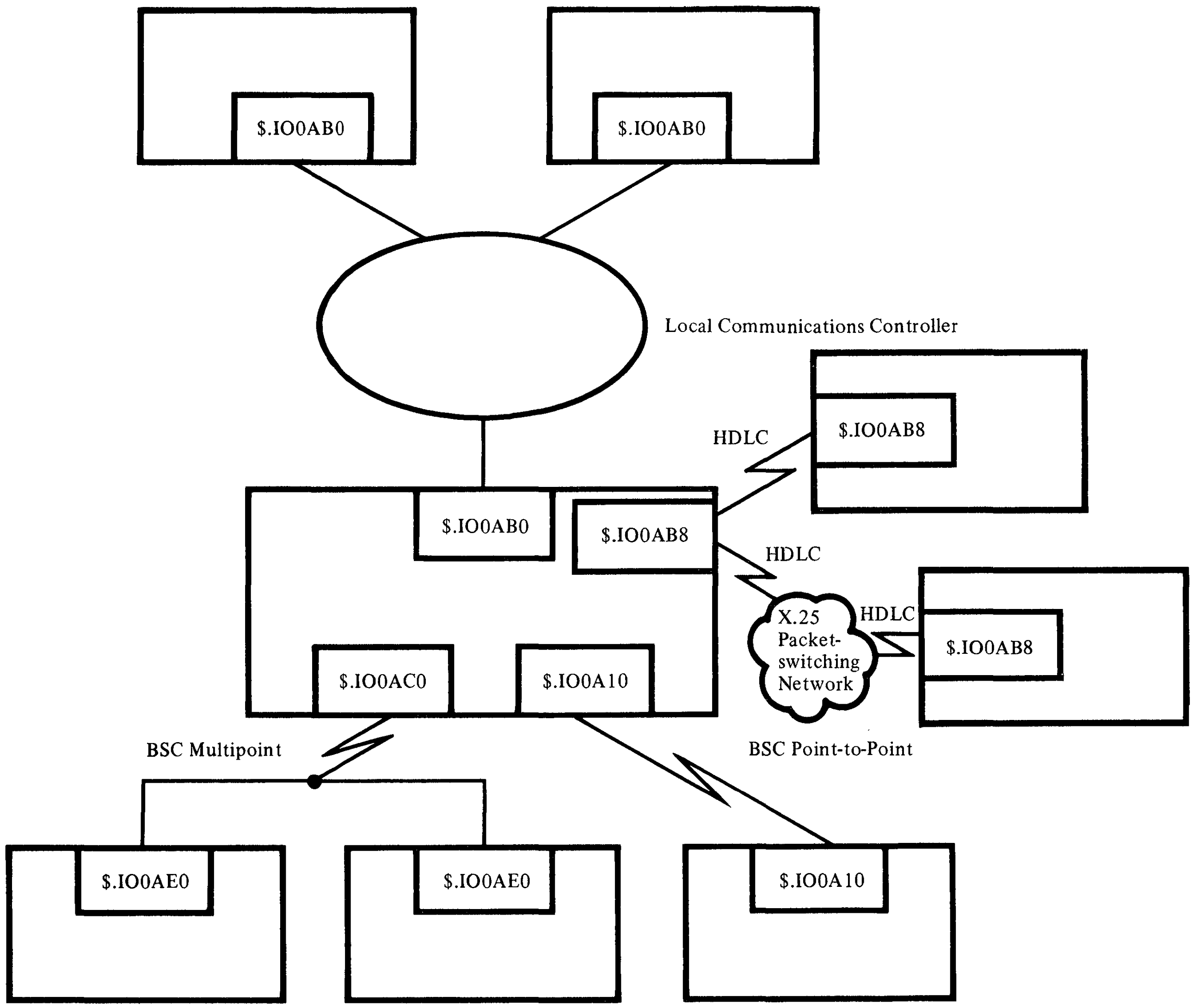


Figure 10. IOCPs that Manage Connections between Series/1s

Figure 11 on page 24 shows the IOCPs that manage communication with other systems and devices:

- $.IO0AA8 manages communication between a Series/1 and a Personal Computer (PC) connected by a Series/1-PC Connect Attachment. It allows all the PCs in the IBM PC Network to communicate with a Series/1 through a gateway PC (PC connected to the Series/1 by Series/1-PC Connect Attachment and its software support). It also allows all the PCs on a non-IBM network to communicate with a Series/1, but the PCs on your network must meet the hardware and software requirements for the type of network you use. When the gateway PC is part of a local area network (LAN), the Series/1 resources are available to all the PCs on the LAN. A LAN is created when PCs are interconnected by hardware and provided with software support to enable the PCs to communicate with each other. $.IO0AA8 can manage multiple attachments to different PCs.

- $.IO0AB8 manages communication between a Series/1 and other computers or devices that function as X.25 data terminal equipment (DTE). The Series/1 is connected by an HDLC line using LAPB protocol and X.25 packet-level procedures, with or without an intervening X.25 packet-switching network.

- $.IO0A20 manages communication between a Series/1 and a variety of other systems over a BSC point-to-point line. $.IO0A20 uses a 2770- and 3741-like protocol. The format of the messages it transmits is whatever is required by the other system.

The chapter "Using the Supplied Input/Output Control Programs" on page 71 gives detailed information about each of the IOCPs. It explains the requirements and restrictions you need to take into account when you use them.

Writing an I/O Control Program

You can extend the Communications Facility to support other devices by writing IOCPs. The *Programmer's Guide* explains how to write an I/O control program and includes an example program.

Intercepting Messages from an I/O Control Program

You can write a program that intercepts all the messages that an IOCP sends into the system from the stations it manages. You may want to do this, for example, to provide an audit trail of messages by writing them to a data set before sending them on to their destination. The *Programmer's Guide* explains how to intercept messages from an IOCP.

I/O Control Program Station Definitions

You should make an entry in $.SYSNET for each of the IOCPs you plan to use. Define them as having type "USER" and a station name the same as the program name.

IOCPs that are not defined in $.SYSNET can be started, but they can't be shut down.

Starting I/O Control Programs

You don't need to explicitly start IOCPs with a CP start command or a $L, unless you wish to control which partition the IOCP runs in. An IOCP is started automatically by the control program when the first station of the type it manages is started. The start command processor uses a station's type and subtype to determine which IOCP manages the station. For example, the type/subtype of a station that represents a 3101F on a feature programmable communications adapter is device/3101F, which is recorded as a hexadecimal code in the station

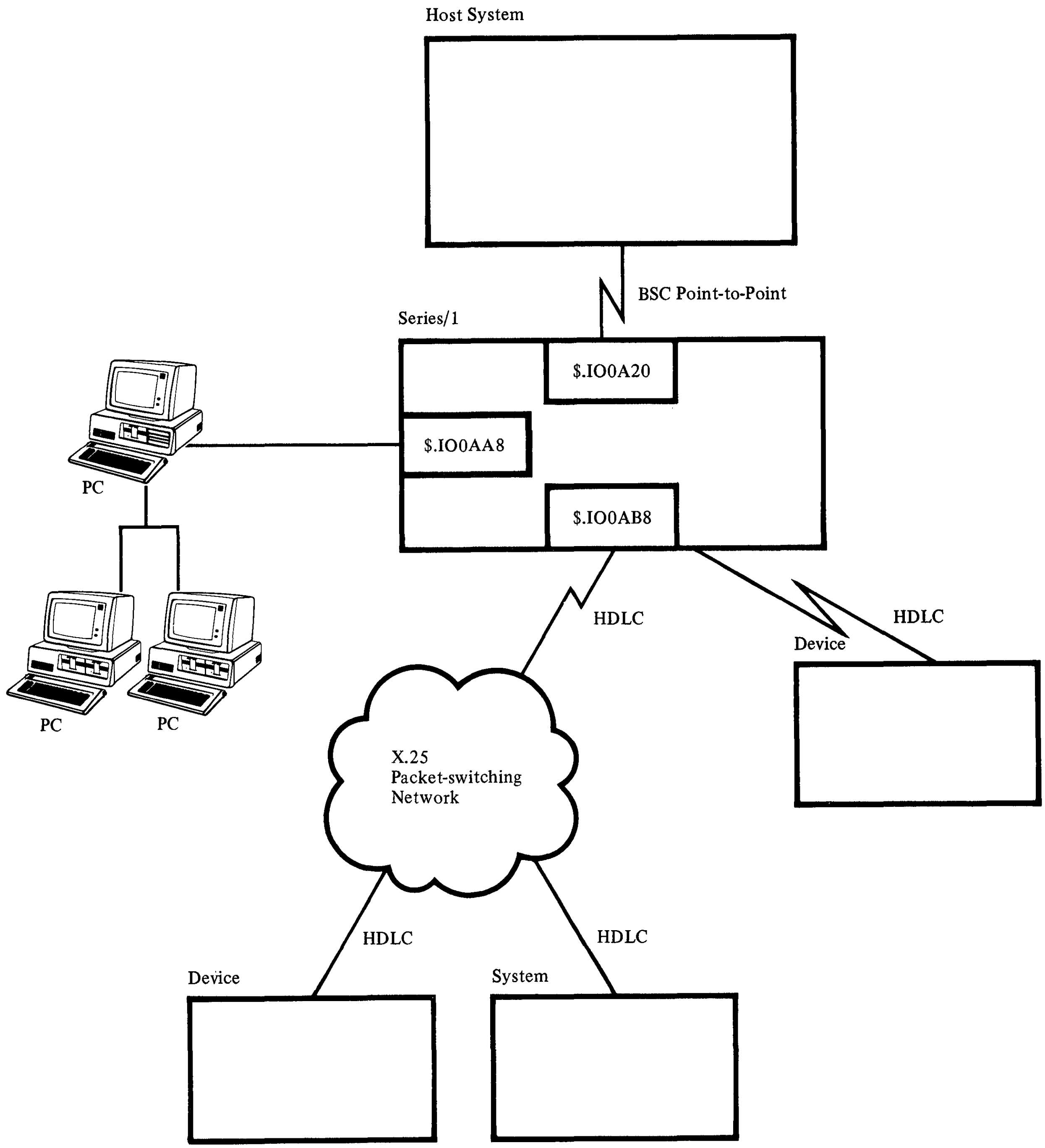


Figure 11. IOCPs that Manage Connections to Non-Series/1 Devices

definition in $.SYSNET. The code for device/3101F is X'06F3', and the IOCP that manages such stations is $.IO06F0.

If the required IOCP is not already started, the start command processor creates a station block to represent the IOCP and loads the program.

Shutting Down I/O Control Programs

You shut down an IOCP by stopping or halting its station. When you do so, the IOCP halts all the stations that it manages, deletes its station block, and terminates execution.

You can't stop or halt an IOCP if it isn't defined in $.SYSNET.

Program Dispatcher

The program dispatcher is that part of the Communications Facility that handles transaction processing. It routes transactions and manages the programs that process them.

The program dispatcher is an optional part of the Communications Facility. It is required only to support transaction-processing programs—those provided with the Communications Facility and those written by users.

The term *transaction-processing program* is used throughout this book to refer to programs that are executed in response to transactions. Note that the program may or may not—at your option—receive the text of the transaction itself. You can define transactions that just cause a program to be loaded and executed, without receiving the transaction as a message.

To the message dispatcher, the program dispatcher is simply a station that represents a program. The name of the station and the program is $.PD. The program dispatcher receives transaction messages from its message queue and performs its transaction-routing function by sending them to other stations.

The program dispatcher's actions are governed by information in the transaction messages, in a path table, and in a transaction table. The path table and transaction table contain information you specify in the $.SYSPD data set, as explained in the chapter "Creating $.SYSPD" on page 187. There is a $.SYSPD data set and, therefore, a path table and a transaction table for each Series/1 where the program dispatcher runs.

Transaction Messages

The first part of a transaction message is a fixed-format header that identifies the transaction and the cell where it is to be processed. A *cell* is a Series/1 where the program dispatcher runs or a non-Series/1 host system where transactions are processed.

The format of a transaction message is:

1-4	5-6	7-10	11-12	13-*n*
tid1	*c1*	*tid2*	*c2*	*transaction data*

where:

tid1
is the primary transaction identifier, a 4-character code that identifies the transaction.

c1
is the primary cell identifier, a 2-character code that identifies the cell where the transaction is to be processed. Two blanks or 00 means that the transaction is to be processed in the cell where it originated. Two asterisks mean that the transaction is a broadcast transaction, which is to be processed in all cells known to the program dispatcher.

tid2
is the secondary transaction identifier, whose meaning is defined by the program that processes the transaction.

c2
is the secondary cell identifier, whose meaning is defined by the program that processes the transaction. When *c2* is ??, the program dispatcher replaces it with the ID of the cell where the transaction originated.

transaction data
is whatever data is sent to the program that processes the transaction.

The minimum length of a transaction message is 6 bytes. The data portion is optional. If a transaction contains no data, the secondary transaction identifier and secondary cell identifier are optional.

Transactions are usually originated by programs, perhaps in response to input from a terminal operator. They can also be read from a data set or entered from a terminal with the PD TRAN command.

Path Table

A path table has an entry for each remote cell known to the local program dispatcher. Each entry contains a cell identifier, the name of the station that represents the path to and from that cell, and optional path attributes. Program dispatchers use this information to route transactions from cell to cell, as explained in the section "Transaction Routing" on page 27.

Transaction Table

A transaction table has an entry for each transaction to be processed in the local cell. Each entry contains a transaction identifier, the name of the program that is to process the transaction, program attributes, and transaction attributes. The program dispatcher uses this information to route transactions to the programs that process them and to manage the programs, as explained in the section "Program Management" on page 31.

Transaction Routing

The initial destination of every transaction is a program dispatcher, which routes the transaction to its ultimate destination—a transaction-processing program. The routing may require that a program dispatcher in one cell send the transaction to a program dispatcher in another cell or to a non-Series/1 host transaction-processing system. Transactions can be routed between Series/1 cells over any of the connections the Communications Facility supports. Transactions can be routed to a non-Series/1 host over a BSC multipoint line or an SDLC line (an SNA connection). They can also be routed over a channel attachment, but there are restrictions. These restrictions are explained in the Series/1-to-System/370 Channel Attach IOCP description, "Restrictions" on page 106.

A program dispatcher routes a transaction to another cell by sending it to the station that represents the path to that cell, as defined in the path table. To the message dispatcher, a transaction is a data message; it places the message on the destination station's queue just as if it were an ordinary data message.

A path station doesn't represent the physical connection (the communication line) between two cells. It represents the logical path from one program dispatcher to another or from a program dispatcher to a non-Series/1 host system. The chapter "Sample Configurations" on page 269 contains examples of configurations that include the program dispatcher. Those examples show how to define path stations and the stations that represent physical connections.

Figure 12 on page 28 is an example of a multicell configuration with path stations defined. Paths are defined between cells:

C1 and HT
C1 and C2
C1 and C3
C1 and C4, through C2
C2 and C4

The program dispatcher can route messages to PC programs on a local area network (LAN) connected through a Series/1-PC Connect Attachment. The program dispatcher routes the transaction through a path station which is a remote station representing the PC program. The PC program will receive the message with the program dispatcher header included in the message.

Because some of the paths are defined as preferred paths, there is a path from any cell to any other cell. A *preferred path* is a path over which transactions for unknown cells are routed.

A transaction that originates in cell C2 and that has C3 as its primary cell identifier is routed as follows:

1. The program dispatcher in cell C2 receives the transaction from its message queue and examines the primary cell identifier to determine whether or not the transaction is to be processed in cell C2. If the primary cell identifier were C2, 00, or two blanks, the transaction would be processed in cell C2. The primary cell identifier isn't any of those, so the program dispatcher examines its path table to determine whether or not the path to C3 is defined. It isn't, so the program dispatcher sends the transaction to the preferred path station, which represents the path to the program dispatcher in cell C1.

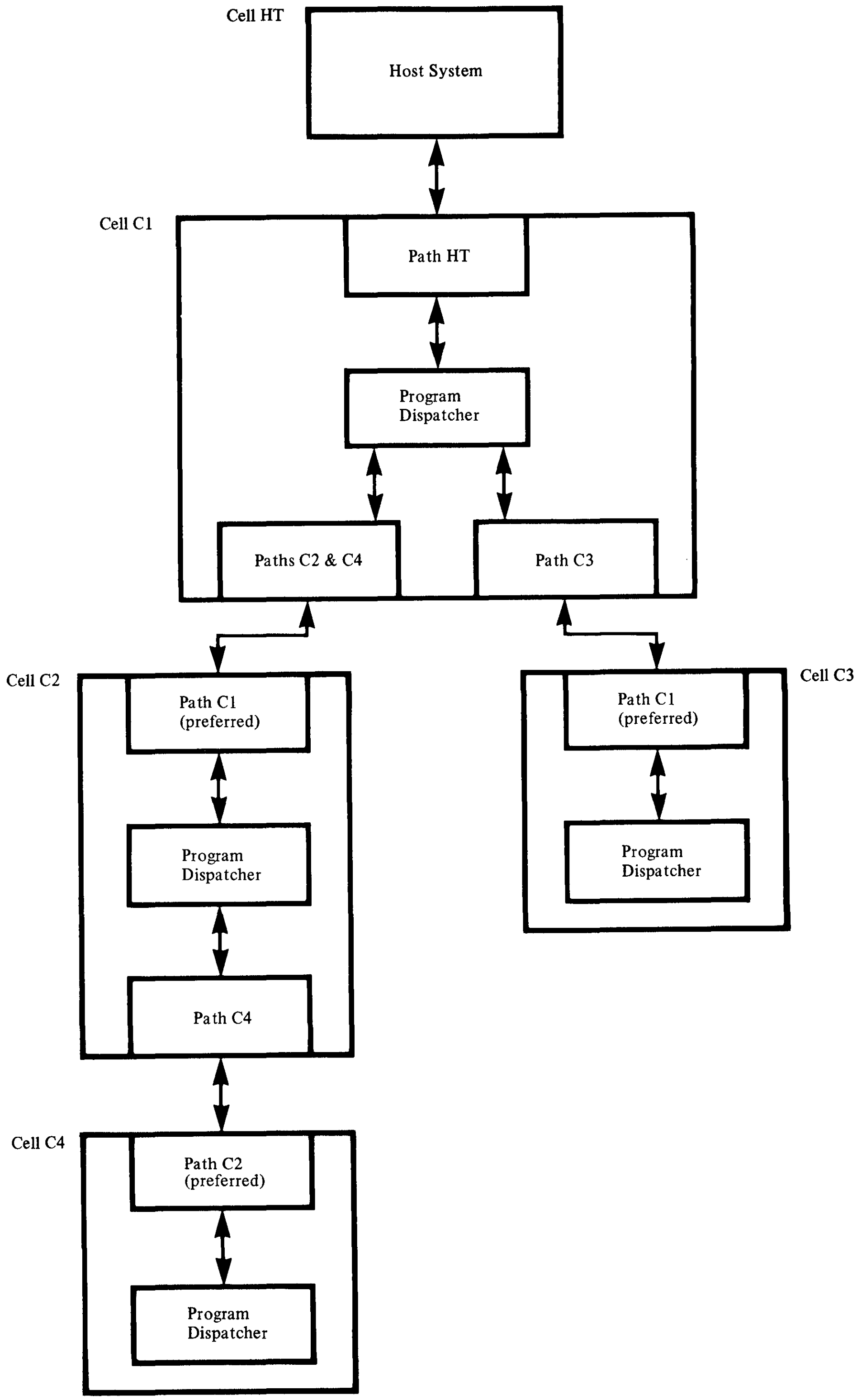


Figure 12. Example Multicell Configuration with Path Stations

2. The program dispatcher in cell C1 receives the transaction and examines the primary cell identifier. The transaction is not to be processed in cell C1, so the program dispatcher examines its path table. The path to C3 is defined, so the program dispatcher sends the transaction to the station that represents the path to the program dispatcher in cell C3.

3. The program dispatcher in cell C3 receives the transaction and examines the primary cell identifier. The transaction is to be processed in this cell, as described in the section "Program Management" on page 31.

Broadcast Transactions

A *broadcast transaction* is one whose primary cell identifier is **. When a program dispatcher receives a broadcast transaction, it processes the transaction in the local cell; sends it unchanged to the preferred path station, if one is defined; and sends it to each other path station with the ** replaced by the remote cell identifier.

Any transaction can be a broadcast transaction. Some of the PD commands and system transactions use the broadcast facility. For example, the PD M (send message) command and the SYST MS transaction processor use it to send a message to all EDX terminals in all cells.

3270 Data Stream Transactions

Transactions are usually character strings built by a program, read from a data set, or entered from an EDX terminal with a PD TRAN command. Transactions can also be 3270 data streams. These are most likely to originate from a Communications Facility terminal (3277, 4978, 4980, 3101) that is linked to the program dispatcher. The programs that process such transactions may need to receive the entire data stream; the 3270 control information at the beginning shows which key was pressed to terminate the input and the position of the cursor. The program may also need to know the origin station name so that it can direct its response back to that terminal.

To make routing of a data stream transaction possible, you must define it as a data stream transaction. If the transaction is processed in a cell other than the one in which the transaction originated, you must also define the paths over which it is routed as data stream paths. Then when the program dispatcher sends the transaction to a path station or to the transaction-processing program, it sends the entire data stream and specifies the origin station name it received. If you don't define a transaction or path as data stream and the program dispatcher receives a data stream, it removes the BSC and 3270 control information before sending the transaction and specifies itself (the station named $.PD) as the origin. The information removed is whatever precedes the first data field and the BSC control character (ETX) at the end.

There is another data stream consideration—IOCPs that manage a BSC multipoint line, a channel attachment, or an SNA connection send and receive 3270/BSC data streams. You don't define these types of paths as data stream paths. The program dispatcher identifies them as such by the path station's type. It appends the required 3270/BSC control information to transactions before sending them to the path station. It removes 3270/BSC control information from transactions received from the path station.

Transaction Prefix

Some host transaction-processing programs require an 8-character transaction identifier, rather than the 4-character one used by the Communications Facility. When you define the path to such a host system, you define a 4-character prefix. The program dispatcher appends the prefix to each transaction sent to that path station.

Binary Transactions

The data portion of transactions may be binary (any value from X'00' to X'FF') rather than characters. You may, for example, transmit object programs or storage dumps as transactions. Binary data can be transmitted between Series/1s and between a Series/1 and a host system connected through a channel attachment or an SNA connection. Host systems accessed through a multipoint BSC line may or may not support transmission of binary data.

Binary data sent over a BSC line must begin with the control characters DLE/STX (X'1002'), which signal start of transparent text. When you define a path as a data stream path, and the connection is a multipoint BSC line (either between Series/1s or between a Series/1 and a host system), binary transactions must begin with DLE/STX. In all other cases, either the program dispatcher or the IOCP that manages the connection provides the required control characters.

If a host system does not support transmission of binary data, you must specify when you define the path that data conversion is required.

When you specify data conversion, the program dispatcher converts the data portion of transactions sent to the path station. It doesn't convert the transaction header, the preceding 3270/BSC control information, or the ETX at the end.

The transaction data is converted as follows:

- Bit 0 of each byte that has a value less than X'40' is set to 1.
- Bit 0 of each byte that has a value of X'FF' is set to 0.
- A control byte is inserted after each group of 7 data bytes and after the last group of 7 or fewer data bytes. Bits 1-7 of a control byte correspond to the preceding 7 data bytes; the control bit is set to 1 if the data byte was modified. Bit 0 is set to 1 if the control byte has a value less than X'40'.

The following example shows the hexadecimal value of transaction data before and after conversion. The underscored bytes are control bytes.

```
Received:  4F F3 32 18 92 61 40 00 C2 96 FF 01 72 49 61 20 19
Sent:      4F F3 B2 98 92 61 40 98 80 C2 96 7F 81 72 49 4C 61 A0 99 83
```

Notice the last control byte, which is for a group of 3 data bytes. The control bits are right-justified. They correspond to the last 3 bytes of the group, not the first 3 bytes.

When the program dispatcher receives a transaction from the path station, it removes the control bytes and restores the modified data bytes. The host transaction-processing program must perform corresponding conversions of the data it receives and sends.

Program Management

The program management functions of the program dispatcher make it possible to optimize the use of processor storage. A transaction-processing program is loaded on demand; that is, when a transaction that it processes is entered. You can write the program to terminate when it has completed its work or when it is told to terminate because another program needs storage. The program dispatcher uses a high-speed loader to load programs, so it is feasible to segment transaction-processing applications into small programs that occupy storage only while they have work to do.

When the program dispatcher receives a transaction in the cell where it is to be processed, it finds the primary transaction identifier in its transaction table and proceeds according to the specifications in that entry. The entry includes the name of the program that is to process the transaction and its type. The type indicates whether or not the program dispatcher is to create a station block to represent the program, send the transaction to the station created, or load the program.

Creating a Station Block

You can define a transaction that simply causes a program to be loaded, and that program may or may not send and receive messages. Alternatively, the program can receive the transaction itself as a message and process the transaction.

If the program does process its transactions, or if it sends and receives messages, it must have an associated station block and message queue. The program name specified in the transaction table entry is also the station name.

When the program dispatcher is to create a station block and the station block doesn't already exist, it is created in one of two ways. If the station is defined in $.SYSNET (the network configuration data set), the program dispatcher issues a start command for the station. The start command processor creates a station block; opens the station's disk queue, if it has one; and loads the program, if the station type is user (the type generally used to represent a program). If the station is not defined in $.SYSNET, the program dispatcher creates a station block with station type "message" and station name the same as the name of the program.

Sending the Transaction Message

The program dispatcher sends the transaction to the transaction-processing program if the transaction table entry so specified. The send causes the transaction to be placed on the message queue of the station that represents the program.

Loading the Transaction-Processing Program

You can define transactions that cause a program to be loaded each time a transaction is entered. You should do this only for programs that terminate execution after doing the work required for one transaction.

You can define transactions that don't cause a program to be loaded. You might do this, for example, to accumulate a queue of transactions to be processed later by a program that you load yourself.

What you will usually specify is that the program is to be loaded if it is not already in storage. It may have been loaded for a previous transaction and still be in storage, or it may have been loaded for this transaction by the start command, as discussed in the section "Creating a Station Block" on page 31.

The program dispatcher does not use the EDX loader ($LOADER). It uses a high-speed loader distributed with the Communications Facility, which resides in the system storage pool (S$POOL). When the program dispatcher is started, it determines the location and size of each transaction-processing program, and saves that information in the transaction table. Therefore, the high-speed loader doesn't need to search directories for the programs it loads.

If the programs the high-speed loader loads have overlays or data sets, they must be prefound. You can specify that the program dispatcher is to do this when it is started, or you can use the EDX utility $PREFIND to do it. Because the high-speed loader is resident, the amount of storage required to load a program is just the storage the program itself occupies.

The program dispatcher uses the high-speed loader to load the transaction-processing program into the partition you specify. If the load fails because there isn't enough storage, the program dispatcher tells an active transaction-processing program to terminate execution, waits 200 milliseconds, and retries the load. It performs this retry procedure 30 times or until the load succeeds. After each ten attempts, it receives and processes any transactions on its message queue before retrying the load. You can use the PD RC (set retry counts) command to alter the number of load attempts. You can also specify, for individual transactions, that the load is to be retried until it succeeds or until the program dispatcher is shut down.

When there is not enough storage to load a program, only those programs that you have designated as *purgable* are told to stop.

Rerouting Transactions

A transaction table can contain reroute entries, as well as transaction definition entries. A *reroute entry* contains a transaction identifier and the identifier of the cell where the transaction is to be processed. You can use reroute entries to change the routing of transactions without changing their content.

Suppose, for example, that you have a multicell configuration and an application where transaction PRT1 is processed in cell C1. The transaction's primary cell identifier is C1, and the transaction table in cell C1 has an entry that defines transaction PRT1. To cause PRT1 transactions to be processed in cell C3, replace the PRT1 definition with an entry that reroutes PRT1 to cell C3, and add a PRT1 definition to the transaction table in cell C3.

Controlling Transaction Flow

Transactions flow according to the transaction, path, and reroute entries you define. While the program dispatcher is active, you can issue PD commands that change the flow. You can insert or delete path and transaction table entries, and hold transactions. All of these actions affect only the path table and transaction table in processor storage. To make permanent changes, you must modify the $.SYSPD data set from which these tables are built.

You use the PD I (insert) command to add an entry to a path or transaction table and the PD R (remove) command to delete an entry. You can't add entries to a table unless you reserved space for them with your $.SYSPD definitions.

You use the PD P (stop) command to inactivate a path or transaction table entry and the PD S (start) command to activate an entry. A transaction is undeliverable if its transaction definition is inactive or if a path over which it is to be routed is inactive.

Holding transactions is the most useful method of controlling transaction flow. If there is a problem with a transaction-processing program or with a communication path, you can hold transactions until the problem is corrected, and then release them for processing. You use the PD F (modify) command to set or release a hold on a path or transaction.

When the program dispatcher finds that a path to a cell is on hold, it creates a station named $.PH*xx* (where *xx* is the cell identifier) and sends the transaction and all subsequent transactions for that path to station $.PH*xx*.

When the program dispatcher finds that a transaction is on hold, it creates a station named $.PH*xxxx* (where *xxxx* is the transaction identifier) and sends the transaction and all subsequent transactions with that identifier to station $.PH*xxxx*.

When you release a held path or transaction, the program dispatcher receives transactions from the appropriate $.PH station's message queue, processes them, and deletes the station block.

Undeliverable Transactions

For various reasons, the program dispatcher may be unable to deliver a transaction. It handles undeliverable transactions in various ways, depending on why it can't deliver the transaction. The program dispatcher may dispose of an undeliverable transaction in these ways:

Alternate: The transaction is sent to the origin station's *alternate link*, if it has one and the alternate link station is started. You use $.CONFIG or the CP LINK command to define alternate links. This type of connection between stations is called an *alternate link vector*.

$.UP: The program dispatcher issues an EDX LOAD instruction for program $.UP*xxxx*, where *xxxx* is the transaction identifier. It doesn't send the transaction to the program; rather, it discards the transaction. Neither does it create a station to represent the program. This facility simply makes it possible for you to load programs in response to transactions without having defined the transactions. If you don't need this capability, you can use the PD UP (user program) command to disable it.

$.WASTE: The transaction is sent to $.WASTE (the station for undeliverable messages), if $.WASTE exists.

Discard: The transaction is discarded.

The rest of this section lists reasons why transactions are undeliverable, and tells which dispositions are attempted, in chronological order.

- The transaction is not a 3270 data stream, and is less than 6 bytes long. Disposition is alternate, $.WASTE, discard.

- The transaction is a 3270 data stream that contains only 3270/BSC control information; it contains no data. These short data streams result from a terminal operator's pressing the CLEAR key or a PA key. Disposition is alternate, $.WASTE, discard.

- The program dispatcher can't route the transaction to the cell where it is intended to be processed. Either the path is not defined and there is no preferred path, or the path is defined but the path table entry is inactive. Disposition is alternate, $.WASTE, discard.

- The path is known and active, but the path station isn't started. Disposition is discard.

- The transaction is in the cell where it is to be processed, but its transaction identifier is not defined in the transaction table. Disposition is alternate, $.UP, $.WASTE, discard.

- The transaction is defined in the transaction table, but the entry is inactive; or the transaction-processing program can't be loaded. Disposition is $.WASTE, discard.

Tracing and Testing Transactions

During application program development or problem analysis, you may need a record of all transactions processed by the program dispatcher. You may also need to execute transaction-processing programs under control of the EDX $DEBUG program. The program dispatcher includes these facilities; they're described in the *Programmer's Guide.*

Program Dispatcher Station Definitions

You must define some stations for the program dispatcher's use in $.SYSNET, the network configuration data set. The stations you must define are the path stations and the station that represents the program dispatcher itself. Define the program dispatcher station with name $.PD and type "user."

You need to define a station to represent a transaction-processing program *only* if you want the station to have a disk queue or you want its type to be other than "message." If you define the station in $.SYSNET because it has a disk queue, give it type "message." Then the start command processor will create the station block and open the disk queue, and the program dispatcher will load the program. If you define the station as type "user" (the type generally used to represent a program), then the start command processor will load the program; it may or may not be loaded into the partition you specified in the transaction definition, and you lose the benefits of the high-speed loader.

The transaction-processing program does not always have to be a program. The "program" name you specify in the transaction definition can be the name of a station of any type; for example, it might be the name of a station that represents a terminal. If you define transactions so that they are routed to terminals, be sure to specify a transaction type that doesn't cause the program dispatcher to try to load a program.

If you plan to hold transactions and you want a hold station to have a disk queue, define it in $.SYSNET as having type "message." *Hold stations* are those whose names begin with $.PH, as discussed under "Controlling Transaction Flow" on page 33.

If you define transaction-processing program stations or hold stations in $.SYSNET, do it before you start the program dispatcher. The program dispatcher determines whether or not these stations are defined in $.SYSNET when it is started, not when it needs to use the stations.

Starting the Program Dispatcher

You start the program dispatcher by starting the station that represents it, $.PD. When the program dispatcher is started, it loads the high-speed loader into the system storage pool and reads and processes its data set, $.SYSPD. The data set can contain commands, path table definitions, transaction table definitions, and transactions.

In the first part of $.SYSPD, you can place any commands you want to be executed when the program dispatcher is started—for example, commands to start the stations that represent physical connections between cells. The program dispatcher sends the commands to the command processor.

The program dispatcher builds its path table from the definitions in the second part of the data set. It issues a start command for each path station and sets a link for it, if one is required. If the path station represents an SNA logical unit or an emulated or real 3277 terminal, messages it sends are undeliverable unless it has a direct link. The program dispatcher links these types of path stations to its own station, $.PD.

The program dispatcher builds its transaction table from the definitions in the third part of the data set. It determines the location and size of each transaction-processing program and, if you so specified, prefinds the program's data sets and overlays.

In the fourth part of the data set, you can place any transactions that you want issued when the program dispatcher is started.

Shutting Down the Program Dispatcher

You can shut down the program dispatcher and the programs it manages by halting station $.PD. The program dispatcher tells each of the programs defined in its transaction table to halt, frees the storage occupied by the high-speed loader, deletes its own station block, and terminates execution.

If you stop station $.PD, the same actions occur except that the program dispatcher doesn't tell the programs it manages to stop.

Communications Facility Transaction-Processing Programs

Transaction-processing programs provide some of the services available with the Communications Facility. Those programs run under control of the program dispatcher.

System Transaction

The system transaction (SYST) performs a variety of functions, which are summarized in Figure 13. The program dispatcher uses some of these transactions to process PD commands. You can use any of them.

Transaction	Activity
SYST CP	Send a command to a cell
SYST HL	Load a program
SYST ID	Check whether a cell is active
SYST MS	Send a message to an EDX terminal
SYST RC	Set retry counts for program load
SYST RH	Release a held transaction or path
SYST SP	Start a remote disk
SYST ST	Check whether a station exists
SYST TI	Set the system time and date in a cell
SYST TR	Start or stop a trace of transactions
SYST WH	Check whether a program is loaded
SYST UP	Set user program ($.UP*xxxx*) load mode

Figure 13. Summary of SYST Transactions

The program dispatcher itself ($.PD), rather than a separate program, processes the SYST transactions. The *Operator's Guide* gives the syntax and an example of each SYST transaction.

Transaction Scheduler

The scheduler transaction (SCHD) schedules transactions for later processing. You can schedule a transaction to be processed once or repeatedly, at a specific time of day or at a specific time interval.

The program $.PDSCHD provides transaction scheduling. The *Operator's Guide* gives the syntax and an example of the SCHD transaction.

Remote Management Functions

The Communications Facility includes a set of functions that help you manage remote Series/1s in your configuration. You can allocate and delete data sets, send and receive data sets, execute programs, dump storage to disk, and use a local terminal as if it were connected to a remote Series/1.

These functions are provided by a pair of programs that exchange transactions. The program that runs in the remote Series/1 is $.RMU (remote management utility). The program that runs in the system from which the functions are requested can be either $.HMU (host management utility) or Communications Facility/Host.

$.HMU runs on a Series/1. It can communicate with $.RMU in another Series/1 over any of the connections the Communications Facility supports—BSC multipoint, BSC point-to-point, Local Communications Controller, and HDLC. $.HMU and $.RMU are provided as part of the Communications Facility. They are described in the *Operator's Guide.*

Communications Facility/Host runs in a System/370, a 30*xx*, or a 4300. It can communicate with $.RMU *only* over a BSC multipoint line or an SDLC line (an SNA connection). Communications Facility/Host is a licensed program (5668-979). It is described in *Communications Facility/Host: General Information Manual.*

The path between the remote Series/1 and either type of host system can be used by application programs while remote management functions are going on.

Remote IPL

The IPL transaction IPLs an EDX supervisor at a remote Series/1. The connection between the Series/1 where the IPL transaction is processed and the Series/1 to be IPLed can be BSC multipoint, BSC point-to-point, or Local Communications Controller. It cannot be an HDLC line.

There are two restrictions:

- When the connection is a BSC line, the attachment at the Series/1 to be IPLed must be a BSC single-line control (feature number 2074 or 2075) or a multifunction attachment (feature number 1310).
- When the connection is a BSC multipoint line, the Series/1 to be IPLed must be the tributary.

There are several types of remote IPL. One type sends a bootstrap and then a nucleus to the Series/1 to be IPLed. The nucleus (the EDX supervisor) must contain support for the Communications Facility language extensions. Two restrictions apply to this type of remote IPL:

- The segment of the nucleus that resides in partition 1 must not exceed X'FE00' in size. Use the EDX utility, $DISKUT2, to determine the size of the nucleus.

- When the connection is a BSC point-to-point line, you must not send any messages (including broadcast transactions) to the remote Series/1 before doing the IPL, unless there is already an operating IOCP ($.IO0A10) to receive the messages.

The other types of remote IPL send just a bootstrap to the Series/1 to be IPLed. The bootstrap causes the Series/1 to be IPLed from one of these devices:

- 4950 disk at address X'03'
- 4956 IDSK disk at address X'60'
- 4962 disk at address X'03'
- 4963 disk at address X'48'
- 4964 diskette at address X'02'
- 4965 diskette at address X'45'
- 4966 diskette at address X'22'
- 4967 disk at address X'C0'
- 5170-495 disk at address X'03'
- DDSK-30 and DDSK-60 disk at address X'44'.

There are two ways of initiating a remote IPL. One is to issue an IPL transaction at a Series/1 where the program dispatcher is running. The other (called a *hardware IPL request*) is to press the load button at the Series/1 to be IPLed.

A hardware IPL request can be used only at a Series/1 with a Local Communications Controller attachment that is strapped as the IPL device. The hardware IPL request is transmitted to all other Series/1s on the Local Communications Controller ring. The Series/1 that is enabled to respond to the IPL request issues an IPL transaction (any of the types described earlier) that causes the Series/1 where the load button was pressed to be IPLed.

A Series/1 is enabled to respond to a hardware IPL request when all these conditions hold:

- The Communications Facility, the IOCP that manages the Local Communications Controller ($.IO0AB0), and the program dispatcher are all active.

- The station representing the Series/1 where the load button was pressed is active.

- The program dispatcher transaction table contains an entry for transaction identifier IPL.

- The IPL transaction data set ($.SYSLCC) contains an entry for the Series/1 that requested the IPL.

Each entry in data set $.SYSLCC contains the name of a station that represents a remote Series/1 and the IPL transaction used to IPL that Series/1. When you create $.SYSLCC data sets, be careful to enable one and only one Series/1 to IPL each Series/1 that will issue hardware IPL requests. If there is none, the hardware IPL request is ignored. If there is more than one, the results are unpredictable.

The program $.PDIPL provides remote IPL. The *Operator's Guide* gives the syntax and an example of the IPL transaction. In this book, the chapter "Communications Facility Data Sets" on page 179 explains how to create $.SYSLCC, and the chapter "Sample Configurations" on page 269 contains an example of a $.SYSLCC data set.

Remote Disk Support

Remote disk support allows programs to access disk volumes attached to remote Series/1s just as if they were attached to the local Series/1. The volumes must be performance volumes. No special programming is required to use remote disk. The EDX I/O instructions to access remote disks are the same as those used to access local disks. Two programs, CSPDIO and $.PD<IO>, provide remote disk support.

CSPDIO is a module included in the EDX supervisor for Series/1s that access remote disks. CSPDIO must be placed between CSXSYS and $EDXDEFO in the static section of the supervisor. You must define remote disk volumes and the communication path used to access them to the EDX supervisor, as explained in the chapter "Installing the Communications Facility" on page 251. CSPDIO intercepts any I/O request to a remote disk, builds a transaction, and sends it to the Series/1 where the disk is located.

$.PD<IO> is a transaction-processing program that runs in Series/1s where remote disks are located. It receives the transactions that CSPDIO issues, performs the requested I/O, and sends back a transaction that contains the result. CSPDIO passes the result to the program that issued the I/O request, just as EDX disk I/O support does.

The connection between the Series/1 that accesses remote disks and the Series/1 where the disks are located can be BSC multipoint, BSC point-to-point, or Local Communications Controller. It cannot be an HDLC line.

You can't access remote disks until the program dispatcher is running in the Series/1s where the disks are located. Neither the Communications Facility nor the program dispatcher has to be running in Series/1s that access remote disks.

CSPDIO uses two different ways of sending disk I/O transactions, depending on whether or not the program dispatcher is running. When the program dispatcher isn't running, CSPDIO sends transactions over the path you define to the EDX supervisor.

There are three restrictions:

- You can define only one path to remote disks.
- When the path is a BSC multipoint line, the Series/1 that accesses remote disks must be the tributary.

- The BSC line or the Local Communications Controller connection can be used only by CSPDIO. You can start the IOCP that manages the line, but it can't access the line.

These restrictions are removed when you start the program dispatcher and issue a SYST SP (start remote disk) transaction. You can include the SYST SP transaction in the program dispatcher's data set ($.SYSPD). The transaction causes CSPDIO to release control of the BSC line or Local Communications Controller connection and to send disk I/O transactions to the program dispatcher. Thereafter, access to remote disks is by the paths defined in the program dispatcher's path table.

Remote disk support provides no protection against concurrent update of a data set. You should use it only to retrieve data sets or to create or update data sets that are used exclusively by one program. If you have an application that requires shared data sets, you should write it so that programs send their disk I/O requests to a single program that performs the requested I/O.

Work Session Controller

The work session controller ($.WSC) provides a high-level terminal access method that you can use to write interactive application programs that communicate with EDX terminals. The work session controller supports communication with 4978, 4979, 4980, and 3101 terminals and with 4973, 4974, 4975, 5219, 5224, 5225, and 5262 printers. The terminals and printers must be defined to EDX; they need not be defined as Communications Facility stations. An application program can communicate with multiple terminals, which can be attached to any Series/1 in the network.

The work session controller provides many terminal I/O and control services. It provides several techniques for transferring control from one program to another, which makes it easy to segment an application into multiple small programs. It enables programs to retrieve screen images from disk, save data associated with a particular terminal on disk, and retrieve saved data.

An application program gets services from the work session controller by exchanging transactions with it. The program sends a WSC transaction that requests a particular service for a particular terminal. The work session controller receives the WSC transaction, performs the requested service, and sends an acknowledgment transaction. The acknowledgment transaction shows which service was performed for which terminal, and may (depending on the service) contain data or a return code. The program that issues a WSC transaction specifies where the acknowledgment is to be sent—it can be routed back to the same program or to some other transaction-processing program.

The *Programmer's Guide* shows the format of all the WSC transactions and explains how to use them.

Utility Programs

The Communications Facility includes several utility programs. Some are used to install and maintain a Communications Facility system. Others are aids for application development.

$.CFMENU, Sample Program

$.CFMENU is a sample application program that serves two purposes. You can run the program to verify that you have installed the Communications Facility correctly, as explained in the chapter "Installing the Communications Facility" on page 251. $.CFMENU communicates with operators at Communications Facility terminals and illustrates many of the programming techniques that you may use in your application programs. The *Programmer's Guide* contains a discussion and listing of the program.

$.CONFIG, Configuration Processor

You may use $.CONFIG to define stations in $.SYSNET (the network configuration data set), to modify and display existing station definitions, and to maintain $.SYSMSG (the log message data set). $.CONFIG is described in the *Operator's Guide.*

$.DSFORM, Emulated PC Disk Management Utility

You use $.DSFORM to create, delete, or rename data sets used as emulated Personal Computer (PC) disks. $.DSFORM is described in the *Operator's Guide.*

$.DSINIT, Disk-Queue Data Set Initialization

You use $.DSINIT to initialize data sets that are to be used as disk queues. $.DSINIT is described in the *Operator's Guide* and in this book, in the chapter "Creating Disk-Queue Data Sets" on page 207.

$.PANEL, Panel Design Aid

You use $.PANEL to create panels that your application programs display. A panel is a screen image for a Communications Facility terminal, either a 3277 or a 4978, 4980, 3101, or 7485 being managed as if it were a 3277. The *Programmer's Guide* describes $.PANEL.

$.PNLUT1, Panel Utility

You use $.PNLUT1 to print the descriptions of panels created by $.PANEL. The *Programmer's Guide* describes $.PNLUT1.

$.PDBSTS, Volume Protection

$.PDBSTS allows you to gain exclusive control of a disk-volume directory in a remote Series/1 before executing a program that would update that directory. You might use it, for example, before running EDX utility $DISKUT1 to allocate a data set on a remote disk volume. $.PDBSTS is described in the *Operator's Guide.*

$.SETSTG, Set $.CFD Storage Size

You use $.SETSTG to alter the size of the message buffer pool and the file control block pool for $.CFD, the disk-queuing version of the Communications Facility control program. $.SETSTG is described in the *Operator's Guide.*

You use the EDX utility $DISKUT2 to alter the size of the message buffer pool for $.CFS, the storage-queuing version of the control program.

$.UT1, Diagnostic Aid

You can use $.UT1 to diagnose Communications Facility problems and to test your application programs. You can use it to manage stations, send and receive messages, display the system storage pool and message buffer pool, and get the addresses of various system control blocks. $.UT1 is described in the *Operator's Guide.*

$.UT2, Message Queue Management

You can use $.UT2 to obtain a report of messages in a disk queue or to purge messages from a disk or storage queue. When you purge messages, you can also get a report of them, and you can send each message on to the destination you specify. You can use most $.UT2 functions with any station's message queue. A few can be used only with the undeliverable message queue, $.WASTE. $.UT2 is described in the *Operator's Guide.*

$.WSCUT1, Image Library Management

Screen images displayed by the work session controller are stored in data set $.WSCIMG. You use the EDX utility $IMAGE to create screen images. You use $.WSCUT1 to convert the images to the format required by the work session controller and to store them in $.WSCIMG. You can also use $.WSCUT1 to display stored images. $.WSCUT1 is described in the *Programmer's Guide.*

$.WSMENU, Work Session Controller Sample Program

You can use $.WSMENU to start communication between an EDX terminal and work session controller application programs. $.WSMENU is described in the *Programmer's Guide.*

User-Written Programs

You may want to supplement the Communications Facility's functions by adding your own programs to:

- Process undeliverable messages
- Process system log messages
- Intercept messages sent by IOCPs
- Provide support for devices
- Provide new commands
- Communicate with operators at terminals.

Language and Assembler Considerations

Regardless of what function a program provides, you can code it in EDL or in assembler language, using the language extension instructions to obtain Communications Facility services. The instructions are provided in two forms—as overlays for the EDL compiler ($EDXASM) and as macros for the Series/1 macro assembler ($S1ASM).

Coding Interactive Programs

There are two ways of writing programs to communicate with terminal operators:

- Exchanging 3270 data streams with an IOCP that manages Communications Facility terminals.

- Exchanging WSC transactions with the work session controller, which manages EDX terminals.

You can write almost any interactive program either way. Here are some factors that you need to consider when you decide which method to use. We'll refer to the first method as 3270DS and the second as WSC.

In support of 3270DS, the Communications Facility provides utility program $.PANEL for building screen images and language extension instructions for building 3270 data streams and extracting information from them.

It's important to consider the type of terminal you'll be communicating with. Only 3270DS allows communication with 3277 and 7485 terminals, and only WSC allows communication with 4979 terminals. Both methods support printers, 4978s, 4980s, and 3101s. WSC supports 3101s in both character and block mode; 3270DS supports them only in character mode.

The processor storage required for EDX control blocks, station blocks, and programs may be a consideration. If you're using 3101 terminals with WSC, you have to define them to EDX; if you're using them with 3270DS, you don't.

The work session controller creates station blocks for the terminals it manages. These station blocks are larger than those for terminals managed by IOCPs. (The sizes of all types of station blocks are given in the chapter "Planning Storage Requirements" on page 213.)

The size of the work session controller is approximately 12K, and the size of the program dispatcher (which is required for WSC) is approximately 11K. The various IOCPs that manage terminals range in size from 10K to 25K. (See the chapter "Planning Storage Requirements" on page 213 for exact sizes.) You can use the program dispatcher with 3270DS, but you don't have to; if you have a multinode configuration, you probably will.

If your application includes communication with a host system, and if the host system requires 3270 data streams, you may find it easier to communicate with terminals the same way.

Managing a Multinode Configuration

If your configuration consists of more than one node, you may want to designate one of the nodes as a central site from which you manage the operation of the other nodes. Although the Communications Facility is a peer-to-peer system, it does provide many facilities that you can use for centralized management. This section summarizes those facilities; most of them have already been discussed in this chapter.

You can IPL a remote node, sending either a bootstrap or a bootstrap and a nucleus. You can set up a node so that the Communications Facility is started automatically at IPL, and so that startup commands (commands to set the log device, start stations, etc.) are issued automatically. The chapter "Operating the Communications Facility" in the *Operator's Guide* explains how to set up a node this way.

The remote management utility programs ($.HMU and $.RMU) allow you to allocate and delete data sets, send and receive data sets, execute programs, dump storage to disk, and use a local terminal as if it were connected to a remote Series/1.

The program dispatcher allows you to send commands to a remote node. The operator at the central site can issue commands to start and stop stations in remote nodes.

Although the Communications Facility doesn't provide a central system log, it's not difficult to write a program that receives log messages at remote nodes and sends them to the central site. The *Programmer's Guide* explains how to code such a program.

Remote disk support allows programs running in remote nodes to access disks at the central node. The remote nodes can be diskless systems. It is recommended, but not required, that each node have a diskette drive.

You can use the remote command and transaction scheduling functions of the program dispatcher to shut down nodes (including the central node) when no operator is present. The chapter "Operating the Communications Facility" in the *Operator's Guide* explains how to do this.

Defining Stations

For each node in your Communications Facility system, you must define all the local stations (stations in that node) and remote stations (stations in other nodes) to which local stations will send messages. When you define a station, the information is recorded in $.SYSNET, the network configuration data set. There is one $.SYSNET data set for each node.

A Communications Facility configuration is dynamic; you can add new station definitions and change existing ones at any time.

To define stations, you can use either the utility program $.CONFIG or the CP DEF command, which are described in the *Operator's Guide*. This chapter describes the various types of stations and the information you specify to define them. The chapter "Sample Configurations" on page 269 shows the station definitions for several Communications Facility configurations.

Station Types

The following sections describe the various types of stations supported by the Communications Facility.

Line Station

A line station represents a Series/1 communication link. The link may be a Local Communications Controller ring, a Series/1-PC Connect Attachment, a channel attachment to a host computer, a binary synchronous communication (BSC) line, or an HDLC line. A BSC line may connect to a 3270 system, a host computer that views the Series/1 as if it were a 3270 system, another Series/1, or some system that uses point-to-point protocol.

IOCPs manage communication over lines. A line station block contains or points to its associated task control block. The TCB is used with reentrant code in the IOCP.

Circuit Station

A circuit station represents an X.25 virtual circuit (channel)—either permanent or switched. Circuit stations are used for sending and receiving messages over HDLC lines to another Series/1 or through an X.25 network.

When you define a circuit station, you specify the name of its associated line station. The line station task, which is part of an IOCP, manages communication with the circuit station.

SNA Physical Unit (PU) and Logical Unit (LU) Stations

SNA stations are used for communication with a host computer over a synchronous data link control (SDLC) line. The host computer views the Series/1 as if it were a 3274-SDLC control unit with attached display stations and printers.

An SNA LU station represents an emulated display station or printer. When you define an SNA LU station, you specify the name of an SNA PU station, which is used to hold information common to all its SNA LU stations. You can define up to four PU stations in a node. EDX Secondary SNA1 allows you to define only one PU station in a node. EDX Secondary SNA2 allows you to define up to four PU stations in a node.

An IOCP manages communication with SNA stations. An SNA LU station block contains or points to a task control block that is associated with reentrant code in the IOCP.

Node Station

A node station represents a remote node accessible from a Local Communications Controller ring, a Series/1-PC Connect Attachment, a BSC point-to-point line, or an HDLC line. The local and remote nodes may be connected to each other, or there may be intermediate nodes or an X.25 packet-switching network between them.

When you define a remote node that is connected to the same Local Communications Controller ring as the local node, you specify the name of the Local Communications Controller line station. The line station task, which is part of an IOCP, manages the node station. The node station is used to hold messages being sent to the remote node, and should be defined with subtype CF or CM.

When you define a remote node that is connected to the same Series/1-PC Connect Attachment as the local node, you specify the name of the Series/1-PC Connect Attachment line station, and the name of the associated volume station, if needed. The PCC line station task, which is part of the IOCP, manages the node station. The name of the volume station is used to route messages to the PC Connect disk-server IOCP, $.IO1CC8. The node station is used to hold messages being sent to the gateway PC and should be defined with subtype PC.

When you define a remote node that is accessible in some other way, you do not specify the name of any associated station. Instead, you must link the node station to the station that represents the path to the remote node, as explained in section "Linking Stations" on page 65. The station you link to receives the messages being sent to the remote node.

Device Station

A device station represents a device attached to a Series/1. The device can be an IBM:

- 4978 terminal
- 4980 terminal
- 3101 terminal
- 3161 or 3163 in 3101 emulation mode
- 7485 terminal
- Personal Computer emulating a 3101
- Series/1 printer.

IOCPs manage communication with device stations. A device station block contains or points to its associated task control block. The TCB is used with reentrant code in the IOCP.

Volume Station

A volume station represents an EDX volume that contains data sets representing emulated PC disks. An IOCP, $.IO1CC8, manages communication with volume stations. Each volume station has a task control block that is associated with a reentrant task in the IOCP.

User Station

A user station represents a program. It may represent a user application program or a Communications Facility program, such as an IOCP or the program dispatcher.

Not every program that uses Communications Facility functions needs a station. A station is required if any of these conditions holds:

- Messages are sent to the program.
- The program requests acknowledgment of the messages it sends.
- You want to use a CP command to stop the program.

Message Station

A message station is just a station block with a message queue; it doesn't necessarily represent anything. The station that receives undeliverable messages, $.WASTE, may be a message station. The program dispatcher creates message stations for transaction-processing programs. Your application programs can create message stations to be used however you want.

Message stations can serve the same purpose as user stations by supplying a station block and a message queue for a program. The difference between the message station and the user station is what happens when they are started. Starting a user station causes the program with the same name as the station to be loaded. Starting a message station doesn't cause a program to be loaded.

Terminal Station

A terminal station represents either a real 3270 terminal connected to a Series/1 with a BSC line, or an emulated 3270 terminal. Emulated 3270 terminals are just station blocks with message queues that are used for communication with a host computer over a BSC line or a channel.

When you define a terminal station, you specify the name of its associated line station. The line station task, which is part of an IOCP, manages communication with the terminal station.

Remote (Vector) Station

A remote station represents a station in another node. "Remote" is not a station type in the sense of other station types described in this section. A station is identified as remote by its network address, not by its type. You can specify whatever type you want when you define a remote station.

When a station is started and its network address shows that it is a remote station, a vector station block is created. A vector station block contains only the station's name, network address, link, and status. Any other information in the station definition is ignored.

Alias Station

An alias station defines an alias name for another station. You can, for example, define XYZ as an alias for station ABC. Each time you refer to XYZ, you are actually referring to station ABC.

System Station

There is only one system station. Its name is $.DISP. It is used by various Communications Facility tasks.

You don't define the system station. The $.DISP station block is created when you IPL an EDX supervisor that supports the Communications Facility.

Other Stations

There are additional types of stations supported by the *Manufacturing Automation Protocol Application Server* (program number 5719-XT1): a link-level service access point (LSAP) station and a line station that represents a link to a network interface unit (NIU).

Identifying Nodes

You identify a node by a 2-digit hexadecimal value called a *node assignment*.

If you have a single-node configuration or a multinode configuration where the only connections between nodes are BSC multipoint lines, the node assignment can be any value from X'00' to X'FF'.

If you have a multinode configuration where the connection between nodes is a Local Communications Controller ring, a BSC point-to-point line, or an HDLC line, each node must have a unique node assignment in the range X'01' to X'FF'.

The program dispatcher creates station blocks for stations that are not defined in $.SYSNET. Other programs, including those you write, may also do this. The network address of these stations is the address of the station block in S$POOL. To avoid having duplicate network addresses, do not use node assignments that fall within the range of S$POOL addresses. For example, if you have a 6000-byte S$POOL that extends from address X'057E' through X'1CEE', do not use X'05' through X'1C' as node assignments.

If your configuration includes Communications Manager or gateway PC nodes, there are special requirements for identifying nodes to both systems. The Communications Manager and the PC Connect use a 2-character alphabetic name to identify nodes instead of a hexadecimal node assignment. Specify a repeated letter (for example, AA) as the Communications Manager or the gateway PC node name. Specify the hexadecimal representation of the letter as the Communications Facility node assignment—for example, X'C1'.

Identifying Stations

A station has two identifiers—a name and a network address.

The *station name* consists of 1 to 8 alphanumeric characters. If your configuration includes Communications Manager or gateway PC nodes, stations that will send messages to or receive messages from the Communications Manager or the PC Connect must have names no longer than four characters. If your configuration includes a volume station, the volume station name must be no longer than six characters and must be the same as the EDX volume that it represents. If your configuration includes an HDLC line, the line station name must be no longer than six characters and must match the name of the line you defined to XHCS. Each station in a node must have a unique name. Stations are referred to by their names in commands, utility programs, and language extension instructions.

The *network address* consists of four hexadecimal digits. The leftmost two digits are the node assignment of the node in which the station exists. The rightmost two digits, called the *station address*, identify the station within the node. Each station in a node must have a unique station address. The required value, which varies according to the station type, is explained in the later sections of this chapter. The message dispatcher and other parts of the Communications Facility refer to stations by their network addresses.

Configuration Worksheets

This chapter includes five configuration worksheets that you can use to record the information that you will specify when you define stations.

Figure 14 on page 51 is a worksheet for user, message, alias, and volume stations.

Figure 15 on page 52 is a worksheet for device stations.

Figure 16 on page 53 is a worksheet for line, terminal, and node stations.

Figure 17 on page 54 is a worksheet for SNA stations.

Figure 18 on page 55 is a worksheet for circuit stations.

At the top of each configuration worksheet, fill in a description of the Series/1 (for your own documentation) and its node assignment.

For each station, fill in its description (for your own documentation) and its station name. The rest of the information, which varies according to station type, is explained in the following sections.

The bottom of each worksheet includes space to record the amount of S$POOL storage that will be required for station blocks. Fill in the number of stations you defined and the total storage requirement. You can use this information when you determine the amount of processor storage your Communications Facility system will require, as explained in the chapter "Planning Storage Requirements" on page 213.

If you use the disk-queuing version of the control program, you also need the disk-queue data set worksheet, which is in the chapter "Creating Disk-Queue Data Sets" on page 207. That worksheet includes information that you will specify when you define a station that has a disk queue.

The disk-queue information you supply when you define a station is the name and volume of the station's disk-queue data set and whether or not disk queuing is to be active when the station is started. If you use $.CONFIG to define stations, the default volume is the one where $.CONFIG resides. If you use CP DEF, the default volume is the one where the Communications Facility control program resides.

Specifying Station Definitions

This section explains how to fill in the station definition worksheets for each station type. It shows the headings on the worksheet and explains what to enter under each heading. Having completed the worksheets, you can use them as input when you use the $.CONFIG, CP DEF, or CP F command to define your stations.

User/Message/Alias/Volume Configuration Worksheet

System: ____________________

Node Assignment: ________

Description	Name	NA	Type	Alias[1]

Specify only for alias station

S$POOL Storage

______User/Msg Stations	× 88 = ______	bytes
______Alias Stations	× 40 = ______	bytes
______Remote Stations	× 32 = ______	bytes
______Volume Stations	× 104 = ______	bytes

Figure 14. User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: ______________________________

Node Assignment: ________

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							
			DEVICE							

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO.

S$POOL Storage

_____3101F Devices	× 160 = _______	bytes
_____4978/Printer Devices	× 264 = _______	bytes
_____3101 Devices	× 304 = _______	bytes
_____Remote Stations	× 32 = _______	bytes

Figure 15. Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: ____________________

Node Assignment: ________

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
			LINE					
			LINE					
			LINE					

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines × 112 = _______ bytes
_____Other Lines × 264 = _______ bytes
_____Terminals/Nodes × 88 = _______ bytes
_____Remote Stations × 32 = _______ bytes

Figure 16. Line/Terminal/Node Configuration Worksheet

SNA Configuration Worksheet

System: ____________________

Node Assignment: ________

PU Definition

Description	**Name**	**NA**	**Type**	**Buffer Size**	**PU #**[1]	**Host Application**	**Host Mode**
			PU			#1:	
						#2:	
						#3:	

LU Definitions

Description	**Name**	**NA**	**Type**	**Term Type**	**PU Name**	**Local Address**	**Logon ID**	**User Data**
			LU					
			LU					
			LU					
			LU					
			LU					
			LU					
			LU					
			LU					
			LU					
			LU					
			LU					
			LU					
			LU					

[1] Valid only for EDX Secondary SNA2

S$POOL Storage

1 Physical Unit	× 128 = 128 bytes
______Logical Units	× 280 = ______ bytes
______Remote Stations	× 32 = ______ bytes

Figure 17. SNA Configuration Worksheet

Circuit Configuration Worksheet

System: ______________________________

Node Assignment: ______

PVC Circuit Definition

Description	Name	NA	Type	Circuit Type	Line	Usage	LCI
			CIRC	PVC			
			CIRC	PVC			
			CIRC	PVC			
			CIRC	PVC			
			CIRC	PVC			

SVC Circuit Definition

Description	Name	NA	Type	Circuit Type	Line	Usage	Contact	Call ID	Packet Size	Window	Protocol ID[1]	Facilities	User Data[2]
			CIRC	SVC									
			CIRC	SVC									
			CIRC	SVC									
			CIRC	SVC									
			CIRC	SVC									
			CIRC	SVC									
			CIRC	SVC									
			CIRC	SVC									
			CIRC	SVC									
			CIRC	SVC									

[1] Specify only for contact type INIT or WAIT
[2] Specify only for contact type INIT

S$POOL Storage

____Circuits × 128 = ______ bytes

Figure 18. Circuit Configuration Worksheet

Defining Line Stations

NA
is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. For a Local Communications Controller or Series/1-PC Connect Attachment line station, the rightmost two digits may be any value from 01 to FE that is unique within the node. For other types of line stations, the rightmost two digits must be the device address pinned on the attachment card that controls the device. It can't be 00; for a device with hardware address 00, specify FF.

Type
is LINE.

Line Type
is one of the following:

PTPT, to define a BSC point-to-point line that connects the Series/1 to some other system.

CPU, to define a BSC point-to-point line that connects two Series/1s.

3271C, to define a BSC multipoint line that connects the Series/1 to a 3270 system (or to a Series/1 that emulates a 3270 system).

3271E, to define a BSC multipoint line that connects a Series/1 that emulates a 3270 system to a host computer.

LCC, to define a Local Communications Controller ring.

PCC, to define a Series/1-PC Connect Attachment.

CA, to define a channel attachment.

DCE, to define an HDLC line where the Series/1 acts as data circuit-terminating equipment. This type must match the XHCS definition for the line.

DTE, to define an HDLC line where the Series/1 acts as data terminating equipment. This type must match the XHCS definition for the line.

Buffer Size
is the size of the buffer required for the station, as explained in the chapter "Planning Storage Requirements" on page 213. If you use $.CONFIG to define stations, specify a decimal value. If you use CP DEF, specify a hexadecimal value.

Device Address
is required only for a Local Communications Controller or a Series/1-PC Connect Attachment line station. Specify the Local Communications Controller or PC Connect device subchannel 0 address, as two hexadecimal digits. The value must be from 00 to FC and a multiple of 4.

Packet Size
is required only for DTE and DCE lines. Specify the size of the largest user data area you intend to send on a data packet. The packet size must be one of the standard X.25 sizes and must agree with the IMAX parameter on the lines XHCS DDM. Packet size can be no larger than IMAX-4. If you are using an X.25 packet-switching network, the size must equal the size you subscribed to from the network. The line packet size is the default for all circuits on the line.

Window Size
is required only for DTE and DCE lines. The line window size is the default for all the circuit stations on the line. If you are using an X.25 packet-switching network, the window size must equal the size you subscribed to from the network.

X.25 Network Address
is valid only for DTE and DCE lines. If you are using an X.25 packet-switching network that requires the calling address on the call request packet, use $.CONFIG or the CP F ADDR command to specify the X.25 network address in the line station definition.

Defining Circuit Stations

NA
is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits may be any value from 01 to FF that is unique within the node.

Type
is CIRCUIT.

Circuit Type
is one of the following:

PVC, to define a permanent virtual circuit.

SVC, to define a switched virtual circuit. You may specify SVC only if the associated line station type is DTE.

Line Name
is the name of the associated DCE or DTE line station. You must define the line station before you can define the circuit stations.

Usage
is one of the following:

CF, if the circuit is used to route messages with their Communications Facility headers between Series/1s.

STD, if the circuit is used just for passing data.

STD+, if the circuit is used for passing both data and X.25 control information.

Packet Size
is the 1- to 4-digit decimal size of the maximum user data field required for a data packet. If the circuit is used to send messages through an X.25 network, the packet size must be the one subscribed to from the X.25 network. Specify 16, 32, 64, 128, 256, 512, or 1024.

You must use CP F PKTSIZ command to add the packet size to the circuit definition. If you do not specify this parameter, the controlling line station's packet size is used.

Window Size
is the number of packets that can be sent before an acknowledgment is expected. Specify a decimal number from 1 to 127. You must use CP F WINDOW command to add the window size to the circuit definition. If you do not specify this parameter, the controlling line station's window size is used.

The following parameter is for PVC circuits only.

LCI
is the logical channel identifier for a PVC circuit. Specify a decimal number from 1 to 4095. This logical channel must be defined to XHCS for use as a PVC. If you are using an X.25 packet-switching network, the channel must be subscribed for use as a PVC.

The following parameters are for SVC circuits only.

Contact
describes the action the IOCP is to take when the circuit station is started. Specify one of the following:

WAIT, if the circuit is to wait for a call.

INIT, if the IOCP is to send a call request to initiate the call.

USERINIT, if the station linked to the circuit provides the call request control message to initiate the call.

Call ID
is the 2-digit ID of the X.25 network address to be called (INIT) or waited on a call from (WAIT) when the station is started. The call ID and its associated X.25 network address must be defined in the $.SYSX25 data set. You may specify 0 to accept any call when the contact type is WAIT. You may also specify 0 if the address has to be obtained from the other facilities.

Protocol ID
is an optional 8-digit hexadecimal protocol identifier. If specified, it is sent with the call request packet when the contact type is INIT, or used to screen incoming calls when the contact type is WAIT. Only incoming call packets with a matching protocol ID are accepted. The first two bits of the protocol ID have special significance to X.25 packet-switching networks.

Facilities
is one or more of the following facilities.

CUG *nn*	Closed user group. *nn* is the 2-digit decimal ID of the group.
BCUG *nnnn*	Bilateral closed user group. *nnnn* is the 4-digit decimal ID of the group.
RPOA *nnnn*	This facility allows you to specify a particular RPOA transit network through which the call is to be routed internationally, when more than one RPOA transit network exists at an international gateway. *nnnn* is the 4-digit decimal data network identification code for the requested RPOA transit network.
FS	Fast select. Fast select allows from 1 to 128 bytes of user data to be carried on a call request, call accept, or call clear packet.
FSR	Fast select restricted. This allows from 1 to 128 bytes of user data to be carried on a call request or a call clear packet; call accept is not allowed.
REV	Reverse charging. The DTE accepting the call is charged for the call instead of the DTE that places the call.

These are the facilities that the Communications Facility allows you to include in your station definition. The IOCP sends them on the call request control message for contact types INIT and USERINIT. Facilities are invalid for circuits with usage of CF or STD.

You may choose only those facilities to which you have subscribed from your X.25 network. If you subscribe to facilities other than the ones allowed in the station definition, you can initiate the call (contact type USERINIT) and specify additional facilities in the call request control message.

If you use CP DEF to define your circuit station, you don't define facilities when you define the circuit station. Instead, use the CP F FAC command after you have defined the circuit station.

User Data
: is an optional 1- to 12-character user data field to be sent with the call request packet when the contact type is INIT.

If you use CP DEF to define your circuit station, you don't define the user data field when you define the circuit station. Instead, use the CP F USER command after you have defined the circuit station.

Defining SNA PU Stations

NA
: is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits may be any value from 01 to FF that is unique within the node.

Type
: is PU.

Buffer Size
is the size of the buffer required for the LU stations associated with this PU station, as explained in the chapter "Planning Storage Requirements" on page 213. If you use $.CONFIG to define stations, specify a decimal value. If you use CP DEF, specify a hexadecimal value.

PU Number
is the number of the PU defined to the EDX Secondary SNA2 for this PU station. If $NET*x* is the PU definition associated with this PU station, *x* is the PU number.

Host Application and Host Mode
are the name and mode of an SNA application that users can select or to which they can be automatically connected. (Users may also specify SNA applications that are not defined in the PU station.) The name and mode are each one to eight characters, and must be defined in the SNA host system. You can define from zero to three SNA applications in the PU station.

If you use CP DEF to define stations, you don't specify SNA applications when you define the PU station. Instead, use the CP F APPL*n* and CP F MODE*n* commands after you have defined the station.

Note that:

- You can't assign a disk queue to a PU station.
- If you use CP DEF to define stations, you must also specify the subtype of the PU station. It is always 3274ES.

Defining SNA LU Stations

NA
is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits may be any value from 01 to FF that is unique within the node.

Type
is LU.

Term Type
is the type of emulated terminal the LU represents. Specify 3277, 3278, 3279, 3286, 3287, or 3289.

PU Name
is the name of the associated PU station. You must define the PU station before you can define LU stations.

Local Address
is either a logical unit number from 1 to 32 defined in the EDX-SNA configuration or 0. If you specify 0, EDX-SNA will assign the next available logical unit number when the station is started.

Logon ID
is the identification of an SNA application. If you want the user of this station to be automatically connected to one of the SNA applications defined in the PU station, specify its number—1, 2, or 3. If you want the user to be prompted for an SNA application, specify 0. If you want the host to initiate sessions with this station, specify HOST.

User Data
is an optional 1- to 16-character user data field to be passed to the SNA application when communication with it is established. Omit this field if you supply a logon ID of 0 or HOST.

Defining Node Stations

NA
is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits must be 00.

For a PC or CM node, the first two digits must be the hexadecimal representation of the alphabetic characters that the PC Connect or the Communications Manager uses to address this PC or CM node. For example, if this node is known to the PC Connect or Communications Manager as BB, the node address must be C200.

Type
is NODE.

Node Type
is CF for a Communications Facility system; CM for a Communications Manager system; and PC for a Personal Computer system.

Ring Address
is required only for Communications Facility or Communications Manager nodes. The ring address is the 2-digit hexadecimal Local Communications Controller ring address of the remote node. The address must have odd parity and a value, without the parity bit, from 16 to 125. Specify this parameter only if the remote node is connected to the same Local Communications Controller ring as the local node.

Line Name
is the name of the associated Local Communications Controller or Series/1-PC Connect Attachment line station. Specify this parameter only if the remote node is connected to the same Local Communications Controller ring or Series/1-PC Connect Attachment as the local node. You must define the line station before you can define the node stations.

Volume name
is required only for a PC node. It is the name of the associated volume station used for disk-server request. Specify the parameter only if the remote node is connected to the same Series/1-PC Connect Attachment as the local node. You must define the volume station before you can define the associated node station.

NA
is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits must be the device address pinned on the attachment card that controls the device. It can't be 00; for a device with hardware address 00, specify FF.

Type
is DEVICE.

Device Type
is one of the following:

4978, to define a 4978 or 4980 terminal.

PRINTER, to define a 4973, 4974, 4975, 5219, 5224, 5225, or 5262 printer.

3101F, to define a 3101 terminal, a 3161 or 3163 terminal, a 7485 terminal, or a Personal Computer emulating a 3101 attached to a Series/1 through an asynchronous adapter.

Buffer Size
is the size of the buffer required for the station, as explained in the chapter "Planning Storage Requirements" on page 213. If you use $.CONFIG to define stations, specify a decimal value. If you use CP DEF, specify a hexadecimal value.

PDS
is required only for a printer device station. Specify Y or N to indicate whether or not you want a member in partitioned data set $.SYSPRT allocated for the station. The member is used to support the printer as a buffered device, as explained in the description of $.IO0674 in the chapter "Using the Supplied Input/Output Control Programs" on page 71.

Polling CUDA
is a 3270 polling address that the Communications Facility doesn't use. You may specify any 4-digit hexadecimal value except 0000. If you link this station to an emulated terminal station for 3270 pass-through operations, you might specify the same polling address for both of them, although this is not required.

Selection CUDA
is a 3270 selection address, specified only for printer and 4978 device stations, that the Communications Facility doesn't use. You may specify any 4-digit hexadecimal value. If you link this station to an emulated terminal station for 3270 pass-through operations, you might specify the same selection address for both of them, although this is not required.

HDCPY
is required only for a 3101 or 3101F device station. Specify the name of the EDX terminal to be used as the hardcopy device—the device on which the screen display is printed when an operator presses the PA6 key.

Bit Rate
is required only for a 3101F device station. Specify the bit rate of the feature programmable communications adapter (300, 600, 1200, 2400, 4800, or 9600) or the multifunction attachment (1200, 2400, 4800, or 9600). You must specify 9600 for 7485s.

Auto Ans
is required only for a 3101F device station. If you use $.CONFIG to define stations, specify YES or NO to indicate whether or not the modem attached to the asynchronous adapter. supports auto answer. If you use CP DEF, specify ON or OFF instead. You must specify NO or OFF for 7485s.

Defining Volume Stations

NA
is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits may be any value from 01 to FF that is unique within the node.

Type
is VOL.

The station name must be the same as the EDX volume name that it represents.

Defining User Stations

NA
is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits may be any value from 01 to FF that is unique within the node.

Type
is USER.

The station name must be the same name as the program it represents.

Defining Message Stations

NA
is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits may be any value from 01 to FF that is unique within the node.

Type
is MSG.

Defining Terminal Stations

NA
is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits may be any value from 01 to FF that is unique within the node.

Type
is TERM.

Term Type
is one of the following:

3271, to define a control unit associated with a 3271C or 3271E line station.

3277, to define a display station associated with a 3271C or 3271E line station.

3286, to define a printer associated with a 3271C or 3271E line station.

PORT, to define an emulated display station or printer associated with a CA line station.

Line Name
is the name of the associated 3271C, 3271E, or CA line station. You must define the line station before you can define the terminal stations.

Polling CUDA
is required only for 3271, 3277, and 3286 terminal stations. Specify the 3270 polling address, as described later in this chapter.

Select CUDA
is required only for 3271, 3277, and 3286 terminal stations. Specify the 3270 selection address, as described later in this chapter.

Port #
is required only for PORT terminal stations. Specify the port number, a decimal value from 0 to 31.

Defining Remote Stations

You can assign a remote station any type you choose. Suppose, for example, you are defining a station to represent a 4978 in a remote node. You can define it as a device station, but it's simpler to define it as a message station. The latter definition provides all the information needed to build a remote station block.

The only requirement is that the network address of the remote station must be the same as that of the corresponding real station in the remote node. The station names need not be the same.

Defining Alias Stations

NA
: is the 4-digit hexadecimal network address. The leftmost two digits are the node assignment. The rightmost two digits may be any value from 01 to FF that is unique within the node.

Type
: is ALIAS.

Alias
: is the 1- to 8-character name of the station for which this is an alias.

Note that you can't assign a disk queue to an alias station.

Linking Stations

Having defined your stations, you will need to define links for some of them. Recall that there are two kinds of links. A *direct link* defines the default destination of messages sent by a station. An *alternate link* defines a station to which the program dispatcher sends undeliverable transactions or to which the X.25 IOCP sends X.25 control messages.

You can use the $.CONFIG LINK or CP LINK command to define both kinds of links.

This section offers some general guidelines for defining direct links. You should also study the chapter "Sample Configurations" on page 269, which shows the links required for various configurations.

Some stations must have a direct link, because otherwise the messages they send are undeliverable.

A device station, an SNA LU station, and any terminal station except one that represents a 3271 control unit require a direct link. To do 3270 pass-through, link pairs of stations to each other. One station of each pair is a device or real 3270 terminal station; the other is an emulated 3270 terminal or SNA LU station.

If you write application programs that communicate with terminal users, link the device or real 3270 terminal stations to the program (that is, to its user station). If you write application programs that communicate with host systems, link the emulated 3270 terminal or SNA LU stations to the program. Interactive application programs are usually written to communicate with multiple users, obtaining the origin of each message they receive and sending the response back to that station. If your program communicates with only one station, you can also define a link from the program to that station.

When you define an emulated terminal or SNA LU station, take care that the station's subtype is consistent with the type of station with which it will be linked:

- Link 3277 emulated terminal stations or 3277 LU stations with stations that represent 4978s, 4980s, 3101s, 7485s, real 3277 displays, or programs that process 3270 data streams.

- Link 3286 emulated terminal stations or 3286 LU stations with stations that represent 4973, 4974, 4975, 5219, 5224, 5225, or 5262.

- Link 3287 LU stations with stations that represent 3287 printers or Series/1 printers managed by $.IO0674 which includes LU-1 support.

- Link 3278 and 3279 LU stations with stations that represent the corresponding real 3270 displays.

A circuit station with usage STD or STD+ requires a direct link. Link the station to the station that is to receive data from the circuit. If usage is STD+, you can also define an alternate link; the direct link will receive data messages from the circuit; the alternate link will receive X.25 control messages.

A node station requires a direct link unless it represents a remote node that is connected to the same Local Communications Controller ring or Series/1-PC Connect Attachment as the local node. If the remote node is accessed through a BSC point-to-point line, link the node station to the CPU line station (see sample configuration 5). If the remote node is accessed through an HDLC line, link the node station to a circuit station with usage CF (see sample configuration 11). If the remote node is accessed through a Local Communications Controller ring or a Series/1-PC Connect Attachment, but it is not physically connected to the local node, link the node station to the node station that represents the intermediate node (see example 11).

A PTPT line station (one that represents a point-to-point connection between a Series/1 and some other system) requires a direct link. Link the station to the program or device that is to receive data sent by the other system.

A volume station (one that represents emulated PC disks on a LAN) requires a direct link. Link the volume station to the PC node station that represents the gateway PC on the LAN which uses the emulated PC disks as LAN-shared disks.

Some stations don't ever need a direct link, because they are never the origin of messages. These stations are:

- Node station for remote nodes that are connected to the same Local Communications Controller ring or Series/1-PC Connect Attachment as a local node

- Circuit stations with usage CF

- Terminal stations that represent 3271 control units

- Line stations except PTPT

- An SNA PU station

- Remote stations.

You can define direct links for such stations, but the definition will have no effect.

Defining Modes of Operation

Some of the IOCPs support alternative modes of operation for the stations they manage, as described in the chapter "Using the Supplied Input/Output Control Programs" on page 71. When you define stations, you establish default modes of operation. Having defined a station, you can use the CP F MODE command to specify some other mode.

3270 Polling and Selection Addresses

The polling and selection addresses for a real or emulated 3270 terminal station depend on the terminal's control unit number and device number. Each control unit has a number from 0 to 31. Each device (display station or printer) is attached to a control unit and has a number from 0 to 31 that represents its position on the control unit. For real 3270 terminals, these numbers are assigned when the terminals are installed. For emulated 3270 terminals, the numbers are defined by the host program with which the Series/1 communicates. If you use CP DEF to define stations, you must specify the polling and selection addresses. If you use $.CONFIG, you can instead specify the control unit number and device number, and $.CONFIG will generate the appropriate polling and selection addresses.

Each address is a 4-digit hexadecimal value. The first two digits are referred to as CU (control unit) and the last two as DA (device address). Use Figure 19 on page 68 and the following rules to determine the polling and selection addresses for a terminal station:

- The polling address for a control unit is the value from column 2 for the corresponding control unit number followed by 7F.
- The selection address for a control unit is 0000.
- The polling address for a device is the value from column 2 for the corresponding control unit number followed by the value from column 2 for the corresponding device number.
- The selection address for a device is the value from column 3 for the corresponding control unit number, followed by the value from column 2 for the corresponding device number.

For example, the polling and selection addresses for control unit 5 and its devices 0, 1, and 2 are:

Station	**Polling Address**	**Selection Address**
Control Unit 5	C57F	0000
Device 0 on CU5	C540	E540
Device 1 on CU5	C5C1	E5C1
Device 2 on CU5	C5C2	E5C2

Remote IPL Note: The strapping for remote IPL on a BSC adapter card is also the high-order bit of the polling address. With IPL enabled, polling addresses must begin with 'C' or 'D' and selection addresses with 'E' or 'F'.

Control Unit or Device Number	**Polling CU Polling DA Selection DA**	**Selection CU**
0	40	60
1	C1	61 (E1)[1]
2	C2	E2
3	C3	E3
4	C4	E4
5	C5	E5
6	C6	E6
7	C7	E7
8	C8	E8
9	C9	E9
10	4A	6A
11	4B	6B
12	4C	6C
13	4D	6D
14	4E	6E
15	4F	6F

Figure 19 (Part 1 of 2). 3270 Polling and Selection Addresses

1 For 3270 emulation, the Series/1 communication hardware doesn't honor selection CU values 61 and F0. Use the values E1 and 70 instead in your station definitions and in host system definitions.

Control Unit or Device Number	Polling CU Polling DA Selection DA	Selection CU
16	50	F0 (70)[1]
17	D1	F1
18	D2	F2
19	D3	F3
20	D4	F4
21	D5	F5
22	D6	F6
23	D7	F7
24	D8	F8
25	D9	F9
26	5A	7A
27	5B	7B
28	5C	7C
29	5D	7D
30	5E	7E
31	5F	7F

Figure 19 (Part 2 of 2). 3270 Polling and Selection Addresses

Using the Supplied Input/Output Control Programs

This chapter explains the requirements and restrictions you need to take into account if you're going to use the IOCPs that are supplied as part of the Communications Facility. For each IOCP, it explains hardware requirements; EDX requirements; station definition requirements; and detailed information about the particular IOCP.

The IOCPs that are described in this chapter are:

- PC Connect IOCP, $.IO0AA8
- Series/1-to-Series/1 Local Communications Controller IOCP, $.IO0AB0
- X.25 IOCP, $.IO0AB8
- 3270 control IOCP, $.IO0AC0
- Series/1-to-System/370 channel attachment IOCP, $.IO0AD0
- Series/1-to-host 3270 emulation IOCP, $.IO0AE0
- Series/1-to-Series/1 BSC IOCP, $.IO0A10
- Point-to-point BSC IOCP, $.IO0A20
- 3101 terminal IOCP, $.IO06F0
- 4978/4980 data stream IOCP, $.IO0670
- Printer IOCP, $.IO0674
- 7485 terminal IOCP, $.IO0680
- PC Connect disk-server IOCP, $.IO1CC8
- Series/1-to-host SNA IOCP, $.IO14E8.

Buffer Requirements

Each IOCP has a buffer pool from which it obtains buffers to process messages for the stations it manages. The sizes of the buffer pools, as distributed, may not be suitable for your installation. The chapter "Planning Storage Requirements" on page 213 explains how to determine the buffer size required for stations and the buffer pool size required for each IOCP.

Modes of Operation

Some of the IOCPs supplied with the Communications Facility allow you to select between different modes of operation.

Figure 20 on page 72 summarizes these modes of operation. It shows the types of stations to which modes of operation apply, the IOCPs that support the modes, the default modes, and the alternatives.

Default modes of operation are established when you define stations. You can use the CP F MODE command to specify alternative modes for stations.

The following sections describe the various modes of operation. For more information about how a particular IOCP supports a mode of operation, see the description of that IOCP.

Station Type	IOCP	Default Mode	Alternative Modes	CP F MODE Parameters
Terminal	$.IO0AC0	Basic Non-display	Record Text	BAS, REC NON-D, TEX
Terminal	$.IO0AE0	Basic	Record	BAS, REC
Terminal	$.IO0AD0	Basic	Record	BAS, REC
SNA LU	$.IO14E8	Basic Unbind stop NETINIT wait No Personal Computer	Record Unbind retry No NETINIT wait Personal Computer	BAS, REC UBSTOP, UBRETRY WAIT, NOWAIT NOPC, PC
Circuit (INIT)	$.IO0AB8	Stop	Retry	STOP, RETRY
4978 device	$.IO0670	Upper/lower	Lower/upper Upper/upper	UL, LU, UU
Printer device	$.IO0674	Basic Non-display 3270 data stream Upper case	— Text SNA character string Mixed case	— NON-D, TEX 3270, SCS UC, MC
3101 or 7485 device	$.IO06F0 $.IO0680	Basic Non-display — Auto-answer off	— Text — Auto-answer on	— NON-D, TEX BIT *rate* AUT ON, AUT OFF
PTPT line	$.IO0A20	Basic Secondary Nontransparent write	Record Primary Transparent write	BAS, REC SEC, PRI XOFF, XON
User	$.IO0670 $.IO0674 $.IO0680 $.IO06F0 $.IO0AC0 $.IO0AD0 $.IO0AE0	Non-remove	Remove	NON-R, REM
User	$.IO14E8	Remove	Non-remove[3]	REM, NON-R

Figure 20. IOCP Modes of Operation

---

2 Terminal operator can use keyboard to select mode.

3 Must be specified after IOCP is started.

Basic and Record Modes

The terms basic mode and record mode are used to describe data as well as modes of operation. *Basic mode* data is data that contains device-dependent information, such as BSC and 3270 control characters. *Record mode* data does not contain device-dependent information.

When an IOCP receives record mode data to be transmitted out of the Series/1 (to a terminal, a device, or a communication line), it appends the required device-dependent information. It does so whether or not the station is in record mode.

Data transmitted into the Series/1 is always basic mode data. When a terminal station (real or emulated) or an SNA LU station is in record mode, the IOCP removes the device-dependent information before sending the data on to the station's direct link. When a PTPT line station is in record mode, the IOCP removes the first character (STX) and the last character (ETB or ETX) from the data before sending it to its destination. When a PTPT line station is in record mode and the data is transparent, the IOCP sends the data without modification to its destination. The IOCP also issues a warning log message to report that it received transparent data for a station in record mode. When a station is in basic mode, the IOCP sends the data without modification. Note that record mode is not supported for device stations.

Two 3270 terminals can communicate directly with each other (not through an application program) only if they are in record mode. They can't communicate directly if they're in basic mode because the device-dependent information in input data is not the same as that required for output. A 3270 terminal in record mode can also send messages to a device station, but the reverse is not true.

Basic and record modes of operation don't apply to the Series/1-to-Series/1 IOCPs, ($.IO0A10, $.IO0AB0, and $.IO0AB8), and they don't apply to the PC Connect IOCP, $.IO0AA8. These IOCPs don't modify the content of messages they transmit between Series/1s or between Series/1 and Personal Computers. $.IO0A10 , $.IO0AA8, and $.IO0AB0 do append the required line protocol characters to output messages and remove them from input messages.

Non-Display and Text Modes

Non-display and text modes apply to 3270 terminals, 3101 and 7485 devices, and printers. For these stations, non-display means uppercase only, and text means uppercase and lowercase. When such a station is in non-display mode, the IOCP translates lowercase characters to uppercase in input from a 3270 terminal or a 7485 device, and in output to a printer or a 3101 or 7485 hardcopy device. When the station is in text mode, the IOCP doesn't translate data.

UBSTOP and UBRETRY Modes

An SNA logical unit (LU) station is in either UBSTOP or UBRETRY mode. These modes affect what happens when the host sends an unbind request to terminate a session that was established automatically. If the station is in *UBSTOP* mode, the IOCP stops the station. If the station is in *UBRETRY* mode, the IOCP tries to rebind the session until it is successfully reconnected. You must stop the LU station to end the session.

STOP and RETRY Modes

An SVC circuit station that was defined with a contact type of INIT is either in STOP or RETRY mode. These modes affect what happens when the virtual call that the station initiated is cleared. If the station is in STOP mode, the IOCP stops the station. If the station is in RETRY mode, the IOCP tries to reinitiate the call. If it is in RETRY mode, you must stop the circuit station to end the session.

WAIT and NOWAIT Modes

An SNA logical unit (LU) station is either in WAIT mode or NOWAIT mode. These modes affect what happens when the IOCP issues a NETINIT and the LU-to-SSCP session is not available. If you specify WAIT mode, EDX-SNA support waits until the session is established. If you choose NOWAIT mode, EDX-SNA support completes the NETINIT with a negative return code.

Personal Computer or No Personal Computer Modes

An SNA logical unit (LU) station is either in PC or NOPC mode. When the station is in PC mode, the SNA IOCP assumes that messages sent to and from the LU station are through a Series/1-PC Connect Attachment. When the station is in NOPC mode, the IOCP assumes that the LU station is not connected through a Series/1-PC Connect Attachment. Specify PC mode when the LU station sends or receives messages through a Series/1-PC Connect Attachment.

Note: You must specify PC or NOPC mode for the SNA IOCP to function correctly.

Upper/Lower, Lower/Upper, and Upper/Upper Modes

For a 4978 device, these modes of operation define the relationship between the keyboard shift and the lowercase and uppercase alphabets. Only the 26 letters of the alphabet are affected by these modes.

When a station is in *upper/lower mode*, downshift (shift key not depressed) produces uppercase letters, and upshift (shift key depressed) produces lowercase letters. When a station is in *lower/upper mode*, downshift produces lowercase letters, and upshift produces uppercase letters. When a station is in *upper/upper mode*, either downshift or upshift produces uppercase letters.

Alphanumeric and Numeric Modes

When a station is in *numeric mode*, the IOCP validates data entered into fields that have the numeric attribute. The only valid characters are digits, periods, and minus signs. When a station is in *alphanumeric mode*, the operator can enter any data in any field, whatever its attribute.

The alphanumeric/numeric support is provided as follows:

- For a 3270 terminal, numeric mode is supported by optional hardware, the Numeric Lock special feature. The operator may or may not be able to disable numeric field validation, depending on the type of keyboard.
- For a 3101 or a 7485 device, the IOCPs support numeric mode; it is always in effect.
- For a 4978 or 4980 device, the control store supports numeric mode. The operator can disable numeric field validation.

3270 and SCS Modes

A printer station is in either 3270 data stream or SNA character string (SCS) mode. If the printer station is in 3270 mode, data received by the IOCP for transmission to the printer is interpreted as 3270 data stream. If the printer station is in SCS mode, data received by the IOCP for transmission to the printer is interpreted as an SCS data stream.

Upper Case and Mixed Case Modes

These modes determine whether printer devices print in all uppercase or mixed case.

Primary and Secondary Modes

A PTPT line station managed by the point-to-point BSC IOCP is in either primary or secondary mode. If both a *primary* and a *secondary* station try to seize control of the line at the same time, the primary station goes first. The secondary station can transmit messages only when the primary station has left the line idle.

A PTPT line station normally operates as the secondary station, because the 2770-like devices with which it exchanges data normally operate as primary stations. If you use the point-to-point BSC IOCP for communication between Series/1s, set one of the PTPT line stations to primary mode and the other to secondary mode.

XON and XOFF Modes

A PTPT line station managed by the point-to-point BSC IOCP is in either XON or XOFF mode. If it is in *XON mode*, the IOCP writes its data using transparent write. If it is in *XOFF mode*, the IOCP writes its data using nontransparent write.

Bit Mode

This mode determines the bit rate for 3101s and 7485s defined as 3101F stations.

Auto-Answer On and Auto-Answer Off Modes

These modes determine whether the IOCP controlling a 3101, defined as a 3101F station, needs to support auto-answer.

Remove and Non-Remove Modes

Remove and non-remove modes apply to the user stations for IOCPs that manage devices, terminals (real or emulated), and SNA LUs. A real 3270 system operates in certain ways because its devices are buffered. When an IOCP is in *remove mode*, it simulates these buffered operations; when it's in *non-remove mode*, it doesn't.

When a 3270 terminal operator presses a program attention key (for example, ENTER), data is transferred to the device buffer, where it remains until the host program solicits input. If response time is slow, an impatient operator may press RESET and then ENTER again. This affects only the device buffer. When the host program solicits input, it receives the last message the operator entered.

In a 3270 pass-through operation, where a device or 3270 terminal station is linked with an SNA LU or emulated terminal station, pressing RESET and then ENTER has a different effect. Each time the operator presses RESET and then ENTER, another message is sent to the emulated terminal station. When the host program solicits input, it may receive duplicate messages.

When an IOCP that manages devices or 3270 terminals receives an input message, it sends the message on to the station's direct link. If the IOCP is in remove mode and if the direct link station is an SNA LU or emulated terminal station, the IOCP purges the direct link station's message queue before sending it the latest message. If the IOCP is in non-remove mode or the direct link station is of some other type, the IOCP just sends the message.

Device buffers also affect the results of a host program sending output to a 3270 terminal. The output data is transferred to the device buffer, and may overwrite pending input data.

When an IOCP that manages SNA LUs or emulated terminals receives data from the host, it sends the data on to the station's direct link. If the IOCP is in remove mode, it first purges pending input data; that is, it purges the message queue of the SNA LU or emulated terminal station. If the IOCP is in non-remove mode, it doesn't purge pending input.

PC Connect IOCP, $.IO0AA8

The PC Connect IOCP manages communication between a Series/1 and a Personal Computer (PC) connected by a Series/1-PC Connect Attachment. The IOCP allows all the Personal Computers in an IBM PC local area network (LAN) to communicate with the Series/1 through a PC, called the gateway PC. The gateway PC is directly linked to the Series/1 and performs all the message handling and program-to-program communications between the PCs on the LAN (user PCs) and the Series/1. The IOCP can manage multiple attachments to various LANs.

You can use the IOCP in these ways:

- To allow a PC user to pass through to one or more hosts as a 3278. See example 15 in the chapter "Sample Configurations" on page 269.
- To allow a PC user to transfer files to and from any host over SDLC
- To use a Series/1 data set as a LAN-shared PC disk
- To allow a PC user to use a Series/1 printer as a LAN-shared printer
- To extend the current message function of the IBM PC network and to allow your computer to communicate with other computers on another LAN (inter-LAN messaging). See example 16 in the chapter "Sample Configurations" on page 269.
- To allow communication between PC programs on separate IBM PC networks
- To allow communication between Series/1 programs and PC programs
- To allow a PC program to issue user error log messages to an EDX error log file on the Series/1.

Hardware Requirements

Use of this IOCP requires the Series/1-PC Connect Attachment (feature number 4000) and the cable (feature number 4001).

EDX Requirements

Include Communications Facility PC Connect support modules in the EDX supervisor. The required modules are:

- CSPCCINT, the PC Connect initialization routine
- CSPCCIPL, the PC Connect initialization call routine
- CSPCCOPN, the PC Connect microcode data set(s) open routine
- CSPCCPTC, the PC Connect microcode patch load routine.

Include EDX Local Communications Controller support (LCCAM) in the EDX supervisor. EDX unmapped storage support must be included.

Personal Computer Requirements

Use of this IOCP requires Series/1-PC Connect (5719-CN1). For information on hardware and software requirements for the PC and Series/1-PC Connect, see the IBM *Series/1-PC Connect for the Event Driven Executive Communications Facility.*

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0AA8, type=USER.
- Each Series/1-PC Connect Attachment. Type=LINE, subtype=PCC.
- Each gateway PC as a remote node station. Type=NODE, subtype=PC.
- Each PC on the LAN as a remote station. This definition is required only if host communication is requested.
- Each Series/1 printer used by the LAN as a printer running in SCS mode. Type=DEVICE, subtype=printer.
- Each volume to be used by the LAN to hold emulated PC disks. Type=VOLUME.

A link between the printer station and the node station is required. A link between the remote station representing the PC and the host emulated terminal is required to pass through and transfer files to and from the host. A link between the volume station and the node station is required.

The gateway PC appears to the Series/1 as a PC node connected through a Series/1-PC Connect Attachment. The gateway PC uses a 2-character alphabetic name to identify nodes instead of a hexadecimal node assignment. Specify a repeated letter (for example, AA) as the LAN name on the gateway PC. Specify a hexadecimal representation of the letter as the node assignment for the PC node (for example, X'C1').

IOCP Operation Overview

The PC Connect IOCP consists of a main task and three subtasks for each started line station—a device task, a read task, and a request task. The main task processes commands to stop or halt the IOCP and commands to start, stop, halt or vary off-line stations. The device task controls the other two subtasks and transmits all outgoing messages and control messages on subchannel 2 of the device. The read task receives all incoming data messages on subchannel 1 and routes them through the message dispatcher to the appropriate station. The request task receives all incoming control messages on subchannel 0.

Once the line station is started, you can start the node station and remote stations representing the PCs on the network. Message transmission cannot occur until the IOCP has established communication between the Series/1 and the gateway PC.

Establishing Communication

To establish communication between the Series/1 and the gateway PC, start the PCC line station and the node station that represents the gateway PC. When the PC Connect in the gateway PC is operational, the two systems exchange a sequence of control messages (on-line broadcast and on-line response) to establish communication.

Once communication has been established, message transmission between the Series/1 and PCs can begin.

Sending Messages

When a message is placed on a PC node station's queue, the IOCP retrieves it and transmits it over the Series/1-PC Connect Attachment. A 20-byte Personal Computer message header precedes each outgoing message. If the flag bit in the Communications Facility message header indicates that the message includes a Personal Computer message header, the Communications Facility header is removed and the message is transmitted. Otherwise, the IOCP converts a Communications Facility message header to a Personal Computer message header before transmitting the message. For more information on the Personal Computer message header, see the *Series/1-PC Connect for the Event Driven Executive Communications Facility.*

Receiving Messages

When the IOCP receives a message from the PC Connect, it identifies the function request by a byte in the Personal Computer message header. The function identifiers are:

- Print-server request
- Host communication request
- Inter-LAN message request
- Program-to-program communication request
- Disk-server request
- Error log request
- User defined function request.

If the IOCP has a direct link vector, it sends the message to the direct link vector station, with the Communications Facility message header in the first 20 bytes of the message data area. For inter-LAN messaging or PC program-to-PC program communication, there will be a Communications Facility and a Personal Computer message header in the first 40 bytes of the message data area.

Print-server Request: The IOCP converts the Personal Computer header to a Communications Facility header and sends the message to the printer. The printer is represented by a printer station running in SNA character string (SCS) mode with the same name as the PC printer that is being emulated (such as LPT1, LPT2, LPT3, COM1, COM2, PRN, and AUX). The printer stations should be linked to the node station for the LAN which they are serving.

Host Communication Request: The IOCP converts the Personal Computer header to a Communications Facility header using the origin station's direct link as the destination station.

The IOCP handles segmented messages also. When the IOCP receives the first message in a chain of segmented messages, it obtains a 2K mapped or unmapped storage block. It stores all the messages from the same origin station in the 2K block until the last message in a chain of messages is received. After storing the last message in the chain or the only message in a chain, the IOCP sends the message to the origin station's direct link vector.

Inter-LAN Message Request: In this case the IOCP does not convert the Personal Computer header to a Communications Facility header. It prefixes a Communications Facility message header to the Personal Computer message header. The Communications Facility header specifies the destination station of the remote node station representing other PCs on another LAN. The Communications Facility header has a flag bit turned on to indicate a PC-to-PC message.

Program-to-program Communication Request: The IOCP tries to locate, by name, the destination station in the local Series/1. If the IOCP finds a station with the same name as the destination station specified, the IOCP assumes that the destination is a Series/1 program. It converts the Personal Computer header to a Communications Facility header and sends the message to the dispatcher for routing. On the other hand, if the IOCP does not find a station with the same name as specified by the destination station name, the IOCP assumes that the destination is a PC program. In that case it processes the message by prefixing the Communications Facility header as it does for inter-LAN messaging.

Disk-server Request: The IOCP converts the Personal Computer header to a Communications Facility header and sends the message to the volume station defined at station definition time. The volume station should be linked to the node station for the LAN which it is serving. It is important to note that for each node station representing a gateway PC, there is only one volume station associated with it.

Error Log Request: If the destination station is LOG0, the IOCP writes the data to the log data set in EDX as a user error with the permanent error indicator on.

User Defined Function Request: The IOCP processes this request in the same way as it handles a request for program-to-program communication. See "Program-to-program Communication Request" for more information.

Control Messages

This section describes the control messages that the IOCP sends or receives. These messages are all received on subchannel 0 of the PC Connect.

On-line Broadcast

The IOCP uses the PC Connect broadcast command to send an on-line broadcast message to the gateway PC when it receives a command to start a node station.

On-line Response

The IOCP sends an on-line response when it receives an on-line broadcast message, or when it receives an on-line response message and the Series/1 and the PC are not in communication. When the IOCP receives the on-line response and recognizes the PC node station that sent the message, communication is established.

The buffer size of the line station is included in this message. If the Series/1 and the PC have different line buffer sizes, the smaller size determines the amount of data they can send to each other.

Off-line Notification

This message is used to stop communications. The IOCP sends an off-line notification message to each gateway PC with which it has established communication, when the line station on the Series/1 is stopped or halted.

Restrictions

The Communications Facility does not have an end-to-end protocol between Series/1 and PC nodes, therefore there is no message integrity. If you require message integrity, you must implement it within the application programs in the Series/1 or the PC.

Do not cancel $.IO0AA8 using the EDX $C command. If you do, the mapped and the unmapped storage obtained by the IOCP is not returned to the system. Use CP H $.IO0AA8 to terminate the IOCP user station and its device stations.

Series/1-to-Series/1 Local Communications Controller IOCP, $.IO0AB0

The Series/1-to-Series/1 Local Communications Controller IOCP manages communication between Series/1s attached to a Local Communications Controller ring. The IOCP can manage multiple attachments to different rings.

You can use the IOCP in these ways:

- To send messages from stations in one Series/1 to stations in another Series/1 where $.IO0AB0 is running. See example 5 in the chapter "Sample Configurations" on page 269. The example explains how a message is routed from its origin in one node to its destination in another node.
- To send messages from stations in one Series/1 to stations in another Series/1 where the Communications Manager Local Communications Controller message path program is running. In such a configuration, you can also receive messages and commands from the Communications Manager system.
- To send messages from stations in one Series/1 to stations in another Series/1 that is not attached to the ring, but that is connected to a Series/1 on the ring. See example 11 in the chapter "Sample Configurations" on page 269.
- To IPL a remote Series/1 that is attached to the ring and strapped for remote IPL.
- To enable a remote Series/1 that is attached to the ring to access disks attached to the local Series/1. The remote Series/1 must have remote disk support in the EDX supervisor, but the Communications Facility need not be running.

Hardware Requirements

Use of this IOCP requires the Local Communications Controller (feature number 1400).

To allow a node to be IPLed by another node on the ring, install the ring remote IPL jumper. If the IPL is to be initiated manually (through the LOAD button on the console), also install the primary IPL source or alternate IPL source jumper.

EDX Requirements

Include the EDX Local Communications Controller support modules in the EDX supervisor. See the EDX *Installation and System Generation Guide* for all the information needed.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0AB0, type=USER.
- Each Local Communications Controller attachment. Type=LINE, subtype=LCC.
- Each remote node to which local stations will send messages. Type=NODE, subtype=CF (Communications Facility system) or CM (Communications Manager system).
- Each remote station to which local stations will send messages.

No links are required for the line or node stations. Link the other remote stations as required for the communication you want. Be sure that each node has a unique node assignment and that the network address of each remote station is the same as the network address of the corresponding real station in the remote node.

IOCP Operation Overview

The Local Communications Controller IOCP consists of a main task and four subtasks for each started line station—a device task, a read task, a request task, and an attention task. The main task processes commands to stop or halt the IOCP and commands to start, stop, halt, or vary off line stations. The device task controls the other three subtasks and transmits all outgoing data messages and control messages on subchannel 2 of the device. The read task receives all incoming data messages on subchannel 1 and routes them through the message dispatcher to the appropriate station. The request task receives all incoming control messages on subchannel 0. The attention task handles all attention interrupts that the Local Communications Controller device generates.

Once the line station is started, you can start node stations representing other Series/1s on the Local Communications Controller ring. You can then IPL a remote node. Message transmission between stations can't occur until the IOCP has established communication with the remote node.

Establishing Communication

To establish communication between two nodes, start the line station and the node station that represents the remote Series/1 in each node. The two nodes exchange a sequence of control messages (on-line broadcast and on-line response) to establish communication. The IOCP tries to establish communications whenever a node station is started or whenever a stopped line station is restarted.

Once communication has been established, message transmission between stations on the two nodes can begin.

Sending Messages

When a message is placed on a node station's queue, the IOCP retrieves it and transmits it over the Local Communications Controller. A 20-byte message header precedes each outgoing message. If the node station's subtype is CF, the header is the Communications Facility message header, minus the first 4 bytes (words M$NIQ and M$PIQ). If the node station's subtype is CM, the IOCP converts the Communications Facility message header to a Communications Manager message header. See the *Debugging Guide* for a description of the Communications Facility message header, and the *Communications Manager Assembler-Language Programmer's Guide* for a description of the Communications Manager message header.

The IOCP blocks outgoing messages when it can. It transmits one or several messages when the buffer is full or when there are no more messages on the node station's queue.

Receiving Messages

When the IOCP receives a message from the Local Communications Controller, it uses the ring address to identify the sending node. If the subtype of the node station is CF and the IOCP has no direct link vector, the IOCP sends the message, with its header, to the message dispatcher. The message dispatcher dispatches the message according to the destination information in the message header. If the IOCP has a direct link vector, it sends the message to that station, with the message header in the first 20 bytes of the message data area.

If the subtype of the sending node station is CM, the IOCP converts the Communications Manager message header to a Communications Facility message header. If the message is a data message, the IOCP then handles it as if it were a message from a Communications Facility node. If the message is a command, the IOCP sends it to the command processor, which calls on module $.CPCMCF to process it. See the *Communications Manager Operator's Guide* for a description of the commands that a Communications Manager node can send to a Communications Facility node.

Control Messages

This section describes the control messages that the IOCP sends and/or receives. These messages are all received on subchannel 0 of the Local Communications Controller.

On-Line Broadcast

The IOCP uses the Local Communications Controller broadcast command to send an on-line broadcast message when it receives a command to start a node station or a command to start or restart a line station.

On-Line Response

The IOCP sends an on-line response message when it receives an on-line broadcast message, or when it receives an on-line response message and the two nodes are not in communication. When the IOCP receives the on-line response and recognizes the node station that represents the Series/1 that sent the message, communication is established.

The buffer size of the line station is included in this message. If the two nodes have different line buffer sizes, the smaller size governs the amount of data they can send to each other.

Off-Line Notification

This message is used to stop communications. The IOCP sends an off-line notification message to each node with which it has established communication when the line station is stopped or halted.

Traffic Control

The IOCP receives this message when a Communications Manager operator issues a traffic control command. The IOCP issues a CP command to stop or start the specified node station.

IPL Request

The IOCP receives this message when the Communications Manager issues a conditional IPL command. The IOCP issues a WRITE IPL REQUEST command enabling the Communications Manager to perform the remote IPL.

IPL Response

An IPL bootstrap sends this message; the IOCP uses it to control the remote IPL process.

Remote Disk On-Line

Remote disk support sends this message during EDX initialization. It notifies the IOCP that remote disk support is available.

Null Message

The IOCP sends a null message when it detects that the attachment has lost an acknowledgment. The message resets hardware sequence counts.

Positive Response

The IOCP sends a positive response whenever it successfully receives a message block greater than 1K bytes on subchannel 1.

Negative Response

The IOCP sends a negative response whenever it receives an incomplete message block on subchannel 1.

Restrictions

Between two Series/1s, you can have only one active Local Communications Controller connection, BSC point-to-point connection, or X.25 connection through circuit stations with a usage type of CF. You can define and start more than one such connection, but only the one you start first is used.

When you're communicating with a Communications Manager node, there are restrictions on node assignment and station names, as described in the chapter "Defining Stations" on page 45.

The Communications Manager supports segmented messages, but the Communications Facility doesn't. The Communications Manager accumulates segments of a message and sends the entire message as one data block to the Communications Facility. The buffers you define for the Communications Facility line station and the Communications Manager message path program must be large enough to handle the largest block that will be transmitted.

The Communications Facility has no end-to-end protocol between nodes. If you require message integrity, you must implement it within the application programs in communicating nodes.

X.25 IOCP, $.IO0AB8

The X.25 IOCP allows communication between a Series/1 and X.25 data terminal equipment (DTE) through a high-level data link control (HDLC) line, with or without an intervening X.25 packet-switching network. The DTEs may be other Series/1s, other computers, or devices. The X.25 IOCP communicates over the HDLC line through the EDX X.25/HDLC Communications Support XHCS licensed program (Program Number 5719-HD2), using LAPB procedures and X.25 packet level protocol. The IOCP can manage multiple HDLC lines. There may be multiple circuits (logical channels) on each line.

You can use the IOCP in these ways:

- To route messages over an HDLC line between stations in one Series/1 and stations in another Series/1 where the X.25 IOCP is running. See "Example 12: X.25 Connection between Series/1s" on page 382. The example explains how a message is routed from its origin in the local Series/1 to its destination in the remote Series/1.

- To route messages through an X.25 packet-switching network between stations in one Series/1 and stations in another Series/1 where the X.25 IOCP is running. See "Example 13: X.25 Connection Between Series/1s through an X.25 Network" on page 392.

- For communication between Series/1 application programs and computers or devices that function as X.25 DTEs. See "Example 14: X.25 Connection between a Series/1 and DTEs" on page 406.

Refer to the CCITT Recommendation X.25 if you need more detailed information on the procedures and terms used to describe the X.25 IOCP. Terms not defined by Recommendation X.25 that are used in this section are:

X.25 control packet
: An X.25-defined packet other than a data packet; for example, a call request packet or a reset indication packet.

X.25 control message
: A fixed format message used within a Communications Facility node that contains information corresponding to an X.25 control packet.

X.25 control message link
: A station that receives X.25 control messages. It may be a circuit station's direct or alternate link, as explained in "Station Definitions" on page 91.

X.25 header
: A header prefixed to messages sent and received by the X.25 IOCP (both data and control) that identifies the message type. It is present only within a Communications Facility node.

The chapter "Writing an X.25 Application Program" in the *Programmer's Guide* gives the format of the X.25 header and X.25 control messages and guidelines for writing a program that communicates with X.25 DTEs.

Hardware Requirements

The X.25 IOCP has no hardware requirements other than those of the Communications Facility and XHCS.

EDX Requirements

XHCS is a prerequisite for the X.25 IOCP. The requirements it imposes on EDX are described in the *XHCS Programming and Operating Reference.*

XHCS Definitions

You must define the XHCS protocol descriptor and vector table (PDVT) and the device descriptor modules (DDMs) for the Communication Facility's use. You define the PDVT with XHCPROT statements. You define DDMs with the $XHCUT1 utility. The XHCPROT statement and the $XHCUT1 utility are described in the *XHCS Programming and Operating Reference.*

PDVT Definitions

The PDVT describes the link and packet protocol characteristics. You define these characteristics to XHCS by editing the XHCS data set ($XHCPROT), assembling it, and link-editing the object module with the XHCS object modules to create $XHCS. The protocol characteristics cannot be changed dynamically.

You must specify the following parameters in the XHCPROT statements for lines used by the Communications Facility:

- The first positional parameter must be PACKET.
- LNKPROT must be LAPB.
- PKTPROT and NODTYP depend on whether or not the X.25 connection has an intervening X.25 packet-switching network.
  - If you are connecting two Series/1s without an intervening X.25 packet-switching network, one of the Series/1s must have PKTPROT of DCE and NODTYP of X25DCE and the other Series/1 must have PKTPROT of DTE and NODTYP of X25DTE.
  - If you are connecting a Series/1 to an X.25 DTE without an intervening X.25 packet-switching network, PKTPROT must be DCE and NODTYP must be X25DCE.
  - If you are connecting a Series/1 to an X.25 packet-switching network, PKTPROT must be DTE and NODTYP must be X25DTE.

DDM Definitions

You must use the XHCS utility $XHCUT1 to create a DDM for each line. The name you give the line will be the name of your Communications Facility line station.

For a line used by the Communications Facility, you must define:

- LEVEL as PACKET
- TIMEOUT as 0
- RDGRPS as 1.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0AB8, type=USER.

- Each HDLC line. Name=ddmname, as defined to XHCS in the device descriptor module (DDM). Type=LINE; subtype=DCE or DTE.

- Each logical channel to be used.
  Type=CIRCUIT; subtype=PVC or SVC. Usage is one of the following:

  - CF if you are using the circuit to route messages between Series/1s.

  - STD if you are using the circuit to exchange data messages with an X.25 DTE.

  - STD+ if you are using the circuit to exchange data and X.25 control messages with an X.25 DTE.

If you are using circuit stations with usage CF to route messages between Series/1s, define stations to represent:

- Each remote node to which local stations will send messages. Type=NODE; subtype=CF.

- Each remote station to which local stations will send messages.

Link the node station to the circuit station. Link the other remote stations as required for the communication you want. Be sure that each node has a unique node assignment and that the network address of the remote station is the same as the network address of the corresponding real station in the remote node.

If you are using circuit stations with usage STD or STD+ for communication between application programs and X.25 DTEs, define stations to represent the application programs. Link the stations as follows, according to the circuit station's usage type:

- STD: Link the circuit station to the station that is to receive data messages through this logical channel.

- STD+: If you have one station that is to receive both data and control messages, link the circuit station to that station, using the direct link vector. If you have one station that is to receive only data messages and another station that is to receive only control messages, link the circuit station to the first station (data only) using the direct link vector; link the circuit station to the second station (control only) using the alternate link vector.

IOCP Operation Overview

The X.25 IOCP consists of one main task and three subtasks for each started line station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt the line stations. The three subtasks are:

- The open subtask, which processes commands to start and stop circuit stations; opens and closes the circuits; and establishes and terminates virtual calls.
- The read task, which processes messages that are read from the line and sends them to the appropriate destination.
- The write task, which processes messages sent to the circuit station associated with the line.

XHCS handles all transmission over the HDLC line. It manages X.25 protocol, pacing, and error recovery.

Establishing Communication

To establish communication through a logical channel, start the line station and the circuit station that represents the logical channel. If the circuit station has usage CF, set the node assignment and start the node station that represents the remote node to be accessed through the logical channel. If the circuit station has usage STD or STD+, start the circuit station's direct link. If the circuit station has usage STD+, start the circuit station's alternate link, if it has one.

When the X.25 IOCP or a DTE or DCE line station is started, the main task of the IOCP loads $XHCS, if it is not already loaded. When a DTE or DCE line station is started, $XHCS loads the associated device descriptor module (DDM) into its partition. If you let the X.25 IOCP load $XHCS, it may be loaded into a partition that has insufficient space for all the DDMs to be loaded. You can avoid this problem by loading $XHCS yourself into the partition you want before you start the X.25 IOCP or any DTE or DCE line station. The partition into which you load $XHCS must be static.

Permanent Virtual Circuits

When you define a permanent virtual circuit (a circuit station with subtype PVC), you specify the ID of the logical channel through which messages will be transmitted. If you are connecting two Series/1s without an intervening X.25 packet-switching network, you must specify the same logical channel ID in both Series/1s. If you are connecting two Series/1s through an intervening X.25 packet-switching network, you must specify a logical channel ID that the X.25 packet-switching network will connect to the Series/1 you want to communicate with.

When a circuit station with subtype PVC is started, the IOCP opens the logical channel specified in the station definition. If the other end of the logical channel is also open, data can flow through the logical channel. If the other end of the logical channel is not open and an attempt is made to transmit data, the IOCP receives a reset indication packet. If the circuit station has usage STD+, the IOCP sends a reset indication control message to the circuit station's X.25 control message link.

Switched Virtual Circuits

When a switched virtual circuit (a circuit station with subtype SVC) is started, a virtual call must be established before data can flow. A virtual call is a logical connection through the X.25 packet-switching network. Unlike permanent virtual circuits, circuit stations with subtype SVC do not have logical channels specified in the station definition. During virtual call establishment, a logical channel is assigned to the circuit station by $XHCS. The connection between logical channels is done by the X.25 packet-switching network.

Virtual call establishment depends on the contact parameter defined in the circuit station:

- WAIT: When the circuit station is started, the IOCP opens the circuit and waits for an incoming call packet from the network. When an incoming call packet arrives, $XHCS determines if the calling address and protocol ID match those defined for the circuit station. If they do, it passes the incoming call packet to the IOCP.

  If the circuit station has usage CF or STD, the IOCP transmits a call accept packet to the network, and the virtual call is established.

  If the circuit station has usage STD+, the IOCP sends an incoming call control message to the circuit station's X.25 control message link and waits for a response to be sent to the circuit station. If the response is a call accept control message, the IOCP transmits a call accept packet to the network, and the virtual call is established. If the response is a clear request control message, the IOCP transmits a clear request packet to the network. When a clear confirmation packet arrives from the network, the IOCP closes the circuit and sends a clear confirmation control message to the circuit station's X.25 control message link.

- INIT: When the circuit station is started, the IOCP uses the station's call ID to obtain an X.25 network address from data set $.SYSX25. It transmits the X.25 network address and the protocol ID and user data defined in the circuit station to the network in a call request packet. When the network responds with a call connected packet, the virtual call is established. If the circuit station has usage STD+, the IOCP sends a call connected control message to the circuit station's X.25 control message link. If there is no CALLID specified for the station, the network must be able to obtain the X.25 network address to call by using other facilities.

  If the network does not respond with a call connected packet within the time specified in Recommendation X.25, or if the network responds with a clear indication packet, the IOCP closes the circuit. If the circuit station has usage STD+, the IOCP sends a clear indication control message to the circuit station's X.25 control message link.

- USERINIT: When the circuit station is started, the IOCP takes no action until a call request control message is sent to the circuit station. The IOCP uses the information in the call request control message to build a call request packet and transmits it to the network. From this point on, call establishment occurs in the same way as for circuit stations with contact INIT.

You cannot define contact USERINIT for a circuit station with usage CF.

Sending Messages

When a message is placed on a circuit station's queue, the IOCP retrieves it and passes it on to $XHCS for transmission over the HDLC line.
How the IOCP handles the data varies according to the circuit station usage:

- CF: the IOCP obtains the 24-byte Communications Facility message header and appends the message data to it. $XHCS transmits the message header and data as X.25 data packets.
- STD or STD+: the IOCP determines whether the message is data or control information from its Communications Facility message type:
  - A Communications Facility data message (one received with return code -1) is data information.
  - A Communications Facility status message (one received with return code +6) is control information.

In either case, the message must begin with an X.25 header, which the IOCP uses to determine the type of control information. The IOCP removes the X.25 header before passing the message on to $XHCS. $XHCS transmits data information as X.25 data packets and control information as X.25 control packets.

Receiving Messages

$XHCS reads data from the HDLC line and passes it on to the IOCP. $XHCS provides a pointer to the circuit station associated with the logical channel over which the data was read. (The pointer is defined by the IOCP when it opens a circuit.) $XHCS also provides an indication of the X.25 packet type—whether it is data or control information.

The IOCP processes the data according to its type and the circuit station usage.

- If usage is CF or STD and the data is control information, the IOCP logs its receipt and responds as required, but it does not send the data on.
- If usage is STD+ and the data is control information, the IOCP logs its receipt, prefixes it with an X.25 header, and sends it as a Communications Facility status message. The destination is the circuit station's alternate link, if it has one; otherwise the destination is the circuit station's direct link.
- If usage is CF and the data is data information, the IOCP assumes that first 24 bytes are a Communications Facility message header. It sends the message header and data to the message dispatcher. The message dispatcher dispatches the message according to the destination information in the message header.
- If usage is STD or STD+ and the data is data information, the IOCP prefixes it with an X.25 header and sends it as a Communications Facility data message to the circuit station's direct link.

If the IOCP's user station has a direct link or an alternate link (user intercept processing is in effect), then, after determining the message destination, the IOCP:

- Builds a Communications Facility message header that includes the message destination and appends the data or control information to the message header.
- Sends the message header and data information as a Communications Facility data message to its direct link.
- Sends the message header and control information as a Communications Facility status message to its alternate link, if it has one; otherwise, sends them to its direct link.

Terminating Communication

Data flow through a permanent virtual circuit stops when the circuit station is stopped.

A virtual call is terminated in the following ways:

- An SVC circuit station is stopped. The IOCP transmits a clear request packet to the network and waits for a clear confirmation packet. When the clear confirmation packet arrives, the IOCP closes the circuit.
- A clear request control message is sent to an SVC circuit station with usage STD+. The same actions occur as when the station is stopped. In addition, the IOCP sends a clear confirmation control message to the circuit station's X.25 control message link.
- The IOCP receives a clear indication packet from the network for a switched virtual circuit. If the circuit station usage is STD+, the IOCP sends a clear indication control message to the circuit station's X.25 control message link.

If the virtual call is terminated by a clear request control message or a clear indication packet, the circuit station may remain active depending on its contact type and mode. If the circuit station contact type is INIT and the station is in STOP mode (which is the default), the IOCP stops the station. In all other cases (contact type WAIT, contact type USERINIT, or contact type INIT with RETRY mode), the station remains active. Another virtual call can then be established, as explained in "Establishing Communication" on page 92.

Restrictions

You can have only one active Local Communications Controller connection, BSC point-to-point connection, or X.25 connection through circuit stations with usage CF between two Series/1s.

The X.25 IOCP has the same restrictions as XHCS. Some of these are:

- Conformance to CCITT Recommendation X.25 as amended in 1980
- No datagram support
- No reject packet support.

The X.25 IOCP does not support diagnostic packets.

The X.25 IOCP provides limited data circuit-terminating equipment (DCE) support. Only permanent virtual circuits are supported for DCE lines. A Series/1 where the X.25 IOCP is running cannot be used as an X.25 packet-switching network, nor can it be part of an X.25 packet-switching network.

The X.25 IOCP provides limited D-bit support. Because the Communications Facility has no end-to-end protocol between nodes, the X.25 IOCP sends the delivery confirmation packet after successful completion of the Communications Facility SEND instruction. Communications Facility application programs cannot send D-bit data messages. If you require message integrity, you must implement it within the application programs in the communicating nodes.

Remote IPL and remote disk are not supported over an X.25 connection.

3270 Control IOCP, $.IO0AC0

The 3270 control IOCP manages communication with physical 3270 control units, Series/1s emulating 3271 control units, 5280 Distributed Data Systems running in 3270 emulation mode, or System/38s running in 3270 emulation mode. The connection is a leased or switched binary synchronous communications (BSC) line, operating in multipoint mode. The IOCP can manage multiple lines. There may be up to 32 control units, each with up to 32 terminals and printers, associated with each line.

Data passing between the IOCP and the real or emulated 3270 system is in 3270 data stream format. The IOCP uses standard BSC multipoint protocol, as described in *An Introduction to the IBM 3270 Information Display System,* GA27-2739.

You can use the IOCP in these ways:

- For communication between host programs and 3270 printers or terminals attached to the Series/1. See example 3 in the chapter "Sample Configurations" on page 269.

- For communication between Series/1 programs and 3270 printers or terminals.

- For communication between Series/1s, when this IOCP is running in one Series/1 and the 3270 emulation IOCP ($.IO0AE0) is running in the other Series/1. See example 7 in the chapter "Sample Configurations" on page 269. The example explains how a message is routed from its origin in one node to its destination in another node.

- To IPL a remote Series/1 that is connected to the BSC line and strapped as a multipoint tributary station that can be IPLed.

- To enable a remote Series/1 that is connected to the BSC line to access disks attached to the local Series/1. The remote Series/1 must have remote disk support in the EDX supervisor, but the Communications Facility need not be running.

The IOCP can manage these control units:

- 3271 model 1 or 2
- 3276 model 1, 2, 3, or 4
- 3274 model 1C, 21C, 31C, 41C, 51C, or 61C
- Series/1 emulating a 3271 control unit
- 5280 Distributed Data System running in 3270 emulation mode
- System/38 running in 3270 emulation mode.

The terminals and printers that can be attached to these control units are described in the *3270 Introduction.* The Communications Facility does not support the 3270 extended color or extended highlighting features. Restrictions on the support of screens larger than 1920 characters (24 rows of 80 columns) are explained in the section "Restrictions" on page 101.

Hardware Requirements

Use of this IOCP requires a Series/1 BSC adapter (feature number 2074, 2075, or 2093/2094), or a multifunction attachment (feature number 1310).

Strap the BSC adapter (jumpers on or off) as follows, according to the type of connection:

	DTR	No Ring Inhibit	Switched Line (MP address bit 7)	Multipoint Tributary
Leased Line	on	on	off	off
Switched Manual	on	on	off	off
Switched Auto-Answer	off	off	on	off

You may need to install other jumpers, depending on the type of modem, as explained in the *Binary Synchronous Features Description.*

EDX Requirements

Include EDX BSC support in the EDX supervisor. Use the BSCLINE statement to define each BSC line. Specify TYPE=MC if the line is leased, TYPE=SM if the line is switched manual, or TYPE=SA if the line is switched auto-answer.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0AC0, type=user.
- Each BSC line. Type=line, subtype=3271C.
- From 1 to 32 control units for each line. Type=terminal, subtype=3271.
- From 1 to 32 terminals or printers for each control unit. Type=terminal, subtype=3277 (terminal) or 3286 (printer).

Link each station that represents a terminal or printer with a station that represents an emulated terminal or printer or an application program. No links are required for the line stations or the stations that represent control units.

IOCP Operation Overview

The 3270 control IOCP consists of one main task and a subtask for each started line station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt line stations. Each line subtask processes data received from the terminals on the line and messages sent to the terminal stations associated with the line station.

The IOCP acts as a multipoint control station. It issues general polls to solicit data from terminals. It issues a specific poll only to receive 3270 status information when a control unit indicates that there is status pending for a terminal or printer.

Input data and 3270 status messages contain the polling address of the device from which the data originated. The IOCP sends data as a Communications Facility message, specifying the terminal station with the indicated polling address as the origin. If the IOCP has a direct link vector, it specifies that station as the destination; otherwise, it specifies a null destination.

When the IOCP has polled all control units, it processes pending output for each terminal station associated with the line station. It retrieves a message from a terminal station's queue, issues a selection sequence using the station's selection address, and sends the data. It continues sending data until the station's message queue is empty, then proceeds to the next terminal station. When it has processed all pending output, the IOCP resumes polling.

If the maximum retry value has not been set in the control unit station with the CP F RETRY command, the control unit will continue to be polled and performance may be affected. If the retry maximum has been set and the maximum number of retries has been reached, the control unit station will be set inactive. The control unit will not be polled until the control unit station is restarted.

Input from Terminals

Input from terminals may be one or more blocks of data. The IOCP accumulates the input data until it receives a block that ends with ETX. If the data begins with X'1002' (DLE/STX), the IOCP removes the DLE before processing the data.

If the terminal station that represents the device from which the data originated is in non-display mode and the data does not begin with DLE/STX, the IOCP translates lowercase characters to uppercase. If the station is in text mode, or the data begins with DLE/STX, the IOCP doesn't translate the data.

Basic Mode Input

If the terminal station is in basic mode, the IOCP processes input data as described in the preceding paragraphs and then sends the message to its destination.

Record Mode Input

If the terminal station is in record mode, the IOCP processes input data according to its content.

If the data is a 3270 status message, the IOCP discards it.

If the data is a short read (one ended by the CLEAR key or a PA key), the IOCP sends the terminal a data stream that clears the screen, displays "READY" in row 22, and unlocks the keyboard.

In all other cases, the IOCP removes the following bytes from the input data:

- The last byte (ETX)
- The first 6 bytes (STX, polling address, attention ID, cursor address)
- The seventh through ninth bytes, if the seventh byte is SBA (set buffer address).

If there is any data left, the IOCP sends it to its destination and sends the terminal a data stream that makes rows 23 and 24 an unprotected field, positions the cursor at row 23, and unlocks the keyboard.

Entering Commands

You can enter CP or PD commands from a 3277 terminal, as follows:

1. Press the CLEAR and RESET keys at the same time. This places the terminal in unformatted mode without sending data to the Series/1.
2. Type the command in uppercase. If you enter a PD command, include the prefix CP. If you enter a command that displays information (such as CP Q) and you want the display to appear at your terminal, specify the terminal's station name as the destination parameter. The destination does not default to the terminal where the command was entered, as it does when you enter commands at an EDX terminal.
3. Press the TEST REQ key. The IOCP will receive the data, remove the test request header, and send the command to the command processor.

You can't enter commands from 3276, 3278, or 3279 terminals, because they don't have the RESET/CLEAR capability.

Output to Terminals

When the IOCP receives a message destined for a terminal station, it sends the message on to the terminal or printer that the station represents. The messages may contain binary data, remote IPL data, basic mode data, or record mode data.

Binary Output

If a message begins with DLE/STX (X'1002'), the IOCP removes ETX (X'03'), if there is one, from the end and sends the data with a BSC transparent write command. You can transmit binary data only to an emulated terminal. A real 3270 system rejects data that starts with DLE.

Remote IPL Output

If a message begins with DC1/DC1 (X'1111'), the IOCP assumes the message is an IPL bootstrap. It inserts the terminal station's 3270 control unit address and sends the data. This causes an IPL of the remote Series/1 whose BSC adapter is strapped with that control unit address.

Basic Mode Output

If a message begins with STX/ESC (X'0227'), the IOCP assumes the message is a 3270 data stream. It checks that the third byte is a valid, supported 3270 command code. The supported commands are: erase all unprotected, read modified, read buffer, write, erase write, and erase write alternate. If the last byte of the message is an ETB (X'26'), the IOCP replaces it with ETX. If the last byte is neither ETB nor ETX, the IOCP appends an ETX.

Record Mode Output

If a message doesn't begin with DLE, DC1, or STX, the IOCP formats it as a 3270 data stream before sending it. If the data is for a printer, the IOCP appends an ETX to the end of the message and the following 4 bytes to the beginning of the message:

X'02'	STX
X'27'	ESC
X'F5'	Erase/write command
X'F8'	Write control character for 80-character print line

If the data is for a terminal, the IOCP appends 9 bytes to the beginning of the message:

X'02'	STX
X'27'	ESC
X'F1'	Write command
X'C3'	Write control character that unlocks the keyboard and resets modified data tags
X'11*xxxx*'	A set buffer address sequence, such that the first output message appears on row 1 of the screen, the second on row 2, and so on, wrapping around from row 21 to row 1. If the operator enters data, it is processed as described in "Input from Terminals" on page 99, and output resumes at row 1.
X'1D40'	Start field order for an unprotected field

The IOCP also appends 4 bytes to the end of the message:

X'125B5F'	Erase unprotected to row 22, column 80
X'03'	ETX

Restrictions

The IOCP does not support the 3270 copy command.

The IOCP does not support the 3270 write structured field command.

The IOCP does not support the sending of multiblock messages to terminals. It handles each message sent to a terminal station as a single complete message.

The IOCP does not support the TEST REQ key, except as described in "Entering Commands" on page 100.

Neither the IOCP nor the Communications Facility language extensions support the 3270 extended color or extended highlighting features.

The IOCP supports screens larger than 1920 characters (24 rows of 80 characters), but other components of the Communications Facility do not. The panel design aid, $.PANEL, supports only 1920-character screens. The language extensions support larger screens to a limited extent—the GET and PUT instructions assume that a field starting at screen location 1919 wraps around to location 0.

Series/1-to-System/370 Channel Attach IOCP, $.IO0AD0

The Series/1-to-System/370 channel attach IOCP allows you to connect a Series/1 and a host processor in such a way that the Series/1 appears to the host as a local 3272 control unit with terminals and printers. The IOCP can manage multiple channel attachments to the same or different host processors. There may be up to 32 emulated terminals and printers associated with each channel attachment.

Applications running in the host communicate with the Series/1 the same way they communicate with an actual local 3270 system, as described in the *3271 Guide*. Although the IOCP operates as a local system, it sends and receives 3270/BSC data stream messages. It removes BSC and 3270 control characters from messages before sending them to the host. It appends BSC and 3270 control characters to messages received from the host before sending them to their destination.

You can use the IOCP in these ways:

- For communication between host programs and terminals or printers attached to the Series/1. See example 2 in the chapter "Sample Configurations" on page 269.

- For communication between host programs and Series/1 programs.

Hardware Requirements

Use of this IOCP requires the IBM Series/1-System/370 channel attachment, which consists of:

- IBM Series/1-System/370 Channel Attachment (feature number 1200)
- IBM 4993 model 1 Series/1-System/370 Termination Enclosure.

EDX Requirements

Include EDX EXIO support in the EDX supervisor. Use the EXIODEV statement to define each channel attachment.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0AD0, type=user.

- Each channel attachment. Type=line, subtype=CA.

- From 1 to 32 emulated devices for each channel attachment. Type=terminal, subtype=port.

Link each station that represents an emulated device with a station that represents a real terminal or printer or an application program. No link is required for line stations.

IOCP Operation Overview

The channel attach IOCP consists of one main task and two subtasks for each started line station—a message task and an attention task. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt line stations. Each attention task processes commands and data received from the host. Each message task processes messages sent to the terminal stations associated with the line station.

When a message is available at one of the terminal stations, the message task issues a set attention command to notify the host that data is available. When the host program is ready to receive the data, it issues a read modified command, specifying the appropriate port number. The attention task responds by receiving a message from the queue of the station with that port number and writing it to the host.

The host transmits data to the Series/1 by issuing an erase/write, an erase/write alternate, or a write command for a particular port. The attention task locates the terminal station with the specified port number. It receives the data and sends it as a Communications Facility message, specifying the terminal station as the origin. If the IOCP has a direct link vector, it specifies that station as the destination; otherwise, it specifies a null destination. If the host selects an undefined port or one whose terminal station is stopped, the IOCP returns an intervention required indication to the host.

Because of hardware timing constraints, the IOCP doesn't process a read buffer command or an unsolicited read modified command as a real 3272 control unit does. (An unsolicited read modified command is one sent by the host without the host's having received an attention interrupt indicating that data is available.)

When the terminal station is in basic mode, the IOCP sends the command to its destination and waits for a response. The hardware will time-out if a response is not received within 480 milliseconds. If a hardware time-out occurs, the host may retry the command or issue a new command. If the host issues a new command, it will ignore any response to the previous read. The IOCP's processing of the terminal's message queue after the time-out depends on the host command received:

- If the host reissues the read buffer or read modified command, the port has an additional 480 milliseconds in which to respond. (This is considered a retry.)

- If the host issues a command to a different port, the late response to the original read command is saved on the queue. It will be sent as the response if the host retries the original command.

- If the host issues a second command to the original port, and the command is different from the first (other than SELECT), the IOCP purges the response to the first command. (This action is considered a give-up.)

When the terminal station is in record mode, the IOCP responds to read buffer and unsolicited read modified commands by sending the data stream stream X'7D4040' to the host. This data stream is what results when a 3277 operator presses the ENTER key with a blank screen and the cursor at row 1, column 1.

Messages from the Host

The IOCP modifies data received from the host before sending it on to its destination.

Basic Mode Input

If the terminal station is in basic mode, the IOCP builds a data stream that looks like those sent by the 3270 emulation IOCP. The data stream begins with STX/ESC (X'0227') and one of the following command codes:

X'F1'	Write
X'F2'	Read Buffer
X'F5'	Erase/Write
X'F6'	Read Modified
X'6F'	Erase All Unprotected

This is followed by the data received from the host for a write or an erase/write command. The first byte of the data is a write control character.

In all cases, ETX (X'03') is appended to the end of the message.

You can use the IOCP to transmit binary data between a host program and a Series/1 program. The Series/1 program should be written to ignore the control information that the IOCP appends.

Record Mode Input

If the terminal station is in record mode, the IOCP ignores an erase all unprotected command. It removes the following control information from the data for a write or an erase/write command:

- The first byte (write control character)
- The second through fourth bytes, if the second byte is SBA (set buffer address)
- The fifth and sixth bytes, if the second byte is SBA and the fifth byte is SF (start field)

Printer Operation

Host programs can send messages to a printer through a terminal station that is linked to a station that represents a printer. The IOCP may notify the host that the message has been received before the message has a chance to be printed. As a result, the host may send printer messages at a higher rate than the printer can handle. To avoid this problem, you should define disk queuing for any printer that will be receiving messages from a host system over a channel attachment.

Messages to the Host

The IOCP sends the host data messages that were sent to terminal stations. It modifies messages before sending them, according to whether they contain basic mode or record mode data.

The IOCP discards 3270 status messages, messages that begin SOH/%/R/STX (X'016CD902'). It issues a log message to report the status information, except when the status is device end.

Basic Mode Output

If a message begins with DLE/STX (X'1002') or STX (X'02'), the IOCP assumes that it is a correctly formatted 3270 data stream. It removes the DLE (if any), the STX, and the next 2 bytes (the 3270 polling address). The IOCP removes the last byte of the message if it is ETB (X'26') or ETX (X'03').

Non-BSC Output

If a message doesn't begin with DLE or STX, the IOCP appends 3 bytes to the beginning of the message:

X'7D'	ENTER key attention ID
X'4040'	Cursor location (row 1, column 1)

Restrictions

The IOCP supports remove mode, which is described in the section "Remove and Non-Remove Modes" on page 76, but only for ports operating in basic mode.

Host programs should issue read buffer commands and unsolicited read modified commands only to a port operating in basic mode. Because of timing considerations, the host should issue these commands only when the destination is on a Series/1 that is attached to the host. If the port is not in basic mode, the host receives the message X'7D4040'.

The IOCP does not support the diagnostic read command.

The IOCP does not support the sending of multiblock messages to the host. It handles each message sent to an emulated terminal station as a single complete message.

The IOCP does not provide any message mapping services except as described for record mode output. If the data going to the host is to be processed by any host mapping services, the appropriate 3270 orders must appear in the data stream. Messages created by real 3270 devices or by 4978/4980/3101/7485 devices as supported by the Communications Facility are formatted correctly. If a user program creates messages, it must format those messages correctly.

The IOCP does not save the last message sent to the host. Therefore, if a read by the host fails, any retry is the responsibility of the program that manages the station that is communicating with the host:

- If the host is communicating with is a local Series/1 terminal, input data is available for retry until a write command is received or until the terminal user presses RESET and enters new data.
- If the host is communicating with a real 3270 device, input data is available for retry in the device's hardware buffer. The time that it takes to retransmit the data will probably cause the channel to time-out. It it does, a user application between the channel attach and 3270 control IOCPs is required.
- If the host is communicating with an application program, the user application must provide the function of saving the message.
- If the host is communicating with a terminal on a downstream Series/1, a user application between the channel attach IOCP and the downstream Series/1 can be written if message saving is required.
- If the host is communicating with the program dispatcher, a user application between the host and the channel attach IOCP is required, because the program dispatcher does not save transactions for retransmission. If the program dispatcher receives a read buffer or read modified command, it discards the command because the command isn't a transaction.

Performance Considerations

When a host application sends data too rapidly to the Series/1 through the channel attachment, the attachment may time out. To avoid this, the Series/1 should pace the host program. Any convenient pacing technique may be used; such as, sending an acknowledgement from a Series/1 application to the host.

Defining the channel attach terminal station as disk-queued or linking the station to a disk-queued station is a good method to avoid filling the Communications Facility buffer pools during large and high-speed transmissions of data. However, the additional overhead of disk I/O may cause time-outs during those transmissions. To avoid this problem, position $.SYSNET and the disk-queue data sets to promote the quickest retrieval. The following suggestions are listed in order of effectiveness:

- Locate $.SYSNET on a fixed-head disk.
- Locate $.SYSNET and the disk-queue data sets on different devices.
- Locate $.SYSNET and the disk-queue data sets close together on one device.

If you still have time-out problems, try pacing the host transmission.

When possible, the channel attachment feature card should be located adjacent (or as close as possible) to the processor card or cards. This technique minimizes time delays in processing channel attachment I/O.

If most of your messages are short ones, it may seem to you that the channel is busy for an excessive amount of time. This is because the Series/1 channel attachment card, unlike the real 3272 control unit, is an unbuffered device. This requires the channel to stay in a busy state from the time the host issues an I/O command until the data transfer is complete and the attachment presents channel-end and device-end to both the host and the Series/1. The same sequence is required for short as well as long messages; therefore, you might notice its effect more when the messages are short.

Unsolicited read commands may cause time-outs unless you provide a program to handle them, as discussed in the section "Restrictions" on page 106. Some host systems issue an unsolicited read command because their buffer is shorter than the message sent from the Series/1. If you experience time-outs on long messages, and your host system allows you to control the size of its buffers, increase the buffer size to that of the longest message sent from the Series/1 (plus the size of any host access method headers).

3270 Emulation IOCP, $.IO0AE0

The 3270 emulation IOCP allows you to connect a Series/1 and a host processor in such a way that the Series/1 appears to the host as a 3271 model 2 control unit with terminals and printers. The connection is a leased or switched binary synchronous communications (BSC) line, operating in multipoint mode. The IOCP can manage multiple connections to the same or different host processors. There may be one emulated control unit and up to 32 emulated terminals and printers associated with each communication line.

Data passing between the Series/1 and the host is in 3270 data stream format, using standard BSC multipoint protocol.

Applications running in the host communicate with the Series/1 the same way that they communicate with an actual 3270/BSC remote system, as described in the *3271 Guide.*

You can use the IOCP in these ways:

- For communication between host programs and terminals or printers attached to the Series/1. See example 1 in the chapter "Sample Configurations" on page 269.
- For communication between host programs and Series/1 programs.
- For communication between Series/1s, when the 3270 emulation IOCP is running in one Series/1 and the 3270 control IOCP ($.IO0AC0) is running in another Series/1. See example 7 in the chapter "Sample Configurations" on page 269. The example explains how a message is routed from its origin in one node to its destination in another node.

The host computers with which the Series/1 may be connected are IBM:

- Series/1
- System/370
- 30*xx* series
- 4300 series
- 8100 series
- System/3
- System/34
- System/7

Hardware Requirements

Use of this IOCP requires a Series/1 BSC adapter (feature number 2074, 2075, or 2093/2094), or a multifunction attachment (feature number 1310).

Strap the BSC adapter (jumpers on or off) as follows, according to the type of connection:

	DTR	No Ring Inhibit	Multipoint Tributary
Leased Line	on	on	on
Switched Manual	on	on	on
Switched Auto-Answer	off	off	on

You must also strap the multipoint address with the host polling address for this tributary. If you want the adapter to respond to a remote IPL, bit 0 of the address must be strapped on, so the host polling address must be 'C*n*' or 'D*n*'.

You may need to install other jumpers, depending on the type of modem, as explained in the *Binary Synchronous Features Description.*

A multifunction attachment does not require strapping. Instead you specify the polling address in the POLL operand of the EDX BSCLINE statement. For example:

```
POLL=(C2).
```

EDX Requirements

Include EDX BSC support in the EDX supervisor. Use the BSCLINE statement to define each BSC line. Specify TYPE=MT if the line is leased, TYPE=SM if the line is switched manual, or TYPE=SA if the line is switched auto-answer.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0AE0, type=user.
- Each BSC line. Type=line, subtype=3271E.
- One emulated control unit for each line. Type=terminal, subtype=3271.
- From 1 to 32 emulated terminals or printers for each line. Type=terminal, subtype=3277 (terminal) or 3286 (printer).

Link each station that represents an emulated terminal or printer with a station that represents a real terminal or printer or an application program. No links are required for the line stations or the stations that represent control units.

IOCP Operation Overview

The 3270 emulation IOCP consists of one main task and a subtask for each started line station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt line stations. Each line subtask processes data received from the host and messages sent to the terminal stations associated with the line station.

The IOCP acts as a multipoint tributary station, responding to polling and selection sequences received from the host. It can operate in a general poll or specific poll environment. When the IOCP receives a general poll, it processes pending output for each terminal station associated with the line station. It retrieves messages from the terminal stations' queues and sends them to the host. When the IOCP receives a specific poll, it processes pending output only for the terminal station with the specified polling address.

If the host issues only specific polls, you shouldn't start terminal stations that won't be polled. If messages are sent to terminal stations that are never polled and the messages are queued in storage, the message buffer pool can fill up, leaving no room for messages for other stations.

When the IOCP receives a selection sequence, it locates the terminal station with the specified selection address. It then receives one or more blocks of data from the host and sends each one as a Communications Facility message, specifying the terminal station as the origin. If the IOCP has a direct link vector, it specifies that station as the destination; otherwise, it specifies a null destination.

Messages from the Host

Each block of data received from the host must begin with X'0227' (STX/ESC) or X'100227' (DLE/STX/ESC) followed by a 3270 command code. If a block begins with DLE, the IOCP removes the DLE before processing the data.

3270 Commands

The only control command accepted is the erase all unprotected command.

Three write commands are accepted: write, erase write, and erase write alternate. These commands require that a write control character immediately follow the command code.

Two read commands are accepted for terminal stations in basic mode: read modified and read buffer. Both of these commands solicit terminal input. The IOCP sends the read command to its destination, receives the next message from the emulated terminal station's queue, and sends this message on to the host as a response to the read command.

If the response does not arrive within a certain period of time, the IOCP terminates the request by sending an EOT to the host. The IOCP waits for a maximum of ten intervals of 500 milliseconds each, for a total time of five seconds. You can use the CP F TIMEOUT command to set a different interval for a 3271E line station.

Basic Mode Input

If the selected terminal station is in basic mode, the IOCP sends the message unchanged to its destination (except for removing an initial DLE).

Record Mode Input

If the selected terminal station is in record mode, the IOCP reformats the data received from the host before sending it to its destination. The IOCP removes the following control information from the data:

- The last byte (ETX)
- The first 4 bytes (STX, ESC, command code, write control character)
- The fifth through seventh bytes, if the fifth byte is SBA (set buffer address)
- The eighth and ninth bytes, if the fifth byte is SBA and the eighth byte is SF (start field).

Printer Operation

When the IOCP receives data for an emulated printer station, with the start printer bit on in the write control character, it sends the message to its destination and then sends a WACK to the host, which indicates that the printer is busy. The host will not send more data for the printer until it receives a device end status message, which indicates that the printer is again ready. (If the host does select the printer again before receiving device end, it will receive a device busy status message.)

The program that manages the real printer to which the emulated printer is linked is responsible for sending a device end status message to the emulated printer station when the print operation is complete. When the IOCP retrieves the device end status message from the printer station's queue, it turns off its indicator that the printer is busy and sends the status message on to the host.

The program that manages the real printer is also responsible for retrying a print operation when the printer is not ready. The IOCP assumes that a print operation will complete successfully once it has sent the data to its destination. If the program that manages the real printer sends back an intervention required status message, the IOCP ignores it.

The IOCPs that manage printers retry print operations, when necessary, and send the required device end status message. If you write an IOCP to manage printers or an application program that processes data for emulated printers, you must do the same. The format of a status message is:

- A 4-byte heading: SOH (X'01')/%/R/STX (X'02')
- 2 bytes for the control unit and device address, which the 3270 emulation IOCP fills in
- The 2-byte status/sense code; X'C240' for device end
- ETX (X'03').

Messages to the Host

The IOCP sends the host data messages that were sent to terminal stations. The messages may contain binary data, basic mode data, or record mode data. The IOCP also sends 3270 status messages to the host.

Binary Output

If a message begins with DLE (X'10'), the IOCP inserts the terminal station's polling address in the third and fourth bytes, removes ETX (X'03'), if there is one, from the end, and sends the message to the host with a BSC transparent write command. The host program is responsible for handling the data, deblocking it if necessary.

Basic Mode Output

If a message begins with STX (X'02'), the IOCP assumes that it is a correctly formatted 3270 data stream. The IOCP inserts the terminal station's polling address in the second and third bytes. If the last byte of the message is ETB (X'26'), it replaces the ETB with ETX. If the last byte is neither ETB nor ETX, it appends an ETX. If the message is longer than 256 bytes, the IOCP sends it to the host in multiple blocks.

Record Mode Output

If a message doesn't begin with DLE or STX, the IOCP formats it as a 3270 data stream before sending it to the host. It appends an ETX to the end of the message and the following 6 bytes to the beginning of the message:

X'02'	STX
X'4040'	Control unit and device address
X'7D'	ENTER key attention ID
X'4040'	Cursor location (row 1, column 1)

The IOCP then handles the reformatted message as described for basic mode output.

3270 Status Messages

The IOCP sends 3270 status messages to the host to report error and exception conditions, as a real 3271 control unit does. The IOCP sends status messages to report conditions it detects. It also sends status messages that it retrieves from terminal stations' message queues. It detects status messages by the fact that they begin with SOH (X'01'). It sends status messages in response to a general or specific poll, according to the status condition.

The 3270 status/sense codes for conditions detected by the IOCP are:

Command Reject (X'4060'): The IOCP received an invalid or unsupported 3270 command from the host, or it received a read command for a terminal station in record mode.

Device Busy (X'C840'): The IOCP received and processed a message for an emulated printer, with the start printer bit on in the write control character. It has not yet received device end status from the corresponding real printer.

Device End (X'C240'): The status of a terminal station has changed from inactive to active. That is, the station has just been started or was stopped and then restarted.

Intervention Required (X'4050'): The IOCP received a specific poll with an undefined polling address, or it received a selection sequence for a terminal station that is stopped (its status is inactive), or it received a 3270 read command for a terminal station that has no active direct link.

Operation Check (X'40C1'): The IOCP received data from the host that did not begin with X'0227' or X'100227'.

Restrictions

The IOCP does not support the 3270 copy command.

The IOCP supports the chaining of write-type commands to a limited extent. Each chained write command that includes data should have an SBA (set buffer address) order immediately following the write control character. This restriction applies because the IOCPs that manage real terminals and printers don't keep track of the current buffer address between write commands.

The IOCP does not accept multiblock messages for the host. It handles each message sent to an emulated terminal station as a single complete message.

The IOCP does not provide any message mapping services except as described for record mode output. If the data going to the host is to be processed by any host mapping services, the appropriate 3270 orders must appear in the data stream. Messages created by real 3270 devices or by 4978/4980/3101/7485 devices as supported by the Communications Facility are formatted correctly. If a user program creates messages, it must format those messages correctly.

The IOCP can be used only in a normal (EBCDIC) data mode, except when the host system supports transmission of binary data. Only EBCDIC data can be transmitted to or from the Series/1 when you're using IMS or CICS 3270 terminal support.

The IOCP does not save the last message sent to the host. It expects that the host will solicit input by polling, as is recommended for remote 3270 configurations. If the host issues read commands (read modified or read buffer) to request retransmission of the prior message, the response to those commands is the responsibility of the program that manages the station that is communicating with the host:

- If the host is communicating with a Series/1 terminal or a real 3270 device, input data is available for retransmission until a write command is received or until the terminal user presses RESET and enters new commands.
- If the host is communicating with an application program, the user application must be prepared to retransmit the prior message if a read command is received.
- If the host is communicating with a program dispatcher, a user application between it and the IOCP is required. The program dispatcher discards a read buffer or read modified command because it is not a transaction.

Series/1-to-Series/1 BSC IOCP, $.IO0A10

The Series/1-to-Series/1 BSC IOCP manages communication between two Series/1s connected by a leased or switched binary synchronous communications (BSC) line, operating in point-to-point mode. The IOCP can manage multiple connections to different Series/1s.

You can use the IOCP in these ways:

- To send messages from stations in one Series/1 to stations in another Series/1 where $.IO0A10 is running. See example 6 in the chapter "Sample Configurations" on page 269. The example explains how a message is routed from its origin in one node to its destination in another node.
- To IPL a remote Series/1 that is connected to the BSC line and strapped for remote IPL.
- To enable a remote Series/1 that is connected to the BSC line to access disks attached to the local Series/1. The remote Series/1 must have remote disk support in the EDX supervisor, but the Communications Facility need not be running.

Hardware Requirements

Use of this IOCP requires a Series/1 BSC adapter (feature number 2074, 2075, or 2093/2094), or a multifunction attachment (feature number 1310).

Strap the BSC adapter (jumpers on or off) as follows, according to the type connection:

	DTR	**No Ring Inhibit**	**Switched Line (MP address bit 7)**	**Multipoint Tributary**
Leased Line	on	on	off	off
Switched Manual	on	on	off	off
Switched Auto-Answer	off	off	on	off

If you want the adapter to respond to a remote IPL, strap bit 0 of the multipoint address.

You may need to install other jumpers, depending on the type of modem, as explained in the *Binary Synchronous Features Description.*

EDX Requirements

Include EDX BSC support in the EDX supervisor. Use the BSCLINE statement to define each BSC line. Specify TYPE=PT if the line is leased, TYPE=SM if the line is switched manual, or TYPE=SA if the line is switched auto-answer.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0A10, type=user.
- The local end of each BSC line. Type=line, subtype=CPU.
- The remote end of each BSC line. Type=line, subtype=CPU.
- Each remote station to which local stations will send messages.

Link each remote line station to its corresponding local line station. Link the other remote stations as required for the communication you want. Be sure that each node has a unique node assignment and that the network address of each remote station is the same as the network address of the corresponding real station in the remote node.

IOCP Operation Overview

The IOCP consists of one main task and a subtask for each started line station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt line stations. Each line task processes data read from the line and messages that the message dispatcher has placed on the line station's queue.

Except for the sending of an IPL bootstrap, exchange of messages does not begin until both IOCPs (one in each Series/1) are started and have established communication with each other. Thereafter, they operate in full conversational mode, writing and reading messages.

When a message is placed on the line station's queue, the IOCP writes the entire message (header and data) directly from the message buffer pool. The IOCP writes it as transparent data, replacing the first 2 bytes of the message header with DLE/STX (X'1002'). When the line station's queue is empty, the IOCP writes just DLE/STX to maintain synchronization with the other IOCP.

When the IOCP reads a message (not just DLE/STX) from the BSC line, it obtains a work area from the message buffer pool, moves the message to the work area, and sends it to the message dispatcher. The message dispatcher dispatches the message according to the destination information in the message header.

Restrictions

The IOCP must run in the same partition as the Communications Facility control program ($.CFS or $.CFD).

You can have only one active BSC point-to-point connection, Local Communications Controller connection, or X.25 connection through circuit stations with usage CF between two Series/1s.

Point-to-Point IOCP, $.IO0A20

The point-to-point IOCP manages communication between a Series/1 and the following IBM 2770 and 3741-like point-to-point devices:

- 6640 Ink Jet Printer
- 6670 Information Distributor
- Office System/6 Information Processor
- Displaywriter
- 5520 Administrative System
- 5280 Distributed Data System
- System/23 Datamaster
- Another Series/1

The connection can be a leased or switched binary synchronous communications (BSC) line, operating in point-to-point mode. The IOCP can manage multiple connections to the same or different devices.

You can use the IOCP in these ways:

- To exchange data with one of the devices listed above.
- To send documents from one device, such as a Displaywriter, to another device, such as a 6640 printer, through the Series/1. Connect each device to the Series/1 with a point-to-point BSC line and link the line stations to each other.
- To send documents from multiple devices to a single destination device through the Series/1. You must provide an application program to prevent documents from being interspersed at the destination.

Hardware Requirements

Use of this IOCP requires a Series/1 BSC adapter (feature number 2074, 2075, or 2093/2094), or a multifunction attachment (feature number 1310).

Strap the BSC adapter (jumpers on or off) as follows, according to the type of connection:

	DTR	No Ring Inhibit	Switched Line (MP address bit 7)	Multipoint Tributary
Leased Line	on	on	off	off
Switched Manual	on	on	off	off
Switched Auto-Answer	off	off	on	off

You may need to install other jumpers, depending on the type of modem, as explained in the *Binary Synchronous Features Description.*

EDX Requirements

Include EDX BSC support in the EDX supervisor. Use the BSCLINE statement to define each BSC line. Specify TYPE=PT if the line is leased, TYPE=SM if the line is switched manual, or TYPE=SA if the line is switched auto-answer.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0A20, type=user.
- Each BSC line. Type=line, subtype=PTPT.

Link each line station to the station that is to receive the data sent by the remote device.

IOCP Operation Overview

The point-to-point IOCP consists of one main task and a subtask for each started line station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt line stations. Each line task processes data read from the line and messages sent to its line station's queue.

The IOCP retrieves a message from a line station's queue and writes it to the BSC line with a write conversational instruction. If the IOCP receives text in response, it sends the data as a Communications Facility message, specifying the line station as the origin. If the IOCP has a direct link vector, it specifies that station as the destination; otherwise, it specifies a null destination.

After sending all output messages, the IOCP checks the line for input. If it receives data, it processes the data as already described. It continues with alternating output and input operations until the line station is stopped or halted.

Input Operations

The IOCP handles input data according to whether the line station is in basic mode or record mode and whether the data is transparent or not.

Basic Mode Input

If the line station is in basic mode, the IOCP sends the data without modification to its destination.

Record Mode Input

If the line station is in record mode and the data is transparent (it begins with DLE/STX, X'1002'), the IOCP sends the data without modification to its destination. The IOCP also issues a warning log message to report that it received transparent data for a station in record mode.

If the data is not transparent, the IOCP removes the first character (STX) and the last character (ETB or ETX) from the data before sending it to its destination.

Output Operations

The IOCP handles output data according to whether it is basic or record mode data and whether the line station is in transparent mode or not.

Basic Mode Output

The data is basic mode data if it begins with DLE/STX (X'1002') or STX. The IOCP handles multiple-block or single-block basic mode data.

In multiple-block data, each block except the last one ends with ETB (X'26'). After sending the first block, the IOCP retrieves the next message from the line station's queue and sends it. (If the queue is empty, it waits until a message arrives.) It continues this process until it has sent the last block. If the last block doesn't end with ETX (X'03'), the IOCP appends an ETX before sending it.

Single-block data is data that doesn't end with an ETB. If the last byte isn't an ETX, the IOCP appends an ETX before sending the data.

Record Mode Output

If the data doesn't begin with DLE or STX, the IOCP appends BSC control information before sending it. If the line station is in transparent mode (XON), the IOCP appends DLE/STX to the beginning of the message. If the line station is not in transparent mode (XOFF), the IOCP appends STX to the beginning of the message and ETX to the end of the message.

Restrictions

You can use the point-to-point IOCP for communication between Series/1s, but it is not as efficient as the Series/1-to-Series/1 IOCP ($.IO0A10).

If you use this IOCP for communication between Series/1s with the program dispatcher, you must set record mode and transparency on for the line station in each cell.

If you transmit large amounts of data with this IOCP, you should assign a disk queue to the line station so as to avoid flooding the message buffer pool.

3101 IOCP, $.IO06F0

The 3101 IOCP allows you to use 3101 terminals (models 12, 13, 22, and 23), 3161, 3163 or 3164 terminals in 3101 emulation mode, and IBM PCs running in 3101 emulation mode as if they were 3277 model 2 terminals. The IOCP, $.IO06F0, manages communication with 3101 terminals connected to the Series/1 by a feature programmable communications adapter, a multifunction attachment, or a 4950/5170-495 terminal/host adapter.

You can use the IOCP in these ways:

- For communication between 3101 terminals and host programs to which the Series/1 appears to be a 3270 system. See example 2 in the chapter "Sample Configurations" on page 269.
- For communication between 3101 terminals and Series/1 programs. See example 5 in the chapter "Sample Configurations" on page 269.

Hardware Requirements

Use of this IOCP requires one of these attachment features:

- Feature programmable communications adapter (feature number 2095/2096 or feature number 2095/RPQ D02350). Up to eight 3101s can be attached locally or remotely with feature number 2095/2096. Up to eight 3101s can be attached locally with an RS422A electrical interface using feature number 2095/RPQ D02350. When any of the terminals are being used as Communications Facility terminals, the one at subaddress 0 must be active; the station that represents it must be started first and stopped last. (Subaddress 0 is the device address that is a multiple of 8.)
- Multifunction attachment (feature number 1310). Up to four 3101s can be attached. Any of the four can be attached locally with an RS422A electrical interface; this interface supports only models 13 and 23. Alternatively, the 3101 at the base address can be attached locally or remotely with an RS232C electrical interface; this interface supports all models.
- A terminal/host adapter is required for the 4950 or the 5170-495. On the 4950, up to four terminals can be attached locally to the terminal/host adapter using an RS422A electrical interface. One terminal can be attached locally or remotely to the Personal Computer Asynchronous Communications Adapter (feature number 1502074) using an RS232C electrical interface.

  On the 5170-495, up to four terminals can be attached locally to the terminal/host adapter using an RS422A electrical interface. Up to two terminals can be attached locally or remotely to the terminal/host adapter using an RS232C electrical interface.

You must set the 3101 setup switches according to the type of attachment and whether you're using the 3101 as an EDX terminal or a Communications Facility terminal. Descriptions of the setup switches are in the *3101 Display Terminal Description* manual. Figure 21 on page 122 shows required and recommended switch settings, assuming a bit rate of 9600. The 32 switches are shown in groups of 8; each number (1-8) represents a switch. A letter above the number of the

Feature Programmable Communications Adapter or Multifunction Attachment
Communications Facility Terminal

```
    F I     I          I     I I  I   X            S     S S     S
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
I I M   X I   X  I I I   I I        I   X I I X X    S S     S S
```

Feature Programmable Communications Adapter or Multifunction Attachment
EDX Terminal

```
B R F I     I          X     X X  I   X X          S     S S     S
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
C   M   X I   X  I I I   I I        I     I I X X    S S     S S
```

4950/5170-495 Terminal/Host Adapter
Communications Facility Terminal

```
      I     I          I     I I  I   X            S     S S     S
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
I I I   X I   X  I I I   I I        I   X I I X X    S S     S S
```

4950/5170-495 Terminal/Host Adapter
EDX Terminal

```
B R   I     I          X     X X  I   X X          S     S S     S
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
C   I   X I   X  I I I   I I        I     I I X X    S S     S S
```

B = Device is defined to EDX as a block mode device.
C = Device is defined to EDX as a character mode device.
F = Feature programmable communications adapter or multifunction attachment with LMODE of PTTOPT, SWITCHED, or LOCAL.
I = Must be in the position shown.
M = Multifunction attachment with LMODE of RS422.
R = Must be in the position shown for a character mode device; can be in either position for a block mode device.
S = Depends on line speed used. 9600 bps is shown.
X = May be in either position, but X is recommended.

Figure 21. 3101 Setup Switches

corresponding switch indicates that the switch is up, and a letter below the number indicates that the switch is down.

Use Figure 22 on page 123 as a guide to setup a 3161, 3163, or 3164 device in 3101 emulation mode.

```
                              SETUP MENU
-----------------------------------------------------------------------

Machine Mode       IBM 3101
Operating Mode     ECHO                    1

                                Main Port          Auxiliary Port
                               -----------        ----------------
  Interface                     RS-232C  2
  Line Control                  PRTS
  Line Speed (bps)              9600     3         9600
  Parity                        SPACE                      SPACE
  Turnaround Character          CR
  Stop Bit                      2                          2
  Word Length (bits)            7        4         7 4
  Response Delay (ms)           100      4
  Break Signal (ms)             500      4

Terminal ID:                             4
                  (Up to 20 alphanumeric characters are allowed.)
```

Figure 22. 3161/3163/3164 Device Configuration Menu

where:

1

Specify ECHO for use as a Communications Facility terminal. Specify CHAR or BLOCK for use as an EDX terminal as defined in your EDX terminal definitions.

2

Specify RS-232C for a feature programmable communications adapter or a multifunction attachment with LMODE of PTTOPT, SWITCHED, or LOCAL. Specify RS-422A for a 4950 or a 5170-495 Series/1 system unit, Terminal/Host Adapter or a multifunction attachment with LMODE of RS422.

3

Depends on line speed. 9600 bits per second is shown.

4

Specified for IBM 3161 only.

Figure 23 shows the recommended configuration for the selection line for a 3161/3163/3164 terminal running as a Communications Facility device.

```
SCROLL=OFF RETURN=CR LINE WRAP=ON AUTO LF=OFF SEND=LINE NULL SUPP=OFF
```

Figure 23. 3161/3163/3164 Selection Line

You must strap the feature programmable communications adapter to match the 3101 configuration, as explained in the EDX *Installation and System Generation Guide.*

You must jumper the interface as either an EIA (RS-232C) interface or a TTY (20mA current loop) interface. The other jumpers that you may need to modify are the request to send (RTS) jumper, the data terminal ready (DTR) jumper, and the data carrier detect (DCD) jumper. You must also jumper the line speed range appropriately.

No strapping is required for the multifunction attachment or the terminal/host adapter on the 4950 and 5170-495. The EDX supervisor definition statements ADAPTER and TERMINAL or CFTERM define the 3101 configuration.

If you are using a remote connection, refer to your modem guide for the possible switch settings on your particular modem. Most modems provide switch settings for data terminal ready (DTR), auto-answer (AA), and carrier detect (CD). Depending on the modem you use, you may need to modify the carrier detect switch setting on the Series/1 modem to enable the remote 3101 to make a connection. See Appendix A, "Configuring and Connecting 3101 Display Terminals" on page 452.1 for information on how to connect a 3101 to $.IO06F0.

EDX Requirements

3101s on a terminal/host adapter or a feature programmable communications adapter require no EDX support if you plan to use them only as Communications Facility terminals. If you're also going to use them as EDX terminals or as Communications Facility work session controller terminals, include EDX terminal support in the EDX supervisor, and use the TERMINAL statement to define each 3101.

For 3101s on a multifunction attachment, include EDX multifunction attachment support in the EDX supervisor, and use the ADAPTER statement to define the attachment. If you plan to use the terminals only as Communications Facility terminals, use the Communications Facility CFTERM instruction to define each 3101; no EDX support is required. If you're also going to use them as EDX terminals or as Communications Facility work session controller terminals, include EDX terminal support in the EDX supervisor, and use the TERMINAL statement to define each 3101.

You must include EDX unmapped storage support in the EDX supervisor.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO06F0, type=user.
- Each 3101. Type=device, subtype=3101F.

When you define a 3101 device station to the Communications Facility, specify auto-answer OFF if you have a local connection, or specify auto-answer ON if you have a remote connection. Specify a bit rate to match the attachment card line speed, the modems (if any), and the bit rate setting on the 3101.

If you have defined the 3101s to EDX, use station names that are different from the EDX terminal names.

Link each device station with a station that represents an emulated terminal or an application program.

Supporting Multiple 3101s

The IOCP can manage multiple 3101s concurrently. Although the 3101 operates in interrupt-per-character mode, the instruction path length through the interrupt handling code is generally short enough so that there is minimal interference with the operation of other 3101s and with any batch work. The exact number of devices that can be supported depends on processor speed, 3101 line speed, available processor storage, and other tasks being executed concurrently.

The amount of available mapped and unmapped storage and the size of $.IO06F0's dynamic storage area are the limiting factors in determining the number of 3101s that can be managed by $.IO06F0. See "Planning Storage Requirements" on

page 213 for a discussion on determining the amount of processor storage and dynamic storage required to support multiple 3101s.

IOCP Operation Overview

The 3101 IOCP consists of a main task and a subtask for each started device station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt device stations. Each device subtask processes data received from a 3101 and messages sent to the station that represents that 3101.

The IOCP makes the 3101 appear as a 3277 to the operator and to the programs that communicate with the terminal. Although the 3101 is an unbuffered device, the IOCP makes it appear as a buffered device. Device buffering is done in Series/1 processor storage, using the unmapped storage management facilities of EDX.

Figure 24 summarizes the 3270 functions and the level of support that the 3101 IOCP provides. For more information about these functions, see the *3271 Guide*.

Function	Supported Yes	Supported No
Remote Command Codes		
Write (F1)	X	
Erase/write (F5)	X	
Erase/write alternate (7E)	X	
Read buffer (F2)	X	
Read modified (F6)	X	
Copy (F7)		X
Erase all unprotected (6F)	X	
Write Control Character		
Reset MDT	X	
Keyboard restore	X	
Sound alarm	X	
Start printer	X[4]	
Define printout format		X
Printer Control Orders	X[5]	
Buffer Control Orders		
Start field (SF)	X	
Set buffer address (SBA)	X	

Figure 24 (Part 1 of 2). 3270 Functions Supported on 3101

4 The start printer bit causes the screen to be printed at the 3102 printer attached to the 3101. This applies only to a 3101 model 2*n*.

5 Printer control orders display on a 3101 the same as they do on a 3277.

Function	Supported Yes	Supported No
Insert cursor (IC)	X	
Program tab (PT)	X	
Repeat to address (RA)	X	
Erase unprotected to address (EUA)	X	
Attributes		
Protected/unprotected	X	
Alpha/numeric	X	
Automatic skip	X	
High/normal intensity	X[6]	
Non-display	X	
Selector-pen detectable		X
Modified data tag	X	
Keyboard		
ENTER	X	
PF1-PF12	X	
PA1-PA3	X	
CLEAR	X	
ERASE EOF	X	
ERASE INPUT	X	
DEL	X	
RESET	X	
FM (field mark)	X	
DUP	X	
INS MODE	X	
TEST REQ		X
CURSR SEL		X
Cursor movement	X	
Lowercase alpha	X	
Card reader		X
Selector pen		X

Figure 24 (Part 2 of 2). 3270 Functions Supported on 3101

Input from Terminals

The IOCP reads input from a 3101 and holds it in a device buffer in processor storage. When the operator presses a program attention key (see the section "3101 Keyboard" on page 127), the IOCP formats the input data as a 3270/BSC data stream. The data stream is the same as if it had originated from a remote 3277 model 2 terminal in response to a 3270 read modified command.

6 A high-intensity field displays at normal intensity.

The IOCP sends the data as a Communications Facility message, specifying the device station as the origin. If the IOCP has a direct link vector, it specifies that station as the destination; otherwise, it specifies a null destination.

3101 Keyboard

Figure 25 on page 128 shows the 3101 keyboard as supported by the 3101 IOCP. The corresponding key numbers for the 3161/3163/3164 keyboards are shown along with the 3101 key numbers. A description of the supported functions follows. Pressing keys that have a legend of "Ignored" causes the alarm to beep; nothing is entered into the device buffer. Pressing keys that have a legend of "Local Function" causes the indicated function, but no data enters the system.

3161/3163/3164 Keyboard

Figure 26 on page 129 shows the 3161/3163/3164 keyboard as supported by the 3101 IOCP. The corresponding key numbers for the 3101 keyboard are shown along with the 3161/3163/3164 key numbers.

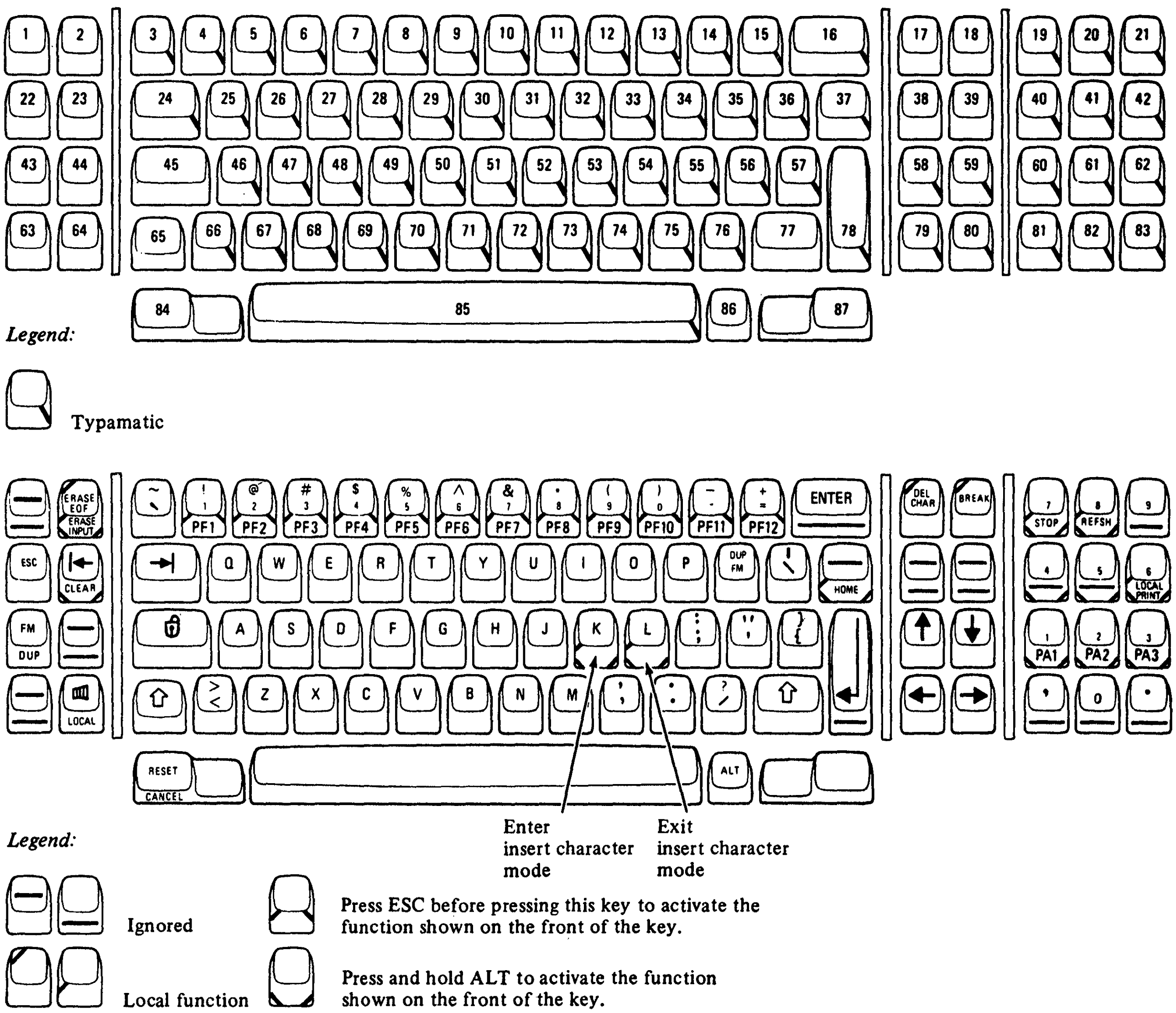
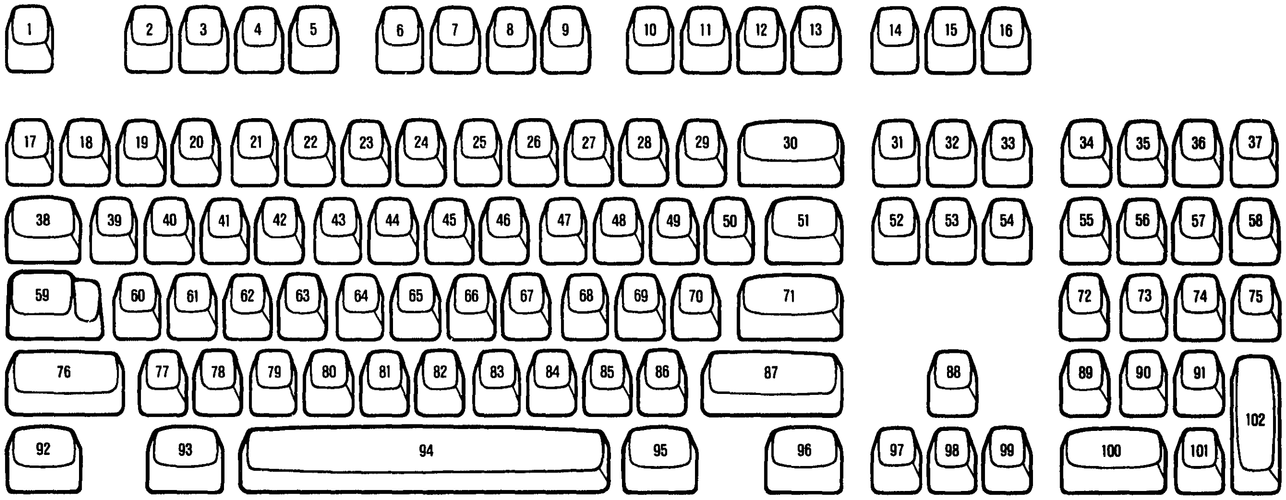
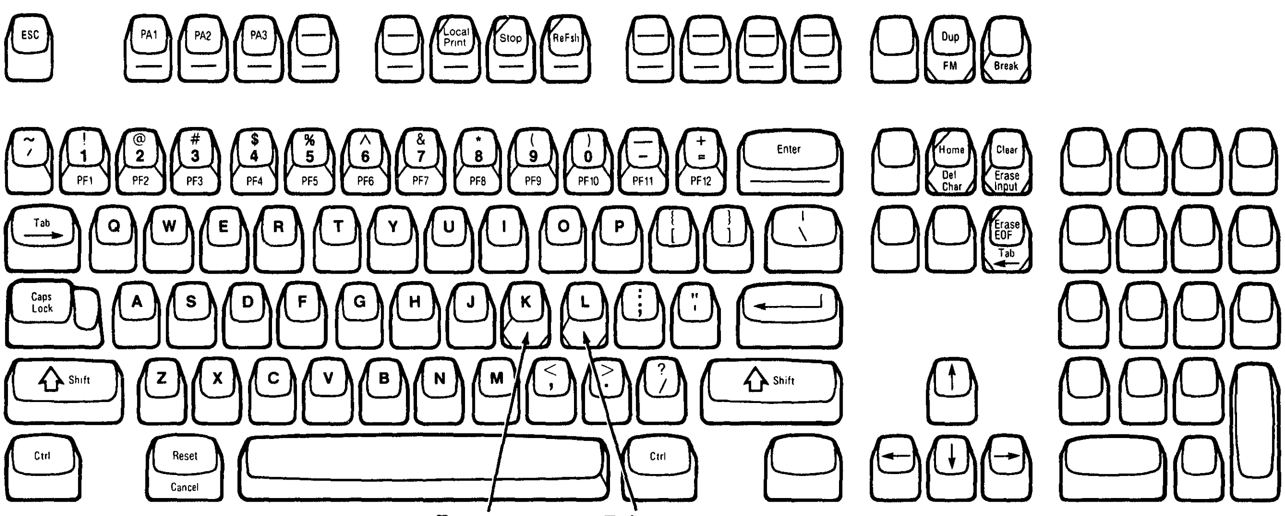


Figure 25. 3101 Keyboard

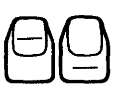


Legend:

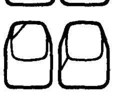
Typamatic



Legend:



Ignored



Local function

Press ESC before pressing this key to activate the function shown on the front of the key.

Press and hold the Ctrl to activate the function shown on the front of this key.

Figure 26. 3161/3163/3164 Keyboard

3101/3161 Key Functions

The following keys correspond to the keys on the 3277 display station keyboard (see the *3271 Guide*):

- Cursor movement keys

  The following keys move the cursor. They don't affect MDT bits.

  - Cursor up, down, left, right (#58/88, 59/98, 79/97, 80/99)
  - Tab key (#24/38)
  - Backtab key (#23/54)
  - New line key (#78/71)
  - Home key (#37/32)

- ERASE EOF key (#2/54)

  When the cursor is located within an unprotected field, the ERASE EOF key clears character locations from the cursor location to the end of the field to nulls. It also sets the MDT bit to 1. The cursor does not move. When the cursor is located at an attribute character or within a protected field, the ERASE EOF key disables the keyboard. No character locations are cleared; the cursor does not move; and the MDT bit is not set.

- ERASE INPUT key (#2/33)

  This key clears all unprotected character locations to nulls, resets MDT bits to 0 in unprotected fields, and repositions the cursor to the first unprotected character location on the screen.

- INS (Insert) MODE

  Two keys on a 3101 keyboard provide this function:

  - ENTER INSERT CHARACTER MODE key (#53/67)

    This key puts the keyboard controls in insert mode. When the terminal operator presses this key, the cursor blinks and the alarm beeps. When the cursor is located within an unprotected field that contains null characters at or beyond the cursor location, pressing an alphanumeric key causes a character to be entered at the cursor location and sets the MDT bit to 1. The character that was at the cursor location and all remaining non-null characters in the field are shifted right one location. After all null characters at or beyond the cursor location in the field are overwritten (or if there were no null characters), pressing an alphanumeric key causes the alarm to beep.

  - EXIT INSERT CHARACTER MODE key (#54/68)

    This key returns the keyboard controls to normal mode. When the terminal operator presses this key, the cursor blinks and the alarm beeps.

- DEL CHAR key (#17/32)

  When the cursor is located within an unprotected field, the DEL key deletes the character from the location occupied by the cursor and sets the MDT bit to 1. The cursor does not move.

  When the cursor is located at an attribute character or is within a protected field, the DEL key disables the keyboard. No character locations are cleared; the cursor does not move; and the MDT bit is not set.

- RESET Keyboard

  Two keys on a 3101 keyboard provide this function:

  - BREAK key (#18/16)

    This key resets a disabled keyboard resulting from a condition that the IOCP detected. 'SYSTEM COMMAND' on line 25 of the screen indicates these conditions.

  - RESET key (#84/93)

    This key resets a disabled keyboard resulting from a condition that the hardware detected. Messages such as 'LOCK-SYSTEM COMMAND' and 'LOCK-REKEY' on line 25 of the screen indicate these conditions.

- FM (field mark) key (#35 or 43/15)

  This key enters a unique character code into the device buffer and sets the MDT bit to 1. The field mark character is X'1E'; it displays as a semicolon (;).

  The field mark function is provided by two keys in case one of the keys is used for another purpose, such as a national character.

- DUP key (#35 or 43/15)

  This key enters a unique character code into the device buffer, performs a tab key operation, and sets the MDT bit to 1. The DUP character is X'1C'; it displays as an asterisk (*).

  The DUP function is provided by two keys with $.IO06F0 in case one of the keys is used for another purpose, such as a national character.

- Program attention keys

  The following keys terminate an input operation and disable the keyboard. An attention identification (AID) character that identifies the key is generated, but the MDT bit is not affected. The AID is included in the 3270 data stream that results from the input operation.

  - CLEAR key (#23/33)

    The CLEAR key clears the device buffer to nulls, positions the cursor at location 0, and resets all MDT bits.

  - ENTER key (#16/30)

- Program access keys 1, 2, 3 (#60/2, 61/3, 62/4)
- Program function keys 1-12 (#4/18, 5/19, 6/20, 7/21, 8/22, 9/23, 10/24, 11/25, 12/26, 13/27, 14/28, 15/29)

3101/3161/3163/3164 Key Functions

The following keys provide functions that are not available on a 3277 keyboard.

- Program access key 6 (#42/7)

  This key copies the contents of the screen to the EDX device defined as the 3101 station's hardcopy device. If the station that represents the 3101 is in non-display mode, lowercase characters are translated to uppercase for printing. If the station is in text mode, lowercase characters are not translated.

- STOP key (#19/8)

  This key stops the 3101 as a Communications Facility terminal. If the 3101 is defined to EDX, it becomes an EDX-only terminal; otherwise it is unavailable. You should use this key to disconnect a 3101 terminal connected over a switched line.

- REFSH (refresh) key (#20/9)

  This key displays the contents of the device buffer.

3101/3277 Keyboard Differences

Because of the physical differences between 3277 and 3101/3161/3163/3164 keyboards, two of the characters of the 3277 keyboard use different keys on the 3101/3161/3163/3164 keyboards. These characters are:

3277	3101/3161/3163/3164
¢	∧ (key #9/23)
¬	\ (key #36/51)

The character shown on the face of the key displays, but the IOCP enters the equivalent 3277 character into the device buffer.

Uppercase/Lowercase Support

Two character sets are supported for the 3101: uppercase (mono) and uppercase/lowercase (dual). The setting of the eighth setup switch (the mono/dual switch) controls character set; up is dual, and down is mono.

When the keyboard is set to dual mode, key #3, key #57, and key #36 (upshift) are active. Using these keys in mono mode causes 'LOCK-MODE/SETUP CHECK' to display on line 25 of the screen. Press the RESET key (#84) to reset the keyboard.

Overrun Considerations

Since the 3101 is unbuffered, overrun situations can occur. An overrun is most likely to occur on 9600 bps lines when several operators simultaneously press typamatic cursor movement keys, causing two characters of data to be sent down the line for each keystroke. If an overrun does occur, the IOCP sends back an audible beep to the overrun device, and ignores the character in error. The keyboard may lock if characters are still being entered when the overrun is detected, causing 'LOCK-SYSTEM COMMAND' to display on line 25 of the screen. Press the RESET key (#84) to reset the keyboard. If 'SYSTEM COMMAND' persists after a reset, then press the BREAK key (#18). You can then rekey the missed character, or, in the case of cursor movement, continue to hold down the typamatic key.

Changing Keyboard Functions

You can change the functions of 3101 keys. To do so, you need to change the input interrupt handler module (S$IO06F3) and, in some cases, the main module of the IOCP (S$IO06F0).

The input interrupt handler processes each input character according to the definition in table BRINDEX in module S$IO06FT. For $.IO06F0, the BRINDEX table is in module S$IO06FT. This table defines an index to branch table BRTABLE for the 128 values of the ASCII character set. There are four general groups of characters: numeric characters, alphabetic and other graphic characters, ignored control characters, and supported control characters. The input interrupt handler processes some of the supported control characters, and the main module processes others. The main module refers to characters by names equated to hex values. If you define different characters for functions processed in the main module, you'll need to change the EQU statements, which are located just before the executable code.

These are the supported control functions:

ASCII Hex	Function	3101 Key
00	Reset keyboard	#18
07	Alarm	
08	Enter	#16
09	Tab	#24
0B	Enter insert mode	ALT/#53
0C	Exit insert mode	ALT/#54
0D	New line	#78
1B	Escape	#22
5B	Field mark	#35 (downshift)
5D	Dup	#35 (upshift)
7F	Delete	#17

The main module processes the reset keyboard and enter functions. The input interrupt handler processes the other functions.

When the input interrupt handler receives an escape character, it proceeds according to the next character it receives. The main module processes these escape sequences:

ASCII Hex	Function	3101 Key
1B31-1B39	PF1-PF9	ESC, #4-ESC, #12
1B30	PF10	ESC, #13
1B2D	PF11	ESC, #14
1B3D	PF12	ESC, #15
1B4B	Erase input	ALT/#2
1B4C	Clear	ALT/#23
1B61	PA1	ALT/#60
1B62	PA2	ALT/#61
1B63	PA3	ALT/#62
1B66	Hardcopy	ALT/#42
1B67	Stop	ALT/#19
1B68	Refresh	ALT/#20

Note that PF1-PF9 must be contiguous values.

The input interrupt handler processes these escape sequences at label ESCOK:

ASCII Hex	Function	3101 Key
1B41	Cursor up	#58
1B42	Cursor down	#59
1B43	Cursor right	#80
1B44	Cursor left	#79
1B48	Home	ALT/#37
1B49	Erase EOF	#2
1B4A	Back tab	#23
1B55	Dup	ALT/#43
1B56	Field mark	#43

Any other escape sequence causes the alarm to beep.

Output to Terminals

The IOCP receives messages sent to device stations and interprets them as 3270/BSC data streams. It processes messages as a 3271 control unit does, according to the 3270 command and orders in the data stream. It supports the same commands as a 3271, except for the copy command. It discards any message that contains a copy command.

If a message doesn't begin with STX (X'02'), the IOCP processes the message as a write command with keyboard restore specified in the write control character.

High-Intensity Fields

A 3101 operating in character mode doesn't provide high-intensity display. You might want to highlight high-intensity fields by displaying a special character instead of a blank at the attribute character location. To do so, you'll need to change the EBCDIC-to-ASCII conversion table in the output interrupt handler module. For $.IO06F0, the table is EBCD2ASC in module S$IO06FT.

By the time an attribute character is ready for display, bits 0, 1, and 6 are zero. A value of X'10' for bits 4 and 5 defines high intensity. Therefore, the attribute character for a high-intensity field has a hex value of 08, 09, 18, 19, 28, 29, 38, or 39. To display these attributes as other than blanks, define the ASCII value of the character you want displayed in those positions of the table.

For example, to change EBCD2ASC to display a caret (∧):

```
ORG EBCD2ASC+X'08'
DC  X'5E5E'
ORG EBCD2ASC+X'18'
DC  X'5E5E'
ORG EBCD2ASC+X'28'
DC  X'5E5E'
ORG EBCD2ASC+X'38'
DC  X'5E5E'
```

Changing ASCII/EBCDIC Translation Tables

You can change the tables that the IOCP uses for translating terminal input from ASCII to EBCDIC and terminal output from EBCDIC to ASCII. You may want to do this, for example, to support your national character set.

There are three tables:

- The EBCDIC to ASCII translation table
- The ASCII to EBCDIC translation table
- A branch index table.

The branch index table defines each ASCII value as an alphabetic character, a numeric character, a supported control character, or an ignored value. The table names and the modules in which they reside are:

Table	$.IO06F0 Name	Module
EBCDIC to ASCII	EBCD2ASC	S$IO06FT
ASCII to EBCDIC	ASC2EBCD	S$IO06FT
Branch Index	BRINDEX	S$IO06FT

For example, the codes for the U.S. character double quote (") are EBCDIC X'7F' and ASCII X'22'. Suppose that EBCDIC X'7F' is instead a national character whose ASCII code is X'5C'. To support that character, change the tables as follows:

1. Change the ECBDIC to ASCII table at offset X'7F' to the ASCII code, X'5C'.

2. Change the ASCII to EBCDIC table at offset X'5C' to the EBCDIC code, X'7F'.

3. Change the branch index table at offset X'5C' to X'24' to define the national character as a numeric character or to X'28' to define it as an alphabetic character.

There are special considerations if the ASCII code for a national character is X'5B' or X'5D'. The IOCP associates these values with 3101 key #35 and interprets them as a *field mark* and a *dup* key respectively. When you change the tables to support those values as national characters, you lose the field mark and dup functions.

$.IO06F0 includes definitions that allow you to retain the field mark and dup functions. Field FMEXTRA contains the ASCII code for field mark and is associated with 3101 key #43. Field DUPEXTRA contains the ASCII code for dup and is associated with 3101 key ALT/#43. Both of these fields are in module S$IO06FT.

To support ASCII X'5B' and X'5D' as national characters, change the tables as described above. Then make these changes to module S$IO06FT to redefine the field mark and dup functions:

1. Change field FMEXTRA to any ignored ASCII value, such as X'01'. (See table BRINDEX to determine which values are ignored.)

2. Change field DUPEXTRA to any ignored ASCII value, such as X'02'.

3. Change table ASC2EBCD at offset X'01' to 'FB' (the internal EBCDIC code for the field mark function).

4. Change table ASC2EBCD at offset X'02' to 'FA' (the internal EBCDIC code for the dup function).

Restrictions

Do not cancel $.IO06F0 using the EDX $C command. If you do, the unmapped and mapped storage obtained by the IOCP is not returned to the system. Use CP H $.IO06F0 to terminate the IOCP user station and its device stations.

4978/4980 Data Stream IOCP, $.IO0670

The 4978/4980 data stream IOCP allows you to use 4978 and 4980 terminals as if they were 3277 model 2 terminals. $.IO0678 is now called $.IO0670.

For 4978s, it uses an RPQ control store called the 4978 Data Stream Control Store (D02428), which provides 3270 functions.

For 4980 terminals, the IOCP uses a control store, image store, and microcode random access memory (RAM), which provide the 3270 functions.

You can use the IOCP in these ways:

- For communication between 4978/4980 terminals and host programs to which the Series/1 appears to be a 3270 system. See example 1 in the chapter "Sample Configurations" on page 269.
- For communication between 4978/4980 terminals and Series/1 programs.

Hardware Requirements

For 4978s, this IOCP requires the 4978 keyboard RPQ D02056.

EDX Requirements

Include EDX terminal support modules in the EDX supervisor, and use the TERMINAL statement to define each terminal.

For 4980s, include EDX 4980 power-on random access memory support in the EDX supervisor.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0670, type=user.
- Each 4978 or 4980 terminal. Type=device, subtype=4978.

When you define the device stations, use station names that are different from the EDX terminal names. Link each device station with a station that represents an emulated terminal or an application program.

Data Set Requirements

For 4978s, this IOCP requires data sets $.RPQCS0 and $.RPQIS0 (the data stream RPQ control and image stores), which are distributed with the Communications Facility, and the standard 4978 control and image stores, $4978CS0 and $4978IS0, which are distributed with EDX.

For 4980s, in addition to the data sets required for 4978s, this IOCP requires data sets $4980CSA, $4980ISA, and $4980R0A (control store, image store, and microcode random access memory), which are distributed with the

Communications Facility; and $4980CS0, $4980IS0, and $4980R01, which are distributed with EDX.

IOCP Operation Overview

The 4978/4980 data stream IOCP consists of a main task and a subtask for each started device station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt device stations. Each device subtask processes data received from a 4978/4980 and messages sent to the station that represents it.

When a device station is started, the IOCP enqueues on the terminal and loads the data stream RPQ control and image stores for a 4978 or the control store, image store, and microcode random access memory for a 4980. The IOCP and the data stream support make the 4978 or 4980 look like a 3277 to the operator and to the programs that communicate with the terminal.

When a device station is stopped, the IOCP loads the standard control and image store for a 4978 or the standard control store, image store, and microcode random access memory for a 4980. The IOCP then dequeues the terminal.

You can enter and display both uppercase and lowercase characters. For a 4978, the relationship between the keyboard shift and the uppercase and lowercase alphabets is defined by modes UL (upper/lower), LU (lower/upper), and UU (upper/upper). When the 4978 station is in UL mode, downshift produces uppercase letters and upshift produces lowercase letters. When the station is in LU mode, downshift produces lowercase letters and upshift produces uppercase letters. When the station is in UU mode, either shift produces uppercase letters.

Input to numeric fields is validated when you enter it at the keyboard, as for a 3277. You can use 4978 or 4980 keys to enable and disable numeric field validation. See the appropriate keyboard description for the location of these keys.

Field intensity (high, normal, or non-display) is as specified in the field attribute character.

Figure 27 on page 139 summarizes the 3270 functions and level of support that the 4978/4980 data stream IOCP provides. For information about these functions, see the *3271 Guide.*

Function	Supported Yes	No
Remote Command Codes		
Write (F1)	X	
Erase/write (F5)	X	
Erase/write alternate (7E)	X	
Read buffer (F2)	X	
Read modified (F6)	X	
Copy (F7)		X
Erase all unprotected (6F)	X	
Write Control Character		
Reset MDT	X	
Keyboard restore	X	
Sound alarm	X	
Start printer		X
Define printout format		X
Printer Control Orders	X[7]	
Buffer Control Orders		
Start field (SF)	X	
Set buffer address (SBA)	X	
Insert cursor (IC)	X	
Program tab (PT)	X	
Repeat to address (RA)	X	
Erase unprotected to address (EUA)	X	
Attributes		
Protected/unprotected	X	
Alpha/numeric	X	
Automatic skip	X[8]	
High/normal intensity	X	
Non-display	X	
Selector-pen detectable		X
Modified data tag	X	
Keyboard		
ENTER	X	
PF1-PF12	X	
PA1-PA3	X	
CLEAR	X	
ERASE EOF	X	
ERASE INPUT	X	
DEL	X	
RESET	X	
FM (field mark)	X	
DUP	X	
INS MODE	X	

Figure 27 (Part 1 of 2). 3270 Functions Supported on 4978/4980—Data Stream Mode

---

7 Printer control orders display on a 4979/4980 the same as they do on a 3277.

8 The 4978/4980 always skips over protected data during keyboard entry.

Function	Supported Yes	No
TEST REQ		X
CURSR SEL		X
Cursor movement	X	
Lowercase alpha	X	
Card reader		X
Selector pen		X

Figure 27 (Part 2 of 2). 3270 Functions Supported on 4978/4980—Data Stream Mode

Input from Terminals

Input from the terminal is a 3270/BSC data stream. The data stream is the same as if it had originated from a remote 3277 model 2 terminal in response to a 3270 read modified command. The IOCP sends the data stream as a Communications Facility message, specifying the device station as the origin. If the IOCP has a direct link vector, it specifies that station as the destination; otherwise, it specifies a null destination.

4978 Keyboard

Figure 28 on page 141 shows the 4978 keyboard as supported by $.IO0670. Pressing keys that have a legend of "Local Function" causes the indicated function, but no data enters the system.

4978/3277 Key Functions

The following keys correspond to the keys on a 3277 display station keyboard (see the *3271 Guide*):

- Cursor movement keys

  The following keys move the cursor. They don't affect MDT bits.

  - Cursor down, up, left, right (#146, 147, 168, 169)
  - Tab key (#88 or 170)
  - Backtab key (#103 or 127)
  - New line key (#124)
  - Home key (#126)

- ERASE EOF key (#83)

  When the cursor is located within an unprotected field, the ERASE EOF key clears character locations from the cursor location to the end of the field to

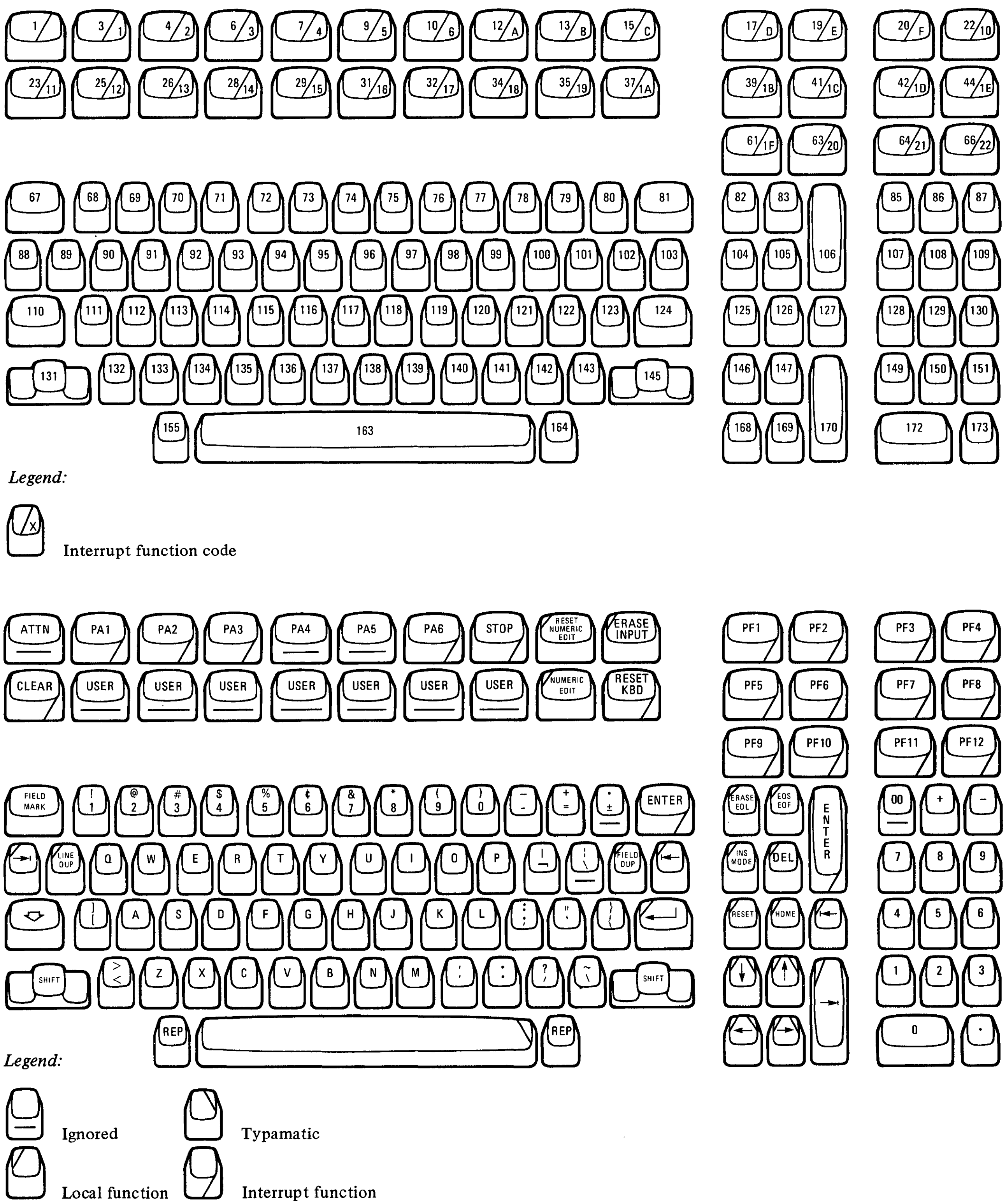


Figure 28. 4978 Keyboard—Data Stream Mode

nulls. It also sets the MDT bit to 1. The cursor does not move. When the cursor is located at an attribute character or within a protected field, the

ERASE EOF key disables the keyboard. No character locations are cleared; the cursor does not move; and the MDT bit is not set.

- ERASE INPUT key (#15)

  This key clears all unprotected character locations to nulls, resets MDT bits to 0 in unprotected fields, and repositions the cursor to the first unprotected character location on the screen.

- INS (Insert) MODE

  Two keys on a 4978 keyboard provide this function:

  - INS MODE key (#104)

    This key puts the keyboard controls in insert mode. When the terminal operator presses this key, the cursor blinks. When the cursor is located within an unprotected field that contains null characters at or beyond the cursor location, pressing an alphanumeric key causes a character to be entered at the cursor location and sets the MDT bit to 1. The character that was at the cursor location and all remaining non-null characters in the field are shifted right one location. After all null characters at or beyond the cursor location in the field are overwritten (or if there were no null characters), pressing an alphanumeric key causes the alarm to beep.

  - RESET key (#125)

    This key returns the keyboard controls to normal mode. When the terminal operator presses this key, the cursor blinks and the alarm beeps.

- DEL key (#105)

  When the cursor is located within an unprotected field, the DEL key deletes the character from the location occupied by the cursor and sets the MDT bit to 1. The cursor does not move.

  When the cursor is located at an attribute character or is within a protected field, the DEL key disables the keyboard. No character locations are cleared; the cursor does not move; and the MDT bit is not set.

- RESET KBD key (#37)

  This key resets a disabled keyboard.

- FM (field mark) key (#67)

  This key enters a unique character code into the device buffer and sets the MDT bit to 1. The field mark character is X'1E'; it displays as a semicolon (;).

- DUP key (#102)

  This key enters a unique character code into the device buffer, performs a tab key operation, and sets the MDT bit to 1. The DUP character is X'1C'; it displays as *.

- Program attention keys

  The following keys terminate an input operation and disable the keyboard. An attention identification (AID) character that identifies the key is generated, but the MDT bit is not affected. The AID is included in the 3270 data stream that results from the input operation:

  - CLEAR key (#23)

    The CLEAR key clears the device buffer to nulls, positions the cursor at location 0, and resets all MDT bits.

  - ENTER key (#81 or 106)
  - Program access keys 1, 2, 3 (#3, 4, 6)
  - Program function keys 1-12 (#17, 19, 20, 22, 39, 41, 42, 44, 61, 63, 64, 66)

4978 Key Functions

The following keys provide functions that are not available on a 3277 keyboard:

- PA6 key (#10)

  This key copies the contents of the screen to the EDX hardcopy device.

- STOP key (#12)

  This key stops the 4978 as a Communications Facility terminal, and releases it for use as an EDX terminal.

- RESET NUMERIC EDIT key (#13)

  This key turns off the numeric edit feature for numeric fields.

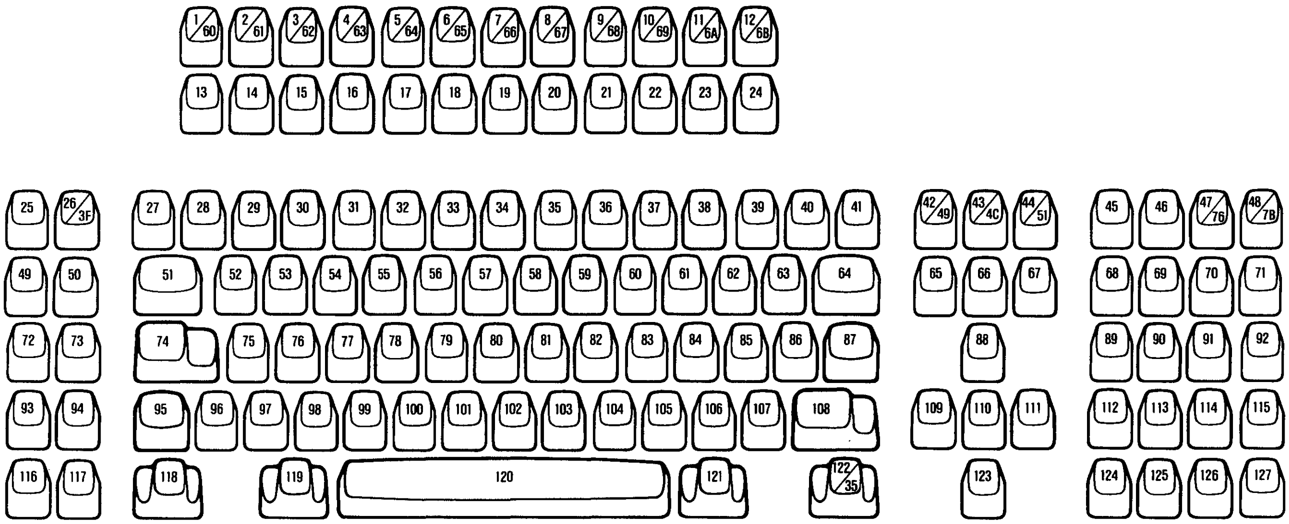
- SET NUMERIC EDIT key (#35)

  This key turns on the numeric edit feature for numeric fields.

4980 Keyboard

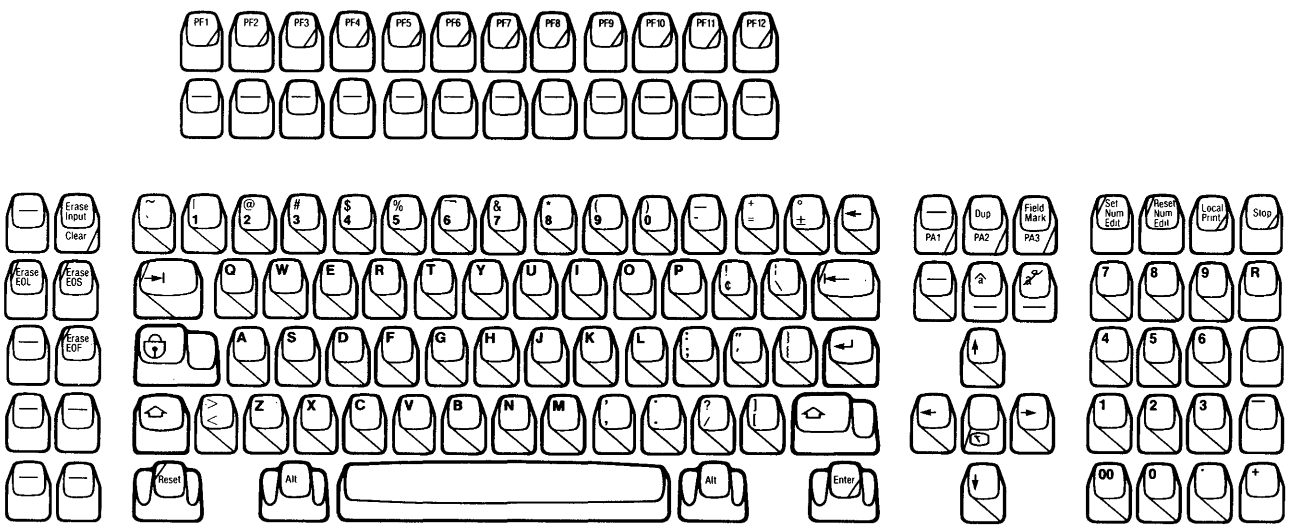
Figure 29 on page 144 shows the 4980 keyboard as supported by $.IO0670. Numbers 1 through 125 on the keys in the top part of the figure have no significance other than for reference numbers. When two numbers are shown on the keyboard, the number beneath the slash (/) is the interrupt function code.

Pressing keys that have a legend of "Local Function" causes the indicated function, but no data enters the system.

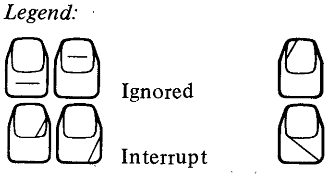


Legend:

Interrupt function code



Legend:



Ignored

Interrupt

Local function

Typamatic

Figure 29. 4980 Keyboard—Data Stream Mode

4980/3277 Key Functions

The following keys correspond to the keys on a 3277 display station keyboard (see the *3271 Guide*):

- Cursor movement keys

  The following keys move the cursor. They don't affect MDT bits.

  - Cursor down, up, left, right (#123, 88, 109, 111)
  - Tab key (#51)
  - Backtab key (#64)
  - New line key (#87)
  - Home key (#110)

- ERASE EOF key (#73)

  When the cursor is located within an unprotected field, the ERASE EOF key clears character locations from the cursor location to the end of the field to nulls. It also sets the MDT bit to 1. The cursor does not move. When the cursor is located at an attribute character or within a protected field, the ERASE EOF key disables the keyboard. No character locations are cleared; the cursor does not move; and the MDT bit is not set.

- ERASE INPUT key (#26)

  This key clears all unprotected character locations to nulls, resets MDT bits to 0 in unprotected fields, and repositions the cursor to the first unprotected character location on the screen.

- INS MODE key (#66)

  This key puts the keyboard controls in insert mode. When the cursor is located within an unprotected field that contains null characters at or beyond the cursor location, pressing an alphanumeric key causes a character to be entered at the cursor location and sets the MDT bit to 1. The character that was at the cursor location and all remaining non-null characters in the field are shifted right one location. After all null characters at or beyond the cursor location in the field are overwritten (or if there were no null characters), pressing an alphanumeric key causes the alarm to beep.

- DEL key (#67)

  When the cursor is located within an unprotected field, the DEL key deletes the character from the location occupied by the cursor and sets the MDT bit to 1. The cursor does not move.

  When the cursor is located at an attribute character or is within a protected field, the DEL key disables the keyboard. No character locations are cleared; the cursor does not move; and the MDT bit is not set.

- RESET KBD key (#118)

  This key returns the terminal to normal mode.

- FM (field mark) key (#44)

This key enters a unique character code into the device buffer and sets the MDT bit to 1. The field mark character is X'1E'; it displays as a ;.

- DUP key (#43)

  This key enters a unique character code into the device buffer, performs a tab key operation, and sets the MDT bit to 1. The DUP character is X'1C'; it displays as *.

- Program attention keys

  The following keys terminate an input operation and disable the keyboard. An attention identification (AID) character that identifies the key is generated, but the MDT bit is not affected. The AID is included in the 3270 data stream that results from the input operation.

  - CLEAR key (#26)

    The CLEAR key clears the device buffer to nulls, positions the cursor at location 0, and resets all MDT bits.

  - ENTER key (#122)
  - Program access keys 1, 2, 3 (#42, 43, 44)
  - Program function keys 1-12 (#1-12)

4980 Key Functions

The following keys provide functions that are not available on a 3277 keyboard:

- Local print key (#47)

  This key copies the contents of the screen to the EDX hardcopy device.

- STOP key (#48)

  This key stops the 4980 as a Communications Facility terminal, and releases it for use as an EDX terminal.

- RESET NUMERIC EDIT key (#46)

  This key turns off the numeric edit feature for numeric fields.

- SET NUMERIC EDIT key (#45)

  This key turns on the numeric edit feature for numeric fields.

- DOUBLE ZERO key (#124)

  This key is ignored by the Communications Facility.

Output to Terminals

The IOCP writes messages destined for a device station to the 4978 or 4980 that the station represents. Each message must begin with a 3270 command or STX/ESC (X'0227') followed by a 3270 command. The IOCP supports the same commands as a 3271 control unit, except for the copy command. It discards messages that contain a copy command.

Printer IOCP, $.IO0674

The printer IOCP allows you to use Series/1 printers as if they were 3286 model 2 printers or 3287 model 2 printers in SNA character string (SCS) mode.

You can use the IOCP in these ways:

- For communication between printers and host programs to which the Series/1 appears to be a 3270 system. See example 1 in the chapter "Sample Configurations" on page 269.
- For communication between printers and Series/1 programs.

Hardware Requirements

This IOCP supports Series/1 printers.

EDX Requirements

Include EDX terminal support in the EDX supervisor. Use the TERMINAL statement to define each printer.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO0674, type=user.
- Each printer. Type=device, subtype=printer.

When you define the device stations, use station names that are different from the EDX terminal names. Link each device station with a station that represents an emulated printer or an application program. If the printer device station is linked to an LU station that represents a type 1 LU, use the CP F MODE command to change the device station's mode to SCS. (3270 is the default.)

Data Set Requirements

This IOCP uses data set $.SYSPRT to support printers as buffered devices.

LU-1 Support

When you link a printer station with an SNA LU station that represents a type 1 LU, the printer station will receive SNA character string (SCS) data. As distributed, this IOCP does not support SCS data. You must link-edit the IOCP to include the support.

To include LU-1 support link-edit object modules O$IO0674, O$IO0675, O$IO0676, and O$IO0677.

IOCP Operation Overview

This IOCP consists of a main task and a subtask for each started device station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt device stations.

Each printer device subtask processes messages sent to the station that represents a printer. The IOCP makes the printer appear functionally as a 3286, or a 3287 in SCS mode to the program that sends data to it. The IOCP shares the printer with EDX. It enqueues on the printer only during a print operation. If the printer is busy when the IOCP needs it, the IOCP waits for the printer to become available.

Figure 30 summarizes the 3270 functions and level of support that the printer IOCP provides. For more information about these functions, see the *3271 Guide*. SCS functions are described under "Page Formatting—SNA Character String Mode" on page 153.

	Supported	
Function	**Yes**	**No**
Remote Command Codes		
Write (F1)	X	
Erase/write (F5)	X	
Erase/write alternate (7E)	X	
Read buffer (F2)	X	
Read modified (F6)	X	
Copy (F7)		X
Erase all unprotected (6F)	X	
Write Control Character		
Reset MDT	X	
Keyboard restore	X	
Sound alarm	X	
Start printer	X	
Define printout format	X	
Printer Control Orders		
New line (NL)	X	
End of message (EM)	X	
Carriage return (CR)	X	
Forms feed (FF)	X	
Buffer Control Orders		
Start field (SF)	X	
Set buffer address (SBA)	X	
Insert cursor (IC)	X	
Program tab (PT)	X	
Repeat to address (RA)	X	
Erase unprotected to address (EUA)	X	

Figure 30 (Part 1 of 2). 3270 Functions Supported on Printer

Function	Supported Yes	No
Card reader		X
Selector pen		X

Figure 30 (Part 2 of 2). 3270 Functions Supported on Printer

Non-Display and Text Modes

If the station that represents a printer is in non-display mode, the IOCP translates lowercase characters to uppercase. If the station is in text mode, the IOCP doesn't translate the data. Refer to the appropriate component book on hardware requirements for lowercase data.

Printer Operation

If the Communications Facility printer IOCP contains the LU-1 support, a printer can operate in 3270 data stream mode or SNA character string mode (SCS). When a printer station is defined, it is in the 3270 data stream mode. You can use the CP F MODE command to set a printer station to either mode. If the IOCP does not contain LU-1 support, it rejects the start of a printer station in SCS mode.

You can determine the printer's mode by using the CP Q PARM command to display the value of Q$DVD in the started printer's station block. Bit 0 in Q$DVD of the printer's station block is on if the printer is in SCS mode and off if the printer is in 3270 data stream mode.

The IOCP receives messages sent to a printer device station and processes them as a 3270 control unit does, according to the 3270 command and orders or SCS control codes in the data stream. If a message doesn't begin with STX/ESC (X'0227'), the IOCP processes the message as an erase/write command with start printer specified in the write control character. If the printer station is in 3270 data stream mode, the write control character specifies an 80-character print line. If the station is in SCS mode, the print operation is unformatted.

Printing doesn't begin until the IOCP receives a data stream that has the start printer bit on in the write control character (bit 4=1). The IOCP uses the PRINTEXT instruction to print data.

Buffered and Unbuffered Support

All 3270 printers are buffered devices. Data sent to a 3270 printer is held in the printer buffer until the control unit receives a data stream that has the start printer bit on.

Series/1 printers are not buffered devices. The IOCP uses data set $.SYSPRT as a printer buffer. When you define a printer device station and specify that the IOCP is to support it as a buffered device, a member with the same name as the station name is allocated in $.SYSPRT. If a printer station has a member in $.SYSPRT, the IOCP uses the member to hold data between print operations.

You can, for example, send a form such as an invoice with an erase/write command and the start printer bit off. The IOCP receives the data and saves it in $.SYSPRT. You can then send variable data, such as name and address, with write commands and the start printer bit on. Each such write command causes the IOCP to update and print the $.SYSPRT member.

If a printer station has no member in $.SYSPRT, the IOCP supports it as an unbuffered device. The IOCP processes each data stream sent to the printer as if it were an erase/write command; no data is retained between print operations. The data is not actually printed unless the write control character in the data stream has the start printer bit on.

To change support for a printer station from buffered to unbuffered, use the EDX utility program $DIUTIL to delete its member from $.SYSPRT. To change from unbuffered to buffered, use $DIUTIL to delete the station definition from $.SYSNET, and then redefine the station as a buffered device.

Page Formatting—3270 Data Stream Mode

A print operation is *formatted* when bits 2 and 3 of the write control character are not 0; the bits define a 40-, 64-, or 80-character print line. A print operation is *unformatted* when these bits are 0; printer control orders in the data stream determine line length.

The IOCP processes printer control orders as a 3270 control unit does, as follows:

- The FF order (forms feed, X'0C') is valid only when it immediately follows the write control character or a CR or NL order. The IOCP executes the FF order, for both formatted and unformatted print operations, by specifying PRINTEXT operand LINE=1. A valid FF order prints as a blank. The IOCP doesn't execute an invalid FF order; it prints as "<" unless it is within a non-display field, in which case it prints as a blank.

- The IOCP doesn't execute an NL order (new line, X'15') for a formatted print operation or if the NL is within a non-display field. It prints as "5" or a blank. Otherwise, the IOCP executes NL by specifying PRINTEXT operand SKIP=1.

- The IOCP doesn't execute a CR order (carriage return, X'0D') for a formatted print operation or if the CR is within a non-display field. It prints as a blank. Otherwise, the IOCP executes CR by specifying PRINTEXT operand SKIP=0.

- The IOCP doesn't execute an EM order (end of message, X'19') for a formatted print operation or if the EM is within a non-display field. It prints as "9" or a blank. Otherwise, EM terminates the print operation.

Because the IOCP uses the PRINTEXT instruction, there can be conflicts between the page formatting specified in a data stream and that performed by EDX. EDX resets its line counter to 0 and starts a new page when:

- It executes a PRINTEXT instruction with LINE=0.

- It processes a $E command.

- It reaches the bottom of a page (as defined by the BOTM parameter of the TERMINAL statement).

A problem arises when both EDX and the program that sends print data realize that they've reached the bottom of a page, and the program then sends multiple NL orders to skip to line 1 of the next page. EDX starts a new page before it receives the PRINTEXT SKIP=1 instructions, which positions the printer beyond the top of the next page. If the program sends an FF order to start a new page, there is no problem; the PRINTEXT LINE=1 instruction causes a skip to line 1.

If the program uses the FF order to start a new page, define the printed page size to EDX in the TERMINAL statement (for example, PAGSIZE=66, TOPM=3, BOTM=63). If the program uses multiple NL orders to start a new page, define the entire physical page size to EDX (for example, PAGSIZE=66, TOPM=0, BOTM=65).

You must also consider how the operator can notify EDX that the forms have been positioned at the top of a page at the start of a print job. (The printer restore or mode switch doesn't do it.) If the printer is the one defined to EDX as $SYSPRTR, the operator can issue a $E command. Otherwise, the operator must run some program that executes a PRINTEXT instruction with LINE=0. The program can be one you provide, or the one that processes the CP SET LOG command. In the latter case, the operator can issue a CP SET LOG command for the printer and then another CP SET LOG command for the actual log device.

Page Formatting—SNA Character String Mode

In SCS mode, print operations are unformatted. Figure 31 lists the SCS control codes that the Communications Facility supports. For more information about these control codes, see the *3287 Printer Component Description.*

Function	Supported: Yes	Supported: No
Back space (BS)	X	
Bell (BEL)		X
Carriage return (CR)	X	
Enable presentation (ENP)		X
End of message (EM)	X[9]	
Form feed (FF)	X	
Graphic escape (GE)		X
Horizontal tab (HT)	X	
Inhibit presentation (INP)		X
Inter-record separator (IRS)	X[9]	
Line feed (LF)	X	
New line (NL)	X	
Set attribute (SA)		X
Set horizontal format (SHF)	X	
Set vertical format (SVF)	X	

Figure 31 (Part 1 of 2). SCS Functions Supported by the Communications Facility

9 Treated as an NL control code.

Function	Supported Yes	No
Set line density (SLD)	X[10]	
Transparent (TRN)	X	
Vertical channel select (VCS)	X	
Vertical tab (VT)	X	

Figure 31 (Part 2 of 2). SCS Functions Supported by the Communications Facility

Nonprintable characters other than SCS control codes, including 3270 display attributes and orders, are replaced with a minus sign (-).

3270 Status Messages

The IOCP sends 3270 status messages to the printer station's direct link to report printer status. It sends a device end status message (X'C240') at the end of a print operation. It sends an intervention required status message (X'4050') if a PRINTEXT instruction fails with return code 5 (device not ready). It sends an equipment check status message (X'40C8') if a PRINTEXT instruction fails for any other reason.

Printer Errors

If you're using a printer for pass-through to a host, the printer station is linked to a station representing an emulated 3286 or 3287 printer. After a successful print operation, the printer IOCP sends a device end status message to the emulated printer station, and the emulation IOCP sends it on to the host. This allows the host to transmit another message to the printer. If an error occurs during the print operation, the IOCP tests periodically to see if the problem has been corrected. When the error is cleared, the IOCP sends a device-end status message to the host and printing continues.

10 Accepts 3, 4, 6, and 8 lines-per-inch.

7485 IOCP, $.IO0680

The 7485 IOCP allows you to use 7485 terminals as if they were 3277 model 2 terminals. $.IO0680 manages communication with 7485 terminals connected to the Series/1 by an asynchronous adapter.

You can use the IOCP in these ways:

- For communication between 7485 terminals and host programs to which the Series/1 appears to be a 3270 system.
- For communication between 7485 terminals and Series/1 programs.

Hardware Requirements

This IOCP supports the 7485 terminal model 53. Use of $.IO0680 requires one of these attachment features:

- Feature programmable communications adapter with feature number 2095/RPQ D02350. Up to eight 7485s can be attached locally with an RS422A electrical interface. When any of the terminals are being used the one at subaddress 0 must be active; the station that represents it must be started first and stopped last. (Subaddress 0 is the device address that is a multiple of 8.)
- Multifunction attachment (feature number 1310). Up to four 7485s can be attached locally with an RS422A electrical interface. The 7485s operate at 9600 bits per second (bps). Be careful not to exceed the aggregate rate of the multifunction attachment. Refer to the *Multifunction Attachment Feature and 4975 Printer Description* for more information. The information given for 3101 terminals applies to 7485 terminals.
- A terminal/host adapter is required for the 4950 or the 5170-495. On the 4950, up to four terminals can be attached locally to the terminal/host adapter using an RS422A electrical interface. One terminal can be attached locally or remotely to the Personal Computer Asynchronous Communications Adapter (feature number 1502074) using an RS232C electrical interface.

  On the 5170-495, up to four terminals can be attached locally to the terminal/host adapter using an RS422A electrical interface. Up to two terminals can be attached locally or remotely to the terminal/host adapter using an RS232C electrical interface.

You must strap the feature programmable communications adapter to match the 7485 configuration, as explained in the EDX *Installation and System Generation Guide*. The information given for 3101 terminals applies to 7485 terminals.

No strapping is required for the multifunction attachment. The EDX supervisor definition statements ADAPTER and CFTERM define the 7485 configuration.

EDX Requirements

7485 terminals are supported as Communications Facility terminals only; you cannot use them as EDX or work session controller terminals.

For 7485s on a multifunction attachment, include EDX multifunction attachment support in the EDX supervisor, and use the ADAPTER statement to define the attachment. Use the Communications Facility CFTERM instruction to define each 7485.

Include EDX unmapped storage support in the EDX supervisor.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO06F0, type=user.
- Each 7485. Type=device, subtype=3101F.

Program Name Requirements

To use the 7485 IOCP, you must rename it $.IO06F0. Either do not install $.IO06F0 or give it a different name.

Supporting Multiple 7485s

The IOCP can manage multiple 7485s concurrently. Although the 7485 operates in interrupt-per-character mode, the instruction path length through the interrupt handling code is generally short enough so that there is minimal interference with the operation of other 7485s and with any batch work. The exact number of devices that can be supported depends on processor speed, available processor storage, and other tasks being executed concurrently.

The amount of available mapped and unmapped storage and the size of $.IO0680's dynamic storage area are the limiting factors in determining the number of 7485s that can be managed by $.IO0680. See "Planning Storage Requirements" on page 213 for a discussion on determining the amount of processor storage and dynamic storage required to support multiple 7485s.

Supporting 7485s and 3101s

If the 3101s are on asynchronous adapters, you must rename both IOCPs. Rename $.IO06F0 to $.IO0630, and then rename $.IO0680 to $.IO06F0. Define all the stations as 3101F devices. Then use the $.CONFIG change command to change the subtype of the stations that represent 3101 terminals to 3101. Changing the subtype does not alter the bit rate and auto-answer definition.

When you start the stations, $.IO06F0 (the renamed $.IO0680) manages the stations defined as 3101F devices (7485 terminals). $.IO0630 (the renamed $.IO06F0) manages the stations defined as 3101 devices (3101 terminals).

IOCP Operation Overview

The 7485 IOCP consists of a main task and a subtask for each started device station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt device stations. Each device subtask processes data received from a 7485 and messages sent to the station that represents that 7485.

The IOCP makes the 7485 appear as a 3277 to the operator and to the programs that communicate with the terminal. Although the 7485 is an unbuffered device, the IOCP makes it appear as a buffered device. Device buffering is done in Series/1 processor storage.

Figure 32 summarizes the 3270 functions and the level of support that the 7485 IOCP provides. For more information about these functions, see the *3271 Guide*.

	Supported	
Function	**Yes**	**No**
Remote Command Codes		
Write (F1)	X	
Erase/write (F5)	X	
Erase/write alternate (7E)	X	
Read buffer (F2)	X	
Read modified (F6)	X	
Copy (F7)		X
Erase all unprotected (6F)	X	
Write Control Character		
Reset MDT	X	
Keyboard restore	X	
Sound alarm	X	
Start printer		X
Define printout format		X
Printer Control Orders	X[11]	
Buffer Control Orders		
Start field (SF)	X	
Set buffer address (SBA)	X	
Insert cursor (IC)	X	
Program tab (PT)	X	
Repeat to address (RA)	X	
Erase unprotected to address (EUA)	X	
Attributes		
Protected/unprotected	X	

Figure 32 (Part 1 of 2). 3270 Functions Supported on 7485

11 Printer control orders display on a 7485 the same as they do on a 3277.

Function	Supported Yes	No
Alpha/numeric	X	
Automatic skip	X	
High/normal intensity	X	
Non-display	X	
Selector-pen detectable		X
Modified data tag	X	
Keyboard		
ENTER	X	
PF1-PF12	X	
PA1-PA3	X	
CLEAR	X	
ERASE EOF	X	
ERASE INPUT	X	
DEL	X	
RESET	X	
FM (field mark)	X	
DUP	X	
INS MODE	X	
TEST REQ		X
CURSR SEL		X
Cursor movement	X	
Lowercase alpha	X	
Card reader		X
Selector pen		X

Figure 32 (Part 2 of 2). 3270 Functions Supported on 7485

Input from Terminals

The IOCP reads input from a 7485 and holds it in a device buffer in processor storage. When the operator presses a program attention key (see the section "7485 Keyboard" on page 159), the IOCP formats the input data as a 3270/BSC data stream. The data stream is the same as if it had originated from a remote 3277 model 2 terminal in response to a 3270 read modified command.

If the station that represents the 7485 is in non-display mode, the IOCP translates lowercase characters to uppercase. If the station is in text mode, the IOCP does not translate the data.

The IOCP sends the data as a Communications Facility message, specifying the device station as the origin. If the IOCP has a direct link vector, it specifies that station as the destination; otherwise, it specifies a null destination.

7485 Keyboard

Figure 33 shows the 7485 keyboard as supported by the 7485 IOCP. A description of the supported functions follows. Pressing keys that have a legend of "Ignored" causes the alarm to beep; nothing is entered into the device buffer. Pressing keys that have a legend of "Local Function" causes the indicated function, but no data enters the system.

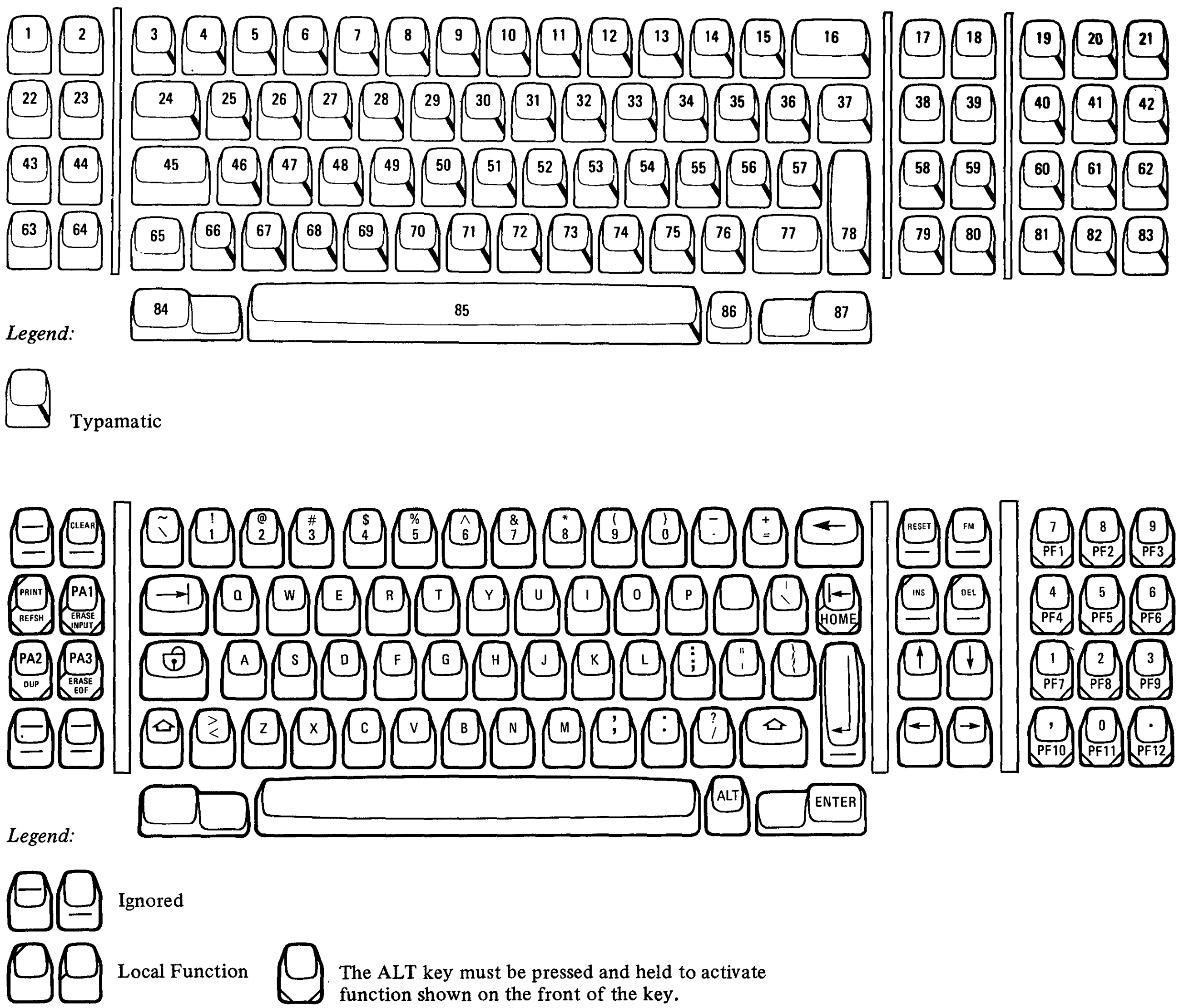


Figure 33. 7485 Keyboard

7485/3277 Key Functions

The following keys correspond to the keys on the 3277 display station keyboard (see the *3271 Guide*):

- Cursor movement keys

  The following keys move the cursor. They don't affect MDT bits.

  - Cursor up, down, left, right (#58, 59, 79, 80)
  - Tab key (#24)
  - Backtab key (#37)
  - New line key (#78)
  - Home key (#37)

- ERASE EOF key (#44)

  When the cursor is located within an unprotected field, the ERASE EOF key clears character locations from the cursor location to the end of the field to nulls. It also sets the MDT bit to 1. The cursor does not move. When the cursor is located at an attribute character or within a protected field, the ERASE EOF key disables the keyboard. No character locations are cleared; the cursor does not move; and the MDT bit is not set.

- ERASE INPUT key (#23)

  This key clears all unprotected character locations to nulls, resets MDT bits to 0 in unprotected fields, and repositions the cursor to the first unprotected character location on the screen.

- INS (Insert) MODE key (#38)

  This key switches the keyboard controls from normal mode to insert mode or from insert mode to normal mode.

  The cursor blinks while the keyboard controls are in insert mode. When the cursor is located within an unprotected field that contains null characters at or beyond the cursor location, pressing an alphanumeric key causes a character to be entered at the cursor location and sets the MDT bit to 1. The character that was at the cursor location and all remaining non-null characters in the field are shifted right one location. After all null characters at or beyond the cursor location in the field are overwritten (or if there were no null characters), pressing an alphanumeric key causes the alarm to beep.

- DEL CHAR key (#39)

  When the cursor is located within an unprotected field, the DEL key deletes the character from the location occupied by the cursor and sets the MDT bit to 1. The cursor does not move.

  When the cursor is located at an attribute character or is within a protected field, the DEL key disables the keyboard. No character locations are cleared; the cursor does not move; and the MDT bit is not set.

- RESET key (#17)

  This key resets a disabled keyboard.

- FM (field mark) key (#18)

  This key enters a unique character code into the device buffer and sets the MDT bit to 1. The field mark character is X'1E'; it displays as a semicolon (;).

- DUP key (#43)

  This key enters a unique character code into the device buffer, performs a tab key operation, and sets the MDT bit to 1. The DUP character is X'1C'; it displays as an asterisk (*).

- Program attention keys

  The following keys terminate an input operation and disable the keyboard. An attention identification (AID) character that identifies the key is generated, but the MDT bit is not affected. The AID is included in the 3270 data stream that results from the input operation.

  - CLEAR key (#2)

    The CLEAR key clears the device buffer to nulls, positions the cursor at location 0, and resets all MDT bits.

  - ENTER key (#87)
  - Program access keys 1, 2, 3 (#23, 43, 44)
  - Program function keys 1-12 (#19, 20, 21, 40, 41, 42, 60, 61, 62, 81, 82, 83)

7485 Key Functions

The following keys provide functions that are not available on a 3277 keyboard.

- PRINT key (#22)

  This key copies the contents of the screen to the EDX device defined as the 7485 station's hardcopy device. If the station that represents the 7485 is in non-display mode, lowercase characters are translated to uppercase for printing. If the station is in text mode, lowercase characters are not translated.

- REFSH (refresh) key (#22)

  This key displays the contents of the device buffer.

7485/3277 Keyboard Differences

Because of the physical differences between 3277 and 7485 keyboards, two of the characters of the 3277 keyboard use different keys on the 7485 keyboard. These characters are:

3277	3101
¢	∧ (key #9)
¬	\ (key #36)

The character shown on the face of the key displays, but the IOCP enters the equivalent 3277 character into the device buffer.

Uppercase/Lowercase Support

Two character sets are supported for the 7485: uppercase (mono) and uppercase/lowercase (dual).

The default character set for the 7485 terminal is uppercase. To change to a dual character set, use the CP F MODE command to change the mode to text for that station.

Overrun Considerations

Since the 7485 is unbuffered, overrun situations can occur. An overrun is most likely to occur when several operators simultaneously press typamatic cursor movement keys, causing two characters of data to be sent down the line for each keystroke. If an overrun does occur, the IOCP sends back an audible beep to the overrun device, and ignores the character in error. The keyboard may lock if characters are still being entered when the overrun is detected, causing 'WAIT-SYSTEM BUSY' to display on line 25 of the screen. Press the RESET key (#17) to reset the keyboard.

Changing Keyboard Functions

You can change the functions of 7485 keys. To do so, you need to change the input interrupt handler module and, in some cases, the main module of the IOCP. The names of these modules are:

IOCP	MAIN	Input Interrupt Handler
$.IO0680	S$IO0680	S$IO0683

The input interrupt handler processes each input character according to the definition in table BRINDEX. This table defines an index to branch table BRTABLE for the 128 values of the ASCII character set. There are four general groups of characters: numeric characters, alphabetic and other graphic characters, ignored control characters, and supported control characters. The input interrupt handler processes some of the supported control characters, and the main module processes others. The main module refers to characters by names equated to hex values. If you define different characters for functions processed in the main module, you'll need to change the EQU statements, which are located just before the executable code.

These are the supported control functions:

ASCII Hex	Function	7485 Key
68	Reset keyboard	#17
07	Alarm	
52	Enter	#87
09	Tab	#24
50	Enter insert mode	#38
50	Exit insert mode	#38
0D	New line	#78
1B	Escape	
49	Field mark	#18
6D	Dup	ALT/#43
51	Delete	#39

The main module processes the reset keyboard and enter functions. The input interrupt handler processes the other functions.

When the input interrupt handler receives an escape character, it proceeds according to the next character it receives. The main module processes these escape sequences:

ASCII Hex	Function	7485 Key
1B71-1B73	PF1-PF3	ALT/#19-ALT/#21
1B74-1B76	PF4-PF6	ALT/#40-ALT/#42
1B77-1B79	PF7-PF9	ALT/#60-ALT/#62
1B7A-1B7C	PF10-PF12	ALT/#81-ALT/#83
1B6C	Erase input	ALT/#23
1B62	CLEAR	#2
1B64	PA1	#23
1B65	PA2	#43
1B66	PA3	#44
1B63	Hardcopy	#22
1B6B	Refresh	ALT/#22

Note that PF1-PF9 must be contiguous values.

The input interrupt handler processes these escape sequences at label ESCOK:

ASCII Hex	Function	3101 Key
1B41	Cursor up	#58
1B42	Cursor down	#59
1B43	Cursor right	#80
1B44	Cursor left	#79
1B48	Home	ALT/#37
1B6E	Erase EOF	ALT/#44
1B32	Back tab	#37

Any other escape sequence causes the alarm to beep.

Output to Terminals

The IOCP receives messages sent to device stations and interprets them as 3270/BSC data streams. It processes messages as a 3271 control unit does, according to the 3270 command and orders in the data stream. It supports the same commands as a 3271, except for the copy command. It discards any message that contains a copy command.

If a message doesn't begin with STX (X'02'), the IOCP processes the message as a write command with keyboard restore specified in the write control character.

Changing ASCII/EBCDIC Translation Tables

You can change the tables that $.IO0680 uses for translating terminal input from ASCII to EBCDIC and terminal output from EBCDIC to ASCII. You may want to do this, for example, to support your national character set.

There are four tables (all in module S$IO068T):

- EBCD2ASC, the EBCDIC to ASCII translation table.
- ASC2EBCD, the ASCII to EBCDIC translation table used when a 3101 station is in text mode (retains lower case input).
- CNVTUPCS, the ASCII to EBCDIC translation table used when a 3101 station is in non-display mode (translates lower case input to upper case).
- BRINDEX, the branch index table that defines each ASCII value as an alphabetic character, a numeric character, a supported control character, or an ignored value.

For example, the codes for the U.S. character double quote (") are EBCDIC '7F' and ASCII '22'. Suppose that EBCDIC '7F' is instead a national character whose ASCII code is '5C'. To support that character, change the tables as follows:

1. Change table EBCD2ASC at offset '7F' to the ASCII code, '5C'.
2. Change table ASC2EBCD and/or CNVTUPCS at offset '5C' to the EBCDIC code, '7F'.
3. Change the branch index table at offset '5C' to '24' to define the national character as a numeric character or to '28' to define it as an alphabetic character.

Restrictions

Do not cancel $.IO0680 using the EDX $C command. If you do, the unmapped and mapped storage obtained by the IOCP is not returned to the system. Use CP H $.IO06F0 (the renamed $.IO0680) to terminate the IOCP user station and its device stations.

PC Connect Disk-Server IOCP, $.IO1CC8

The PC Connect disk-server IOCP allows PC users in a local area network (LAN) to access Series/1 disks as if they were PC disks. A PC can have a maximum of 26 disks, from A to Z. Since the PC considers drives A and B as its local diskettes, the IOCP can support only 24 emulated PC disks for each LAN. If the gateway PC has a hard disk attached, it treats that as a local disk and the number of emulated PC disks is reduced by the number of hard disks attached.

The IOCP recognizes emulated PC data sets on the Series/1 that are consecutively active. It disregards all active data sets after it encounters an inactive data set. For example, if data sets d, e, f, and h are active, the IOCP only recognizes d, e, and f because g is inactive.

This IOCP requires the PC Connect IOCP, $.IO0AA8, and the emulated PC disk management utility program, $.DSFORM.

Hardware Requirements

Use of this IOCP requires the Series/1-PC Connect Attachment (feature number 4000) and a Series/1 disk device, such as a 4967.

EDX Requirements

Include the Communications Facility PC Connect support modules in the EDX supervisor. The required modules are:

- CSPCCINT, the PC Connect initialization routine
- CSPCCIPL, the PC Connect initialization call routine
- CSPCCOPN, the PC Connect microcode data set(s) open routine
- CSPCCPTC, the PC Connect microcode patch load routine.

Include the appropriate disk support for your Series/1 disk device. Use the DISK statement to define each Series/1 disk. For increased performance, define the volumes to be managed by this IOCP as performance volumes on the DISK statement. Include EDX Local Communications Controller support (LCCAM) in the EDX supervisor.

Personal Computer Requirements

Use of this IOCP requires Series/1-PC Connect (5719-CN1). For more information on hardware and software requirements for the PC and the Series/1-PC Connect, see the IBM *Series/1-PC Connect for the Event Driven Executive Communications Facility.*

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO1CC8, type=USER.

- Each volume. Type=VOL.

Link the volume station to the associated PC node station.

IOCP Operation Overview

The PC Connect disk-server IOCP consists of a main task and a subtask for each started volume station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt the volume stations. The subtask processes three I/O requests from the gateway PC—directory, read, and write. The subtask also processes messages sent from $.DSFORM—create, delete, and rename data sets.

The PC Connect IOCP, $.IO0AA8, receives the PC disk I/O requests from the LAN and sends it to the volume station. The PC Connect disk-server IOCP, $.IO1CC8, checks the message queue on the volume station and retrieves the I/O requests to be executed by the IOCP. The IOCP sends the message to the PC node, which is directly linked to the volume station. This is a one-way link: the volume station is linked to the associated PC node station.

Messages to and from the Gateway PC

The following is a list of messages that flow between the volume station and the gateway PC:

- DIRECTORY

  This request is issued when the gateway PC is IPLed. The response provides the Disk Operating System (DOS) with the number of emulated PC disks and their basic input/output support (BIOS) parameter blocks. The IOCP performs this function by reading a control data set, $.DSPROF, that is created by $.DSFORM. It logs all the active emulated PC disks in a directory table when the volume station is started.

- READ

  This request is passed from DOS to the IOCP. DOS requests a read of a number of sectors starting at a certain sector on the PC disk. The PC converts this information into the number of blocks to read starting at a certain block number on the corresponding data set on the Series/1. The IOCP gets this information, issues the read, and sends the result back to the gateway PC.

- WRITE

  This request is passed from DOS to the IOCP. DOS requests a write of a number of sectors starting at a certain sector on a PC disk. The PC converts this information into the number of blocks to write starting at a certain block number on the corresponding data set on the Series/1. The IOCP gets this information, issues the write, and sends the result back to the gateway PC.

Messages from $.DSFORM

There are two kinds of messages that are sent from $.DSFORM when an emulated PC disk is created or deleted. The messages are 10 bytes long and contain the Series/1 data set name representing the emulated PC disk and a flag word indicating whether the emulated PC disk has been created or deleted. When the IOCP receives these messages, it updates the corresponding entry in the directory table and frees the data set control block (DSCB), if it is not needed.

Series/1-to-Host SNA IOCP, $.IO14E8

The Series/1-to-Host SNA IOCP allows you to connect a Series/1 and a host processor in a Systems Network Architecture (SNA) environment. The Series/1 appears to the host as a type 2 PU (a cluster controller) with type 2 LUs (terminals) and types 1 and 3 LUs (printers). There may be up to 32 LUs (emulated terminals and printers) associated with each PU, but the practical maximum is less. If you define too many LUs to EDX-SNA, the program will exceed the EDX partition size. The maximum number of LUs varies according to the size of the Communications Facility PU buffers, and the size of S$POOL.

The IOCP communicates with the EDX Secondary SNA1 network through the EDX Secondary SNA1 Licensed Program (5719-SX1), which is described in the *EDX Secondary SNA1 Guide.*

The IOCP also communicates with the EDX Secondary SNA2 network through the EDX Secondary SNA2 Licensed Program (5719-XX9), which is described in the *EDX Secondary SNA2 Guide.* EDX Secondary SNA2 supports up to four PUs.

Applications running in the host communicate with the Series/1 the same way that they communicate with an actual remote 3270 SNA/SDLC system, as described in the *3274 Guide.*

Although the SNA IOCP operates in an SNA environment, it sends and receives 3270/BSC data stream messages. It removes BSC and 3270 control characters from messages before sending them to the host. During a type 2 or 3 session, it appends BSC and 3270 control characters to messages received from the host before sending them to their destination. During a type 1 session, messages received from the host are sent to their destination unchanged.

You can use the IOCP in these ways:

- For communication between host programs and terminals or printers attached to the Series/1. Example 3 in the chapter "Sample Configurations" on page 269 includes EDX Secondary SNA1 and host system definitions. Example 4 in the chapter "Sample Configurations" on page 269 includes EDX Secondary SNA2 and host system definitions.
- For communication between host programs and Series/1 programs.

Hardware Requirements

The SNA IOCP has no hardware requirements other than those of the Communications Facility and EDX-SNA.

EDX Requirements

You can choose between EDX Secondary SNA1 support and EDX Secondary SNA2 support. The requirements they impose on EDX are described in the *EDX Secondary SNA1 Guide* and in the *EDX Secondary SNA2 Guide.*

EDX-SNA Definitions

You define the EDX-SNA configuration by changing the SNA configuration data set(s) provided, assembling it, and linking it with the EDX-SNA object module. EDX Secondary SNA1 provides only one configuration data set. EDX Secondary SNA2 also provides one configuration data set, but you may duplicate, edit, assemble, and link up to 4 PUs. You must define the configuration before EDX-SNA is started, and you can't change the configuration dynamically.

You can define up to 4 PUs for EDX Secondary SNA2 and from 1 to 32 LUs per PU to the SNA network.

For EDX Secondary SNA2, you must edit the SNAINIT data set to specify whether your PUs will be activated when $SNA is started, and what partitions will be used by the PUs. Specifying partitions to be used by PUs also saves storage.

It is recommended that PUs are not loaded into the partition that contains $.CF. If the PUs are loaded in the same partition that contains $.CF, after $.CF has been loaded, $.CF is unable to load transient programs it requires to process the CP commands. You can avoid this by activating your PUs before starting the Communications Facility, or by using the SNAINIT data set to specify partitions to be used by the PUs. See the *EDX Secondary SNA2 Guide* for more information on defining your PUs.

The parameters for defining each PU are determined mostly by the type of line and modem used. These parameters are described in the *EDX Secondary SNA1 Guide* and *EDX Secondary SNA2 Guide*. The parameters that have an impact on Communications Facility usage are BUFSIZ and DCBNO.

BUFSIZ specifies the size of the buffers to be used. The value specified for BUFSIZ depends on the size of the messages expected to flow through EDX-SNA. The default of 256 is recommended. DCBNO specifies the maximum number of consecutive frames that may be received or sent. DCBNO=7 is recommended; this corresponds to the specifications for the SDLC 3274.

You specify LUs to EDX-SNA by number. You must either specify 0 to allow EDX-SNA to assign the next available logical unit number, or assign LU numbers that begin with 1 and proceed sequentially up to 32. The number assigned to each LU corresponds to the LOCADDR (local address) definition in the NCP that controls the line. The LU number and the PU station are used to map the Communications Facility station definitions to EDX-SNA LUs. (Note that some LUs defined to EDX-SNA may be used by other application programs, and in that case need not be defined to the Communications Facility.)

The *EDX Secondary SNA1 Guide* and *EDX Secondary SNA2 Guide* give complete instructions for defining an EDX-SNA configuration.

Host System Definitions

Define the SDLC line, the PU, and the LUs to the network and to host subsystems (if necessary) as a standard SDLC 3274 with attached terminals. Define the PU as a type 2 (SNA cluster controller).

Define the LUs to host subsystems just as if they represented real devices attached to an SDLC 3274 control unit. Define the LUs that represent terminals as type 2 and the LUs that represent printers as types 1 and 3. Be sure that the LU type is consistent with the corresponding Communications Facility station definitions. LUs that will be connected to terminals must be defined as terminals. LUs that will be connected to printers must be defined as printers. Note that the only screen size supported by the Communications Facility is the 1920-character screen, 80 x 24 characters. The Communications Facility accepts binds for alternate screen sizes if the subtype of the LU is 3287 or 3289.

Station Definitions

Define stations to represent:

- The IOCP. Name=$.IO14E8, type=user.
- For EDX Secondary SNA1: One PU. Type=PU, subtype=3274ES.
- For EDX Secondary SNA2: From 1 to 4 PUs. Type=PU, subtype=3274ES.
- From 1 to 32 LUs for each PU station defined. Type=LU, subtype=3277, 3278, or 3279 for type 2 LUS; subtype=3286 or 3289 for type 3 LUs; and subtype=3287 or 3289 for type 1 LUs. See "Planning Storage Requirements" on page 213 to determine how many LUs your configuration can support.

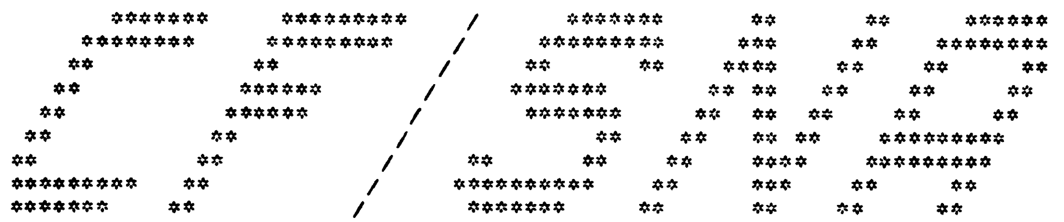
Link each station that represents an LU with a station that represents a real terminal or printer or an application program. For printer stations linked to type 3 LUs, use the CP F MODE command to set the printer station's mode to 3270. For printer stations linked to type 1 LUs, set the mode to SCS. (The default mode is 3270.) No link is required for the PU station.

Logon Parameter Definitions

To initiate communication with the host, the name and mode of an SNA application are required. There are three ways you can supply these parameters:

- You can define up to three sets of logon parameters for the PU station, as explained in the chapter "Defining Stations" on page 45. You can also define a logon ID for an LU station—a number from 1 to 3 that corresponds to one of the sets of logon parameters defined for the PU station. When the LU station is started, the IOCP initiates a session with the specified SNA application. If you defined a user data field for the LU station, the IOCP sends the data to the SNA application. You must use this method if there is no terminal operator involved in a session (for example, if the session is with a printer).
- You can define logon parameters for the PU station, but define no logon ID for an LU station. When the LU station is started, the IOCP sends a prompt screen to the station to which the LU station is linked. The prompt screen, shown in Figure 34 on page 172, shows the logon parameters defined for the PU station. The terminal operator may select one of these and, optionally, enter a password and/or data in the user data field.

  An alternate logon prompt screen, which has separate fields for USERID and PASSWORD, is provided for connections to 8100 Distributed Processing Executive (DPPX) systems. Procedures for installing this alternate panel are described in "Installing the Communications Facility" on page 251.



```
---------------------------------------------------------------------------------

              SELECT LOGON APPLICATION FROM LIST:
                 1:  APPLID  TSO        MODE  LU2256
                 2:  APPLID  CICS       MODE  LU2256
                 3:  APPLID  CICS       MODE  STD32743

              OR, SELECT 4 AND ENTER APPLID AND MODE:
                 4:  APPLID  ________   MODE  ________

      ENTER SELECTION==> _           USER DATA==> ________________

   OR, SELECT 0 AND CHANGE DEFAULT APPLID AND MODE VALUES (1-3 ABOVE)
             CHANGES WILL BE REFLECTED IN ALL FUTURE LOGONS
```

Figure 34. SNA Logon Prompt Screen

- The terminal operator can define the logon parameters. The operator can select option 0 and enter or change any of the three sets of default logon parameters. The information is recorded in the PU station block and in the PU station's definition in $.SYSNET; it applies to all subsequent logons. Alternatively, the operator can select option 4 and enter logon parameters to be used only for this session.

If the session is to be initiated by the host, you define the logon ID as HOST or an ID that has no application associated with it. In either case, when the LU is started, a host-initiated session begins. You need not specify any logon parameters.

IOCP Operation Overview

The SNA IOCP consists of one main task and a subtask for each active LU station. The main task processes commands to stop or halt the IOCP and commands to start, stop, or halt LU stations. Each LU subtask controls the initiation and termination of a session between EDX-SNA and an LU. It processes data received from the host and messages sent to the LU station.

EDX-SNA handles all transmission over the SDLC line that connects the Series/1 and the host. It also manages SNA protocol, pacing, and error recovery.

Note that there is no task associated with the PU station. The PU station block contains information for the associated LU stations—buffer size, PU number (for EDX Secondary SNA2), and logon parameters.

Session Initiation

The IOCP uses one of two logon methods to establish communication with an SNA application. If you specified a logon ID for the LU station when you defined it, session initiation begins depending on which mode the LU station is in. If the mode is NOPC, it begins when the LU station is started. If the mode is PC, session initiation begins when a device end is received. If you didn't specify a logon ID or if you specified 0, the IOCP issues a prompt for an application when the station is started. The operator at the terminal to which the LU station is linked enters logon information, as explained in the section "Logon Parameter Definitions" on page 171. The IOCP builds an EDX-SNA logon parameter list and sends it to EDX-SNA.

EDX-SNA attempts to establish a session by sending an INIT SELF request to the host. The host application either rejects the request or responds by returning a BIND request. If EDX-SNA finds the BIND parameters defining the session to be satisfactory, it accepts the BIND and completes session establishment before returning to the IOCP. The IOCP further checks the BIND session parameters and ends the session if they are unsatisfactory. (BIND session parameters that are acceptable for LU-1, LU-2, and LU-3 are described in the *3274 Guide.*)

Session States

After a session with the host application has been established, the IOCP processes session traffic by passing through several states corresponding to the states of the session. These correspond to processing states of a 3274 under the same conditions.

The states defined are:

- *Between brackets.* (A transaction is called a *bracket* in SNA terminology.) No transaction is currently active.

  The IOCP waits for input from either the Communications Facility or EDX-SNA. If both send messages simultaneously, input from the Communications Facility takes precedence. If a message transmitted during this state does not include an end bracket (EB) indication, the state is changed. If there is no EB, the next state is determined by the next system to send. If a message flows from the Communications Facility to SNA, the IOCP always gives SNA the right to send and the IOCP enters the RECV state. If a message flows from SNA to the Communications Facility, SNA may or may not give the IOCP the right to send. If the IOCP gets the right to send, the IOCP enters the SEND state; otherwise it enters the RECV state. If the device being used is a printer, the IOCP enters the printer busy state.

- *SEND.* The next message should be from the Communications Facility to EDX-SNA, but the host may request the right to send. The IOCP waits for a message to be placed on the LU station's queue. If the appropriate host command is received, the state may change to RECV to allow the host to transmit. If a message is sent to the LU station, the IOCP sends it on to EDX-SNA, and changes the LU state to RECV.

  LU-2 protocol does not allow the secondary LU (the IOCP) to end a bracket, so the state will never change from SEND to between brackets as a result of an inbound, or Series/1-to-host, message. Printer devices supported by LU-1 and LU-3 protocol never enter the SEND state.

- *RECV.* The next message should be from EDX-SNA to the Communications Facility. The IOCP waits for messages or control information from EDX-SNA. If a message arrives, the IOCP sends it as a Communications Facility message, specifying the LU station as the origin. If the IOCP has a direct link vector, it specifies that station as the destination; otherwise, it specifies a null destination. If an EB is received, the state is changed to between brackets. If EB is not received, the state may still be changed. If the device being used is a printer, the state is changed to printer busy pending completion by the printer. Otherwise, if SNA gives the IOCP the right to send, the state changes to SEND.

- *ERP1.* A status indication has been sent to the LU station, indicating that the device to which it is linked is unavailable and that the host application has been notified. The LU will wait for an indication from the Communications Facility that the device is available. When this occurs, the host application is notified, and the LU goes into the RECV state waiting for a message from the host.

- *Printer busy.* The IOCP has received a message for a printer. It has sent the message to its destination and is waiting for a device end status message. If a message is received from SNA, it is rejected with a device busy sense value (X'0814'). When a device end status message is received from the Communications Facility, a positive response is sent, signaling that the printer is ready. If the EDX-SNA message contains an EB, the state is changed to between brackets; if not, the state is RECV.

Session Termination

A session ends when the host application sends an UNBIND request. This may be the result of a particular message to the application, such as a LOGOFF, or the IOCP finding that the LU station has been stopped. In the latter case, the IOCP issues an EDX-SNA NETTERM, causing a RSHUTD (request shutdown) request to be sent to the host.

If the session was established automatically when the station was started (a logon ID was defined to the LU station), the station is in either UBSTOP mode or UBRETRY mode. If UBSTOP, the station is stopped. If UBRETRY, the IOCP tries to rebind the session.

Normally, when the session is ended, the station is stopped. To establish another session, you must start the station again. However, you may use the CP F MODE command to set the UBRETRY mode of operation. When the mode is UBRETRY, you must stop the LU station to end the session.

Messages from the Host

Each message received from the host must contain a 3270 command that is supported for the type of LU to which the message is sent.

3270 Commands

These commands are acceptable for an LU station whose subtype is 3277:

Erase all unprotected
Read modified
Read buffer
Write
Erase write
Erase write alternate
Read modified all (processed as read modified)

These commands are accepted for an LU station whose subtype is 3286 or 3287:

Erase all unprotected
Write
Erase write
Erase write alternate

Any command is accepted for an LU station whose subtype is 3278 or 3279. The IOCP assumes that the destination of the message is a device attached to a real 3274 control unit, and that the control unit will perform error checking.

Basic Mode Input

During a type 2 or type 3 session, if the LU station is in basic mode, the IOCP reformats the message so that it looks like those sent by the 3270 emulation IOCP. It appends STX/ESC (X'0227') to the beginning of the message, and ETX (X'03') to the end of the message before sending the message to its destination. During a type 1 session, messages are sent to their destination unchanged.

Record Mode Input

During a type 2 or type 3 session, if the LU station is in record mode, the IOCP removes the following control information before sending the message to its destination:

- The first 2 bytes (command code and write control character)
- The third through fifth bytes, if the third byte is SBA (set buffer address)
- The sixth and seventh bytes, if the third byte is SBA and the sixth byte is SF (start field).

During a type 1 session, messages are sent to their destination unchanged.

Printer Operation

When the IOCP receives data for an emulated printer station (LU type 3286, 3287, or 3289), with the start printer bit on in the write control character, it sends the message to its destination and sets the LU state to printer busy. It sends a positive response to the host, which indicates that the printer is again ready, when a 3270/BSC device end status message is sent to the LU station. The host will not send more data for the printer until it receives the positive response. (If it does, the IOCP rejects the data with a device busy sense value.)

The program that manages the real printer to which the emulated printer is linked is responsible for sending a device end status message to the emulated printer station when the print operation is complete.

The program that manages the real printer is also responsible for retrying a print operation when the printer is not ready. The IOCP assumes that a print operation will complete successfully once it has sent the data to its destination. If the program that manages the real printer sends back an intervention-required status message, the IOCP ignores it.

The IOCP that manages printers retries print operations, when necessary, and sends the required device end status message. If you write an IOCP to manage printers or an application program that processes data for emulated printers, you must do the same. The format of a status message is:

- A 4-byte heading: SOH (X'01')/%/R/STX (X'02').
- 2 bytes for the control unit and device address, which the 3270 emulation IOCP fills in.
- The 2-byte status/sense code; X'C240' for device end.
- ETX (X'03').

Messages to the Host

When the IOCP receives messages destined for LU stations, it sends them on to EDX-SNA. It modifies messages before sending them, according to whether they contain basic mode data or record mode data.

Basic Mode Output

If a message begins with DLE (X'10') or STX (X'02'), the IOCP assumes that the message is a correctly formatted 3270 data stream. It removes the DLE (if any), the STX, and the next 2 bytes (the 3270 polling address). The IOCP removes the last byte of the message if it is an ETB (X'26') or ETX (X'03').

Record Mode Output

If a message doesn't begin with DLE or STX, the IOCP appends 3 bytes to the beginning of the message:

X'7D'	ENTER key attention ID
X'4040'	Cursor location (row 1, column 1)

Remove and Non-Remove Mode

The IOCP operates in remove mode (except for messages from the program dispatcher), unless you specify otherwise. As a result, if a message is sent to an LU station when the host has the right to send, the IOCP discards the message. You can run the IOCP in non-remove mode. If you do, the host program must be aware of it, because data sent to the host may not be a response to the last message the host sent to the Series/1. The IOCP sets itself to remove mode when it is started. If you want to set non-remove mode, do so *after* the IOCP is started.

Messages from the program dispatcher are handled differently, because the program dispatcher is a transaction processor and cannot synchronize its session. When an LU station is linked to a station whose name is $.PD or begins with the characters $.PD, it always operates in non-remove mode. If a message is received when the Series/1 does not have the right to send, the message is left on its queue, and the host is signalled that the Series/1 wishes to send. When the host gives the right to send, the IOCP sends the message.

Certain levels of the Communications Facility/Host (Program Number 5668-979) documentation state that you must include a CP Modify command in your $.SYSPD data set to set $.IO14E8 to non-remove mode. This is no longer necessary.

Error Reporting

The IOCP reports error conditions to the SNA host application, using either SNA negative responses with sense bytes or LUSTAT commands with sense codes. The *3274 Guide* contains possible sense codes for an LU-2 session implemented on a 3274. The IOCP uses a subset of these codes. Some are not discernible to the IOCP because of the distance to the end user. Others do not occur because the Communications Facility queues messages. EDX-SNA detects and handles many of the error conditions.

Figure 35 lists the sense codes that the IOCP returns as a result of 3270 status messages sent to an LU station, or other causes as described.

SNA Sense	BSC Status	Description
1002		RU too long. Returned if SNA message exceeds PU station buffer size value.
1003	4060, 40C1	Function not supported. Also returned if 3270 command validation fails in IOCP.
0802	C250	Intervention required.
081C	C6D8, C4C4, 40C4	Request not executable—not possible to recover from error.
0829		*Change direction* required. Returned if a message with a 3270 read-type command has *end bracket* or does not have *change direction.*
082A		Presentation space altered; command was executed. Returned if the Communications Facility terminal or program sent a message while LU was in RECV state, and then the host sent a write-type command.
082B	C6C4	Device available, presentation space integrity lost.

Figure 35 (Part 1 of 2). Sense Codes Returned by SNA IOCP

SNA Sense	BSC Status	Description
0831	*xx1x, xxx4*	LU component disconnected. Device/terminal power off, cable disconnected.
0843		Required function manager synchronization not supplied. RUs containing start print bit must require definite response, or exception response RU must carry *change direction.*
084A		Presentation space altered; command was not executed. Returned if the Communications Facility terminal or program sent a message while LU was in RECV state, and then the host sent a read-type command.

Figure 35 (Part 2 of 2). Sense Codes Returned by SNA IOCP

Restrictions

FM headers are not supported. The IOCP rejects a BIND that allows FM headers.

The IOCP does not support the 3270 copy command.

The IOCP accepts the 3270 read modified all command, but processes it as a read modified command.

The IOCP does not support the sending of multiblock messages to the host. It handles each message sent to an LU station as a single complete message.

The IOCP does not provide sufficient protocol to allow implementation of request definite response (RQD) support in SNA. SNA messages with RQD are processed, but in some cases positive responses are returned even though error conditions are encountered within the Communications Facility. Avoid RQD wherever possible when using the SNA IOCP.

The IOCP does not provide any message mapping services except as described for record mode output. If the data going to the host is to be processed by any host mapping services, the appropriate 3270 orders must appear in the data stream. Messages created by real 3270 devices or by 4978/4980/3101/7485 devices as supported by the Communications Facility are formatted correctly. If a user program creates messages, it must format those messages correctly.

The IOCP accepts the BIND for 3286 and 3287 LU3 stations, 3278 and 3279 LU2 devices, and LU1 devices. It assumes that the LU is associated with a real device that can handle alternate screen sizes. The Communications Facility only supports a 24 by 80 screen size and a 1920-byte printer buffer size for emulated devices.

See the *EDX Secondary SNA1 Guide* and the *EDX Secondary SNA2 Guide* for restrictions imposed by EDX-SNA.

Communications Facility Data Sets

The Communications Facility, as you receive it, includes data sets used to hold information the Communications Facility needs for its operation. Some of the data sets are required for all Communications Facility configurations; others are optional. During the installation process, you copy these data sets to your disk. Then you can use various methods (explained in the description of each data set) to add information to the data sets.

This chapter explains how to enlarge the supplied partitioned data sets. Then it describes the supplied data sets, plus EDX data sets and PC Connect microcode patch data sets that the Communications Facility uses.

Enlarging Partitioned Data Sets

If your configuration is large, the data sets as supplied may not be big enough to hold all the information you need to store in them. To make a partitioned data set bigger, use the EDX utility $DISKUT1 to allocate space for a data set of the size you require. Then use the EDX utility $DIUTIL to initialize it as a partitioned data set and to copy members from the old data set into the new one.

Be sure you allocate records for the directory as well as for the data set's members. Each record in the directory of a partitioned data set, except the first, can contain 16 directory entries; the first can contain 15. For example, if you need space for 40 members of five records each, you should allocate 203 records: 200 for members and three for the directory.

See the *Event Driven Executive Operator Commands and Utilities Reference* for information about $DISKUT1 and $DIUTIL.

Network Configuration Data Set ($.SYSNET)

The network configuration data set ($.SYSNET) is the data set that contains the definitions of all the stations in a node and of remote stations that will be communicated with from that node. $.SYSNET is a required partitioned data set; the Communications Facility can't be started without it. As supplied by IBM, it is an empty data set with space for 47 members.

Before you can start a station, there must be a definition of the station in $.SYSNET. You may enter and modify station definitions using the $.CONFIG utility or the command processor commands.

Each station definition is a one-record member in $.SYSNET; the name of the member is the same as the name of the station.

Once you have defined a station in $.SYSNET, you can delete that definition only through the delete function of the EDX utility $DIUTIL. Deleting a member only marks it as deleted; it doesn't free the space the member occupies. Therefore, it is recommended that you compress $.SYSNET after deleting station definitions from it. Note that the compress function must *not* be performed while the Communications Facility is in operation.

$.SYSNET must reside on the same disk or diskette volume as the control program ($.CFS or $.CFD) unless the control program resides on a disk device with a fixed-head volume. In that case, $.SYSNET must reside either on the fixed-head volume or on the same disk volume as the control program.

When a station that has a disk queue is active, the Communications Facility maintains control information in the station's $.SYSNET member. Therefore, queuing of messages to disk is most efficient if $.SYSNET resides on a fixed-head volume.

Message Data Set ($.SYSMSG)

The message data set ($.SYSMSG) is the data set that contains the text of error messages and informational messages issued by the Communications Facility and, optionally, by user programs.

$.SYSMSG is a required partitioned data set; the Communications Facility can't be started without it. It need not, however, contain any members. If you need to conserve disk storage, you can replace the $.SYSMSG data set supplied by IBM with a small empty partitioned data set. If you do, log messages will contain identifying and variable information but no fixed text.

$.SYSMSG must reside on the same volume as the Communications Facility control program.

Each member of $.SYSMSG contains the messages for a particular component of the Communications Facility. The name of the member is the same as the 2-character code at the beginning of each message that identifies the component. Each member contains 99 messages (not all of which need be used).

As supplied by IBM, $.SYSMSG contains nine members:

CA Channel attach IOCP messages
CF Control program messages
CP Command processor messages
IO I/O control program messages
I1 I/O control program messages
I2 I/O control program messages
PD Program dispatcher messages
PN 3270 panel design aid messages
SN SNA IOCP messages

You can create your own error messages and alter existing messages by using the $.CONFIG utility. Each new member that you create requires 25 records. As supplied, $.SYSMSG is large enough so that you can create one member before you need to enlarge the data set.

Initialization Data Set ($.SYSIPL)

The initialization data set ($.SYSIPL) contains commands that are executed each time the Communications Facility is started.

$.SYSIPL is an optional sequential data set. If it exists, it must reside on the same volume as the Communications Facility control program.

As supplied by IBM, $.SYSIPL contains ten records with a /* statement. You can use any EDX text editor to enter commands into $.SYSIPL. One record is required for every two statements. The last command must be followed by a /* statement to indicate the end of the list of commands. See the *Operator's Guide* for an example of a $.SYSIPL data set.

Printer Data Set ($.SYSPRT)

The printer data set ($.SYSPRT) is a partitioned data set that contains members used to support printers as buffered devices.

$.SYSPRT is required if you use the printer IOCP, $.IO0674. It is also required when you define a station with either the CP DEF command or the $.CONFIG utility program. It must reside on the same volume as the program that uses it—the I/O control program, the define command processor, or $.CONFIG.

$.SYSPRT contains three types of members:

- Image store: Eight records used to load the 4978 image store.
- Control store: 16 records used to load the 4978 control store.
- Printer station member: An 8-record member for each printer station to be supported as a buffered device. The member name is the same as the station name. If a particular printer doesn't have a member in this data set, it is supported as an unbuffered device. See the description of the printer IOCP, $.IO0674, in the chapter "Using the Supplied Input/Output Control Programs" on page 71, for more information.

The $.SYSPRT data set as supplied by IBM contains space for three printer station members. One of these members is allocated, if you so request, when you define a printer station.

If you don't use $.IO0674 and you need to conserve disk storage, you can replace the $.SYSPRT data set supplied by IBM with a small empty data set.

4978 Data Stream Data Sets ($.RPQCS0 and $.RPQIS0)

$.RPQCS0 and $.RPQIS0 are the control and image stores for the 4978 data stream RPQ, D02428. These data sets are required only if you use the data stream IOCP, $.IO0670, to control 4978 or 4980 terminals. They must reside on the same volume as the I/O control program.

4978 EDX Data Sets ($4978CS0 and $4978IS0)

$4978CS0 and $4978IS0 are the control and image stores for the 4978 terminal that are distributed with EDX. They must reside on the same volume as the I/O control program.

4980 Data Stream Data Sets ($4980CSA, $4980ISA, and $4980R0A)

$4980CSA, $4980ISA, and $4980R0A are the data sets required for Communications Facility support of the 4980 as a 3277. These data sets must reside on the IPL volume if you are using $.IO0670 to control 4980 terminals.

4980 EDX Data Sets ($4980CS0, $4980IS0, and $4980R01)

EDX requires data sets $4980CS0, $4980IS0, and $4980R01 for support of the 4980. They must reside on the IPL volume.

Panel Data Set ($.SYSPNL)

The panel data set ($.SYSPNL) is a partitioned data set that contains panels displayed by the panel design aid ($.PANEL), the sample program ($.CFMENU) and the SNA I/O control program ($.IO14E8). It is required only if you use any of those programs; it must reside on the same volume as the program.

As supplied by IBM, $.SYSPNL contains 16 members, each of which contains a panel description as follows:

CFMENU The panel used by $.CFMENU.

PHASE*n* The six prompting panels used by $.PANEL, where *n* is the number of the design phase. Phase 4 has two panels, one for protected fields and one for unprotected fields.

HELP*xy* The seven help panels used by $.PANEL. *x* is the number of the design phase. *y* is 1 for the first help panel of those phases that have two panels. *y* is 2 for the second or only help panel of a phase.

SNALOGON The standard panel used by the SNA I/O control program.

SNALOG02 An alternate panel used by 8100 DPPX users of the SNA I/O control program.

You can use $.PANEL to modify these supplied panels, but be sure that you change only the contents of fields. If you rearrange fields or change their attributes, the programs that use the panels may not work correctly. If you modify any of the panels used by $.PANEL itself, first make a copy of $.SYSPNL with a different name or on a different volume. Modify the panels in the second data set and then copy them to the original $.SYSPNL.

$.SYSPNL contains no space for additional members. Allocate other data sets for application program panels, as described in the *Programmer's Guide*.

Program Dispatcher Data Set ($.SYSPD)

The program dispatcher data set ($.SYSPD) contains CP commands, path definitions, transaction definitions, and transactions that are to be processed when the program dispatcher is started. It is a sequential data set and is required only if you use the program dispatcher.

The chapter "Creating $.SYSPD" on page 187 explains the contents of this data set in detail.

IPL Transaction Data Set ($.SYSLCC)

The IPL transaction data set ($.SYSLCC) contains IPL transactions used to IPL remote nodes on a Local Communications Controller ring. $.SYSLCC is an optional sequential data set. It is required if you do remote IPLs of any type, but is actually used only to process hardware IPL requests initiated at a remote node, as described in the chapter "Communications Facility Components" on page 1. It must reside on the same volume as the program that processes IPL transactions, $.PDIPL.

As supplied by IBM, $.SYSLCC contains ten records with a /* statement. You can use any EDX text editor to enter transactions into $.SYSLCC. One record is required for every two statements. The last transaction must be followed by a /* to indicate the end of the list of transactions. Any statement that begins with an asterisk is ignored.

The format of each statement is:

name *IPL transaction*

where *name* is the 8-byte station name of the remote node to be IPLed; the *IPL transaction* begins at the ninth byte. For the format of the IPL transaction, see the *Operator's Guide*.

Emulated PC Disk Control Data Set ($.DSPROF)

The emulated PC disk control data set ($.DSPROF) contains information about all the emulated PC disks on a Series/1 volume. $.DSPROF is created by $.DSFORM, the emulated PC disk management utility program, when a command to create, delete, rename, or list data sets is entered. It has 24 entries—each entry represents an emulated PC disk. For each emulated PC disk created, deleted, or renamed, the corresponding Series/1 data set name, data set size, and an active flag will be updated in $.DSPROF. $.DSPROF data set exists for each Series/1 volume used to contain emulated PC disks. A maximum of 24 emulated PC disks can be created for each Series/1 volume.

PC Connect Microcode Patch Data Set ($PCCR0n)

$PCCR0*n* is the data set required to support PC Connect. It is used by the PC Connect initialization routines and it must reside on the IPL volume.

When you IPL the Series/1, the Communications Facility initialization module CSPCCPTC reads the EC level for each Series/1-PC Connect Attachment and loads the microcode patches from the appropriate data set.

X.25 Call ID Data Set ($.SYSX25)

The X.25 call ID data set ($.SYSX25) contains 2-digit call IDs associated with X.25 network addresses used to establish virtual calls.

It is a sequential data set and is required only if you use the X.25 IOCP ($.IO0AB8). It must reside on the same volume as the X.25 IOCP.

As supplied by IBM, $.SYSX25 contains 10 records with a /* statement. You can use any EDX text editor to enter call IDs and network addresses into $.SYSX25. The last address must be followed by a /* statement to indicate the end of the list of addresses. Any statement that begins with an asterisk is ignored.

The format of each statement is:

call-id name call-addr comment

where:

call-id	is any 2-digit decimal ID between 1 and 99.
name	is any 1- to 8-character name of your choice.
call-addr	is the 1- to 15-digit X.25 network address to be associated with the call ID.
comment	is an optional comment field. It is never examined.

The parameters may start in any column and must be separated by at least one blank. See "Example 13: X.25 Connection Between Series/1s through an X.25

Network" on page 392 or "Example 14: X.25 Connection between a Series/1 and DTEs" on page 406 for sample $.SYSX25 data sets.

Work Session Controller Data Set ($.WSCIMG)

The work session controller data set ($.WSCIMG) is a partitioned data set that contains images that can be displayed through the work session controller and areas used by application programs to save data. $.WSCIMG is required only if you use the work session controller. It must reside on the same volume as the work session controller, $.WSC.

As supplied by IBM, $.WSCIMG contains one member:

MENU — The image used by the work session controller sample program, $.WSMENU.

It also contains 29 directory entries and 82 member records that are available for your images and save areas.

You can use the EDX utility $IMAGE to create screen images and then transfer them to the $.WSCIMG data set using the utility program $.WSCUT1, as described in the *Programmer's Guide*. Each image member requires eight records.

Your application programs can use the save areas through the work session controller save (SV) and restore (RS) commands, also described in the *Programmer's Guide*. Use the EDX utility $DIUTIL to allocate save area members of whatever size your application requires. The member name of a save area must be the same as the terminal name that the program is using for the session in progress.

Work Session Controller Image ($.MENU)

$.MENU image is described under "Work Session Controller Data Set ($.WSCIMG)," in the format created by the EDX utility $IMAGE. The Communications Facility doesn't use this data set. It is provided so that you can modify the image. You may want to do this, in conjunction with modifying the sample programs that use it, to suit your installation's transaction-processing requirements. Use $IMAGE to modify the image and then use $.WSCUT1 to transfer it to data set $.WSCIMG.

Disk-Queue Data Sets

A data set is required for each station that uses disk-queued messages. See the chapter "Creating Disk-Queue Data Sets" on page 207 for information about these data sets.

Creating $.SYSPD

$.SYSPD is the data set that contains commands, path definitions, transaction definitions, and transactions that are processed when the program dispatcher is started. You need to create a $.SYSPD data set for each Series/1 where the program dispatcher runs. The data set must reside on the same volume as the program dispatcher.

This chapter describes the contents of $.SYSPD and the statements that you use to define paths and transactions. The chapter "Communications Facility Components" on page 1 explains how the information you supply in $.SYSPD governs the program dispatcher's actions. The chapter "Sample Configurations" on page 269 contains several examples of $.SYSPD data sets.

$.SYSPD Contents

$.SYSPD is a sequential data set divided into four sections by /* statements:

Commands
/*
Cell and path definitions
/*
Transaction definitions
/*
Transactions
/*

As supplied by IBM, $.SYSPD contains ten records with four /* statements. Use any EDX text editor to enter the data you require between the /* statements. The program dispatcher ignores any statement that begins with an asterisk.

Commands

The first section of $.SYSPD contains commands, which the program dispatcher sends to the command processor for execution. You can enter any CP or PD commands that you want to be executed when the program dispatcher is started. For example, you might enter commands to start the stations that represent physical connections between cells. The *Operator's Guide* gives the syntax of commands.

Cell and Path Definitions

The program dispatcher builds its path table from the definitions you enter into the second section of $.SYSPD. You can enter a CELL statement to define the local cell, PATH statements to define paths to remote cells, and a RESERVE statement to reserve space in the path table. The section "$.SYSPD Definition Statements" gives the syntax of these statements.

Transaction Definitions

The program dispatcher builds its transaction table from the definitions you enter into the third section of $.SYSPD. You can enter TID and EQU statements to define transactions, RTE statements to define reroute entries, and a RESERVE statement to reserve space in the transaction table. The section "$.SYSPD Definition Statements" gives the syntax of these statements.

Transactions

The fourth section of $.SYSPD contains transactions, which the program dispatcher routes to the appropriate transaction-processing program. You can enter any transaction you want to be processed when the program dispatcher is started. For example, you might enter a SYST SP transaction to notify CSPDIO (the remote disk module) that the program dispatcher is active, or you might enter transactions to start your transaction-processing applications. The *Operator's Guide* gives the syntax of IBM-supplied transactions.

See "Remote IPL" on page 37 for restrictions on sending broadcast transactions.

$.SYSPD Definition Statements

This section gives the details of the format and operands of the $.SYSPD definition statements—the statements that you can insert into the second and third parts of $.SYSPD. First, here's an explanation of the syntax notation that's used to describe the statements.

Syntax Notation

Each statement format in this chapter appears in a box with two columns:

Operation	Operand

Operation Field. This field contains the statement name.

Operand Field. This field contains the operands associated with the statement.

The following conventions are used within the format descriptions:

- Words in **BOLD CAPITAL** letters must be coded exactly as shown. Commas must also be coded exactly as shown.

- Words in *italics* are symbols for which you must substitute actual values.
- Brackets ([ ]) indicate that the operand or suboperand is optional. For example, if the syntax of an operand is shown as:

  [[**A**],[**B**],[**C**]]

  then you could leave the operand out entirely; or you could code any of these variations:

  A,B,C
  A,B
  A,,C
  A
  ,B,C
  ,B
  ,,C

- Braces ({ }) indicate a group of mutually exclusive operands or values, of which you can code only one.
- A vertical bar (|) separates the mutually exclusive items within braces.

CELL—Define the Local Cell

The CELL statement defines the local cell's identifier. Enter one CELL statement in the second section of $.SYSPD. If you omit the CELL statement, the default cell identifier is two EBCDIC zeros.

CELL Syntax

CELL	*cellid*

CELL Operands

cellid
is a 2-character cell identifier. Any two characters except ?? and ** are valid.

CELL Example

```
CELL S1
```

This statement assigns the identifier S1 to the local cell.

EQU—Define Equivalent Transaction

The EQU statement defines a transaction that has the same attributes as another transaction, except as specified by optional operands. EQU gives you a simple way to define transactions with identical or similar attributes. Enter EQU statements in the third section of $.SYSPD.

EQU Syntax

EQU	*tranid-1* *tranid-2* [[*type*],**[P]**,**[S]**,**[R]**] **[H]**

EQU Operands

tranid-1
is the 1- to 4-character identifier of the transaction whose attributes are being defined.

tranid-2
is the 1- to 4-character identifier of the transaction whose attributes are to apply to the transaction *tranid-1*. *tranid-2* must be defined by a TID statement.

type
is a 2-digit transaction type code, as explained in the description of the TID statement. If you omit this operand, the type defined for *tranid-2* applies.

P
indicates that the program defined for *tranid-2* is purgable.

S
indicates that this transaction is a 3270 data stream that the program dispatcher is not to modify. If you omit this operand, the specification defined for *tranid-2* applies.

R
indicates that the program load retry limits are to be ignored for this transaction. If you omit this operand, the specification defined for *tranid-2* applies.

H
indicates that the transaction is to be held on the queue of station $.PH*xxxx*, where *xxxx* is the transaction identifier.

EQU Example

```
EQU TR3B TR3A
```

This statement defines a transaction with identifier TR3B that has the same attributes as transaction TR3A.

PATH—Define a Path

The PATH statement defines the path to a remote cell. Enter PATH statements in the second section of $.SYSPD.

PATH Syntax

PATH	*station-name* *,cellid* [{,**S** | ,**C**,*prefix*}] [[**P**],[**H**]]

PATH Operands

station-name
is the 1- to 8-character name of the station that represents the path to a remote cell.

cellid
is the 2-character identifier of the remote cell.

S
indicates that transactions routed over this path are 3270 data streams that the program dispatcher is not to modify.

C
indicates that the program dispatcher is to do binary data conversion of the transactions routed over this path. Use this parameter only for a BSC path to a non-Series/1 host.

prefix
is a 4-character prefix to be appended to transactions routed over this path.

P
indicates that this is the preferred path over which transactions for unknown cells are to be routed. You may define only one preferred path in each cell.

H
indicates that transactions for this path are to be held on the queue of station $.PH*xx*, where *xx* is the identifier of the remote cell.

PATH Examples

```
PATH E3277A,S2
```

This statement specifies that transactions with cell identifier S2 are to be sent to station E3277A.

```
PATH HOSTTERM,CI,,CSSN P
```

This statement specifies that transactions with cell identifier CI or with an unknown cell identifier are to be prefixed with CSSN and sent to station HOSTTERM.

RESERVE—Reserve Table Entries

The RESERVE statement reserves space in the program dispatcher's tables. If you have reserved space in the tables, you can use the PD I command to insert path or transaction definitions after the program dispatcher is started. Enter a RESERVE statement in the second section of $.SYSPD to reserve space in the path table. Enter a RESERVE statement in the third section of $.SYSPD to reserve space in the transaction table.

RESERVE Syntax

RESERVE	*number*

RESERVE Operand

number
is the number of entries to be reserved.

RESERVE Example

```
/*
CELL S1
PATH E3277A,S2
RESERVE 2
/*
TID IPL $.PDIPL 22
RESERVE 4
/*
/*
```

This example reserves two entries in the path table and four entries in the transaction table.

RTE—Define a Reroute Entry

The RTE statement defines the destination cell for a transaction whose primary cell identifier is the local cell. You can't have both an RTE statement and a TID statement for the same transaction. Enter RTE statements in the third section of $.SYSPD.

RTE Syntax

RTE	*tranid* *cellid* **[H]**

RTE Operands

tranid
is a 1- to 4-character transaction identifier.

cellid
is the 2-character identifier of the cell to which the transaction is to be routed.

H
indicates that the transaction is to be held on the queue of station $.PH*xxxx*, where *xxxx* is the transaction identifier.

RTE Examples

```
RTE PRT1 C3
```

This statement specifies that transactions with identifier PRT1 are to be routed to cell C3 instead of being processed in the local cell, as their primary cell identifier specifies.

```
RTE PRT1 C3 H
```

This statement specifies that transactions with identifier PRT1 are to be held on the queue of station $.PHPRT1. When the hold is released (by a PD F command or a SYST RH transaction), the held transactions are routed to cell C3.

TID—Define a Transaction

The TID statement defines a transaction to be processed in the local cell. Enter TID statements in the third section of $.SYSPD.

TID Syntax

TID	*tranid* *program* [,[*volume*],[*partition*],[*prefind*]] [[*type*],[**P**],[**S**],[**R**]] [**H**]

TID Operands

tranid
is a 1- to 4-character transaction identifier.

program
is the 1- to 8-character name of the program that is to process the transaction.

volume
is the 1- to 6-character name of the volume where the program resides. The default is the IPL volume.

partition
is the partition where the program is to be loaded. Specify one of the following:

0
to mean any available mapped partition. 0 is the default.

1 to 16
to mean the specified mapped partition. If the partition does not exist or is not mapped, the partition specification is set to 0 in the transaction table.

-1 to -16
to mean any available mapped partition except the one specified.

CF
to mean the $.CF partition.

NCF
to mean any mapped partition except the $.CF partition.

prefind
indicates whether or not a prefind of the program's data sets and overlays is to be performed when the program dispatcher is started. Omit the operand to enable prefind; specify N to disable prefind.

type

is a 2-digit transaction type code. The first digit indicates the transaction-processing program's type. Specify one of these values:

1 A single-transaction program that processes one transaction and then terminates.

2 A multitransaction program that processes multiple transactions and then terminates.

3 A continuous program that remains in storage, processing transactions, until it is told to stop.

4 A continuous, reentrant program that remains in storage, processing transactions, until it is told to stop.

The only significance of these codes to the program dispatcher that a type 1 program is loaded (if the second digit of the type code so specifies) each time the transaction is entered, even if the program is already in storage; type 1 and 2 programs are not purgable.

The second digit of the type code indicates what the program dispatcher is to do when it receives the transaction. Specify one of these values:

0 Load a program; do not create a station; do not send the transaction.

1 Load a program; create a station and its associated message queue; do not send the transaction.

2 Load a program; create a station and its associated message queue; send the transaction to the station.

3 Do not load a program; create a station and its associated message queue; send the transaction to the station.

P

indicates that the program is purgable; the program dispatcher can tell it to stop if its storage is needed for some other transaction-processing program. You can specify that the program is purgable only if the transaction type code is in the range 30-43.

S

indicates that the transaction is a 3270 data stream that the program dispatcher is not to modify.

R

indicates that the program load retry counts are to be ignored for this transaction. When there is no storage available for the program, the load is to be retried until it succeeds or until the program dispatcher is shut down.

H

indicates that the transaction is to be held on the queue of station $.PH*xxxx*, where *xxxx* is the transaction identifier.

Figure 36 summarizes the transaction types. The default transaction type is 10.

Transaction Type	Load Single Transaction Program	Load Multi Transaction Program	Load Continuous Program	Load Reentrant Continuous program	Create Station and Queue	Put Transaction On Queue
10	Y	N	N	N	N	N
11	Y	N	N	N	Y	N
12	Y	N	N	N	Y	Y
13	N	N	N	N	Y	Y
20	N	Y	N	N	N	N
21	N	Y	N	N	Y	N
22	N	Y	N	N	Y	Y
23	N	N	N	N	Y	Y
30	N	N	Y	N	N	N
31	N	N	Y	N	Y	N
32	N	N	Y	N	Y	Y
33	N	N	N	N	Y	Y
40	N	N	N	Y	N	N
41	N	N	N	Y	Y	N
42	N	N	N	Y	Y	Y
43	N	N	N	N	Y	Y

Figure 36. Transaction Types

TID Examples

```
TID MENU $.WSMENU 22
```

This statement specifies that the program $.WSMENU is to process transactions with transaction identifier MENU. When the program dispatcher is started, it will do a prefind for $.WSMENU. When the program dispatcher receives a MENU transaction, it will load $.WSMENU from the IPL volume into any available partition, if the program isn't already in storage. It will create station $.WSMENU, if the station doesn't already exist; and it will send the transaction to $.WSMENU.

```
TID RPT REPORT,USRLIB,NCF 20
```

This statement specifies that transactions with transaction identifier RPT are to cause program REPORT to be executed. When the program dispatcher is started, it will do a prefind for REPORT. When the program dispatcher receives an RPT transaction, it will load REPORT from volume USRLIB into any available partition except the one in which $.CF is loaded, if the program isn't already in storage.

```
TID T327 T327PGM,,,N 22,,S
```

This statement specifies that transactions with identifier T327 are 3270 data streams to be processed by program T327PGM. When the program dispatcher is started, it will *not* do a prefind for T327PGM. When the program dispatcher receives a T327 transaction, it will load T327PGM from the IPL volume into any available partition, if the program isn't already in storage. It will create station T327PGM, if the station doesn't already exist. It will send the transaction, without modification, to T327PGM.

Definitions for IBM-Supplied Transactions

This section shows the TID statements for IBM-supplied transactions. Enter the ones you require in your $.SYSPD data sets. You can enter them as shown, or you can supply the volume name or the partition where the program is to be loaded. Note that no TID statement is required for SYST transactions.

Transaction Scheduler:

```
TID SCHD $.PDSCHD,,,N 32
```

Remote Management Functions, Host Cell:

```
TID HMU $.HMU 22
```

Remote Management Functions, Remote Cells:

```
TID RMU $.RMU 22
```

Remote IPL:

```
TID IPL $.PDIPL 22
```

Remote Disk Support, Cells where Disks are Located:

```
TID <IO> $.PD<IO>,,,N 32
```

Work Session Controller:

```
TID WSC $.WSC 42
```

Work Session Controller Sample Program:

```
TID MENU $.WSMENU 22
```

Using a Private Data Set

When you're testing transaction-processing applications, you may want to use a program dispatcher data set whose name is other than $.SYSPD or one that resides on a different volume from the program dispatcher. You can do so by starting the program dispatcher as follows:

1. Use the AL command of utility program $.UT1 to create a station block named $.PD with a network address appropriate to your configuration.

2. Use the SE command of utility program $.UT1 to send a message to $.PD, specifying the data set name and volume name you want to use. The text of the message is *xxxxxxxxyyyyyy*, where *xxxxxxxx* is the 8-character data set name and *yyyyyy* is the 6-character volume name.

3. Use the EDX $L command to load $.PD.

The program dispatcher now starts with the data set you specified for this session only.

Creating Disk-Queue Data Sets

A disk-queue data set is a data set on disk that is used to hold low-priority messages sent to, but not yet received by, a particular station. Each station that is to have a disk queue requires a separate disk-queue data set. This chapter explains how to create disk-queue data sets and how to display information about them.

Disk-Queue Data Set Specifications

To create a disk-queue data set, you must allocate the data set, initialize it, and assign it to a station. You can use the worksheet shown in Figure 37 on page 208 to record the information you will specify when you create disk-queue data sets.

Fill in the worksheet as follows:

Station Name
: is the 1- to 8-character name of the station to which this disk-queue data set is assigned.

Data Set Name
: is the 1- to 8-character name of the disk-queue data set.

Volume Name
: is the 1- to 6-character name of the volume on which the data set resides. The volume must not be a fixed-head volume.

Size
: is the size of the data set, expressed as the number of 256-byte records. The first record of the data set is used for control information. The remaining records are used to hold messages. Each message includes a 24-byte message header and requires one or more entire records.

  To determine the size required for a disk-queue data set, you must have some idea of the size of the messages that will be sent to the station and the maximum number of messages that will accumulate in the data set. If you have enough disk storage, start out by allocating data sets larger than you expect to need. After the data sets have been in use for a while, determine the maximum space ever used (see "Displaying Information about Disk-Queue Data Sets" on page 211) and adjust the data sets' sizes accordingly.

  The data set for $.WASTE (the undeliverable message station) requires more space than most disk-queue data sets, for two reasons. First, the messages aren't received (unless you've written a program to do so); they accumulate in the data set until you use utility program $.UT2 to process them. Second, each undeliverable message results in two messages being written to the data set: a

Disk-Queue Data Set Worksheet

Station Name	Data Set Name	Volume	Size in Records	Active (Y/N)	Warning Level %	Warning Dead-band %	Overlay Mode (Y/N)

Figure 37. Disk-Queue Data Set Worksheet

22-byte reason message that contains the date, time, and a reason code, and the undeliverable message itself.

Active
indicates whether or not the disk queue is to be active when the station is started.

Warning Level %
is the level, expressed as a percentage of total data set capacity, at which you want a system log message issued when the data set contents have exceeded that level. If you specify that you want no warning messages, the level is set to 0.

Warning Deadband %
is the level, expressed as a percentage of total data set capacity, that controls the number of warning messages issued. After a warning message is issued, no more are issued until the data set contents drop below the warning level by the amount of the deadband and then rise above the warning level again. The default warning deadband is 10%.

Overlay Mode
indicates whether or not the data set is to be used in overlay mode. In overlay mode, when the data set doesn't have enough free space for a new message, the new message overlays the oldest messages already in the data set. If the warning level is other than 0, this overlaying causes a system log message to be issued.

In nonoverlay mode, a message is not written to the data set unless there is enough free space. When there isn't enough space for a message, the sender gets a return code indicating that the message was not sent.

Allocating Disk-Queue Data Sets

Use the EDX utility program $DISKUT1 to allocate disk-queue data sets. For example:

```
AL QUEFILE 50 D
```

Initializing Disk-Queue Data Sets

Use the Communications Facility utility program $.DSINIT to initialize the data sets with the control information required for disk queuing. The program prompts you for the required information.

Here is an example:

> $L $.DSINIT

```
$.DSINIT          22P,01:52:06, LP= 4000
THIS PROGRAM INITIALIZES PREVIOUSLY ALLOCATED
DATA SETS FOR USE AS MESSAGE QUEUES

ENTER DATA SET NAME,VOLUME: QUEFILE12
INITIALIZE QUEFILE,EDX002 ? Y
IS THE DATA SET TO BE COMPLETELY INITIALIZED ? Y
CAPACITY WARNING LEVEL TO BE SPECIFIED ? Y 80
THE DEFAULT DEADBAND VALUE FOR WARNING MESSAGES IS 10%, OK ? Y
IS THE DATA SET TO OPERATE IN OVERLAY MODE ? N
QUEFILE,EDX002 HAS BEEN INITIALIZED FOR MESSAGE QUEUING
MORE DATA SETS TO INITIALIZE ? N
$.DSINIT ENDED AT 01:52:54
```

If you need to initialize multiple data sets with identical sizes and specifications, you can allocate and initialize one of them and then use the EDX utility program $COPYUT1 to make copies with different names.

You can also reinitialize disk-queue data sets with $.DSINIT, either completely or partially. You can change the warning level, the warning deadband, overlay/nonoverlay mode, and the high water mark (maximum data set level reached during operation). The description of $.DSINIT in the *Operator's Guide* gives more information about reinitializing disk-queue data sets and an example.

Assigning Disk-Queue Data Sets to Stations

Notice that you don't specify a station name when you initialize a disk-queue data set. The data set doesn't show which station it's assigned to (although it does show which station last used it). Instead, the definition of a station in $.SYSNET shows which disk-queue data set is currently assigned to that station.

This arrangement gives you considerable flexibility in associating disk-queue data sets with stations. For example, if a disk-queue data set fills up, you can stop the station, assign it another data set, restart the station, and save the first data set for later processing. You can deliberately collect messages for later processing by assigning a data set to a station that doesn't receive the messages sent to it; when you're ready to process the messages, stop the station and assign the data set to another station that does receive and process its messages. What you *can't* do is assign the same data set to more than one station at a time.

You can assign a disk-queue data set to a station when you define the station, using either $.CONFIG or the CP DEF command. You assign the data set by specifying its name, volume, and whether or not it is to be active when the station is started. The data set doesn't have to exist when you do this; you can allocate and initialize it later. You must allocate and initialize it before you start the station.

---

12 The default volume name for the disk-queue data set is the one where $.DSINIT resides.

You can use the CP FILE NAME command or the $.CONFIG CHANGE command to modify the disk-queue data set assignment for an existing station. You can assign a data set to a station that doesn't have one, assign a different data set to a station that does have one, or unassign a data set. You can use the CP F DISKQ command to change the status of a station's disk queue from active to inactive, or vice versa.

Displaying Information about Disk-Queue Data Sets

The CP FILE PARMS command displays the names and volumes of the disk-queue data sets assigned to stations. If a data set is open (is currently in use by the station), the command also displays the information you specified for the data set, the number of messages it contains, the percentage of capacity currently in use, and the high water mark.

The CP ST command displays, for an open disk-queue data set, the number of messages it contains, the percentage of capacity currently in use, and the high water mark.

Utility program $.DSINIT displays the current high water mark when you request a partial reinitialization of a disk-queue data set.

Stations That Don't Need Disk Queues

There are some types of stations for which disk queues are never used. Either the messages sent to the station are placed on the queue of some other station, no messages are sent to the station, or the few messages that are sent are queued in storage.

The stations that don't need disk queues are:

- A user station that represents an I/O control program
- A line station of subtype 3271C, 3271E, LCC, PCC, CA, DTE, or DCE
- A terminal station of subtype 3271
- An SNA physical unit station
- An alias station
- A remote station
- A remote node station that is not on the local node's Local Communications Controller ring or the Series/1-PC Connect Attachment.

Planning Storage Requirements

This chapter will assist you in determining how much processor storage the Communications Facility will require.

The Communications Facility occupies space in the following parts of an EDX system:

- The EDL language extensions and remote disk support become part of the EDX supervisor. They must be located in partition 1.
- The system storage pool (S$POOL) and system tables (CSXSYS) become part of the common area.
- The Communications Facility control program ($.CFS or $.CFD), the Communications Facility message pool (CFBUF), the command processor transients, and the optional Series/1-to-Series/1 IOCP must all be in the same partition, called the $.CF partition.
- The remaining Communications Facility programs can run in any partition in which the common area is mapped.

Figure 38 on page 214 shows the partition structure of an EDX system with the Communications Facility installed.

To plan how much storage the Communications Facility will require, you need to consider:

- Which Communications Facility programs you will use.
- How large the various working storage areas used by the Communications Facility need to be to satisfy your requirements.
- Which programs will run in which partitions.
- The number of partitions and their sizes.

Communications Facility Programs

Figure 39 on page 216 shows the size of each executable component of the Communications Facility. The sizes listed may not always be accurate because of maintenance changes to the modules. Always check the latest program directory for the current module sizes.

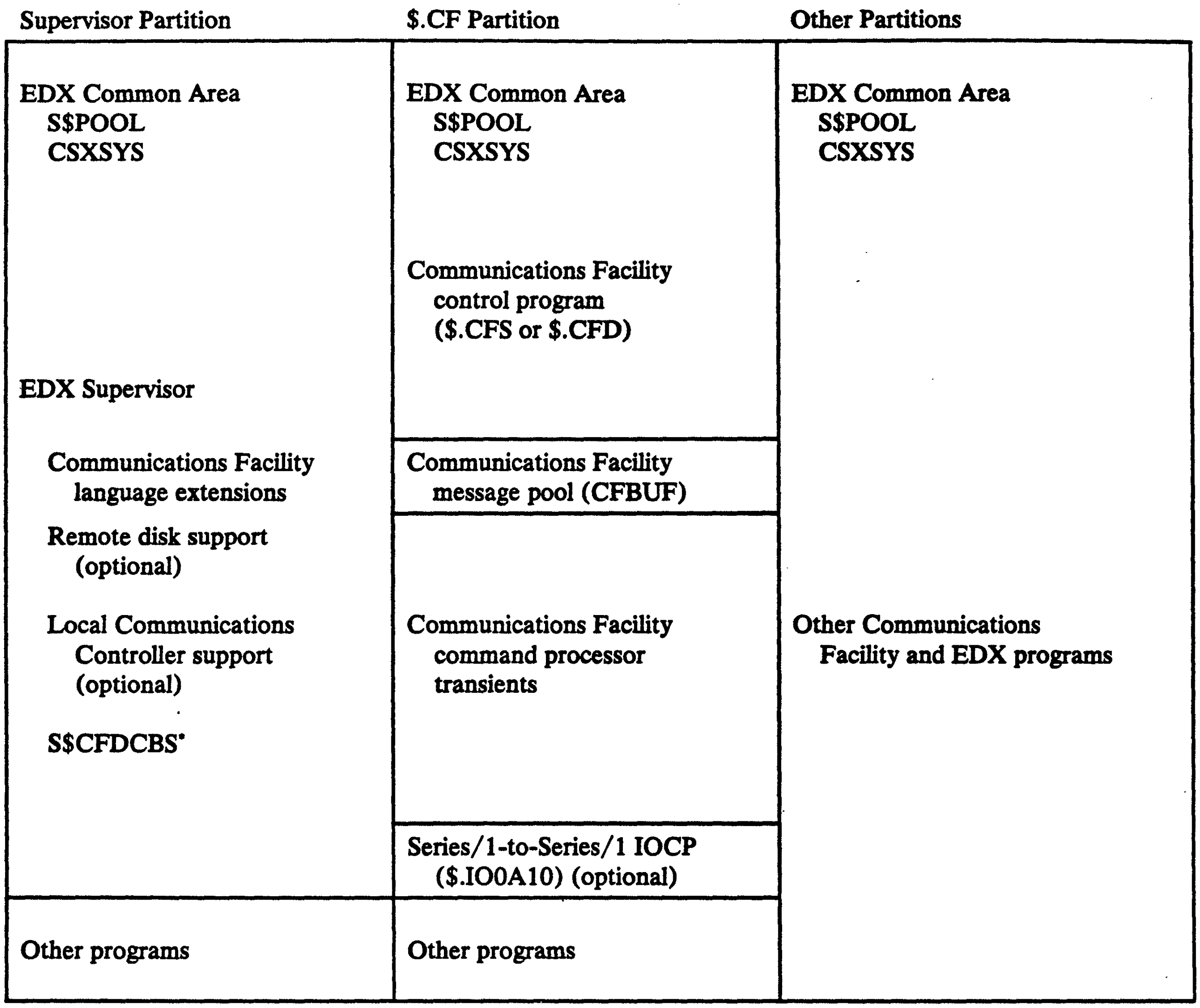


Figure 38. EDX/Communications Facility Partition Structure

Those whose names begin with 'A.' run as overlays of $EDXASM. They assemble the language extension instructions.

Those whose names begin with 'CS' are object modules that become part of the EDX supervisor. The section "EDX Supervisor Storage Requirements" on page 220 explains which ones are required.

Those whose names begin with '$.' are load modules. The required programs are:

- The Communications Facility control program, either $.CFS or $.CFD.
- The initialization program, $.CFINIT. It runs in the $.CF partition in the area that is subsequently used for command processor transients.
- Command processor transient programs, those whose names begin with '$.CP'.

---

* This support is produced in EDX Version 5.2.

The rest of the programs are optional. Determine which ones you will be using and calculate the total storage requirements. Add to this total the storage required for your Communications Facility application programs and other EDX programs. Unless you have far more storage than required, decide which programs will run in each partition and size the partitions accordingly. The section "Program Loading and Partition Use" on page 232 explains how you can control which partition Communications Facility programs are loaded into.

Figure 39 on page 216 includes the programs that run as overlays, but you can ignore them when you're planning storage requirements. The size of the largest overlay is included in the total size of the programs that have overlays.

Module Name	Function	Size without Dynamic Storage		Dynamic Storage Size (Bytes)	Total Size (Bytes)
		Hex	Decimal		
A.CFID	Compile CFID instructions	0200	512	0	512
A.CFTERM[1]	Compile CFTERM instructions	0600	1536	0	1536
A.CSXDEF[1]	Compile DEFINE instructions	0D00	3328	0	3328
A.GETPUT[1]	Compile GET, PUT, and MOV instructions	1700	5888	0	5888
A.SENREC[1]	Compile SEND, RECEIVE, ACTIVATE, FREE, and LOCATE instructions	1600	5632	0	5632
CSA	Global attention list	14A	330	0	330
CSF	Process GET FIELD and PUT instructions	506	1286	0	1286
CSM	Process SEND and RECEIVE instructions	D1A	3354	0	3354
CSP	Communications Facility parameter table	C	12	0	12
CSPCCINT	PC Connect initialization	204	516	0	516
CSPCCIPL	PC Connect initialization call routine	18	24	0	24
CSPCCOPN	PC Connect microcode data sets open routine	6D0	1744	0	1744
CSPCCPTC	PC Connect microcode patch load routine	8C6	2246	0	2246
CSPDIO	Remote disk	1248	4680	0	4680
CSX	Process ACTIVATE, FREE, GET, LOCATE, and MOV instructions	B34	2868	0	2868
CSX[7]	Process ACTIVATE, FREE, GET, LOCATE, and MOV instructions	B34	2868	0	2868
CSXINIT	Initialize EDX command table	31E	798	0	798
CSXRDINT	Remote disk initialization	250	592	0	592
CSXSYS	Communications Facility system tables	2CC	716	0	716
$.CFD	Control program (disk queuing)	2A00	10752	7680	18432
$.CFINIT	System initialization	0B00	2816	0	2816
$.CFMENU	Sample program	0C00	3072	0	3072
$.CFS	Control program (storage queuing)	1A00	6656	7680	14336
$.CFSHUT	Shut down the Communications Facility	0600	1536	0	1536
$.CK14E8[1]	Process SNA logon responses	0800	2048	0	2048
$.CK14E8[1], [7]	Process SNA logon responses	0800	2048	0	2048
$.CONFIG	Configuration processor utility	1100	4352	0	4352
$.CPCMCF	Process Communications Manager commands	1300	4864	512	5376
$.CPDEF	Process CP define commands	1600	5632	0	5632
$.CPDEF[7]	Process CP define commands	1600	5632	0	5632
$.CPF	Process CP modify commands	1D00	7424	0	7424

[1] Overlays
[2] Includes the size of the largest overlay
[3] Includes the size of the EDX largest overlay
[4] With LU-1 support
[5] Without LU-1 support
[6] Requires an additional 6144 decimal bytes in the same partition on a 2K boundary
[7] EDX Secondary SNA2 support

Figure 39 (Part 1 of 4). Module Sizes

Module Name	Function	Size without Dynamic Storage		Dynamic Storage Size (Bytes)	Total Size (Bytes)
		Hex	Decimal		
$.CPF[7]	Process CP modify commands	1D00	7424	0	7424
$.CPFILE	Process CP file commands	0E00	3584	0	3584
$.CPH	Process CP halt commands	0900	2304	0	2304
$.CPHELP	Process CP help commands	1B00	6912	0	6912
$.CPHELP[7]	Process CP help commands	1C00	7168	0	7168
$.CPLINK	Process CP link commands	0800	2048	0	2048
$.CPP	Process CP stop commands	A00	2560	0	2560
$.CPPD	Process PD commands	1C00	7168	0	7168
$.CPQ[2]	Process CP query commands	600	1536	0	5888[2]
$.CPQOV1[1]	Process CP query commands	1100	4352	0	4352
$.CPQOV2[1]	Process CP query commands	0800	2048	0	2048
$.CPQOV3[1]	Process CP query commands	0A00	2560	0	2560
$.CPQOV4[1]	Process CP query commands	0700	1792	0	1792
$.CPQOV5[1]	Process CP query commands	1200	4608	0	4608
$.CPQOV5[7]	Process CP query commands	1200	4608	0	4608
$.CPREAD	Process CP read commands	0B00	2816	0	2816
$.CPS	Process CP start commands	2000	8192	0	8192
$.CPS[7]	Process CP start commands	2000	8192	0	8192
$.CPSET	Process CP set commands	800	2048	0	2048
$.CPSHUT	Shut down the Communications Facility	0200	512	0	512
$.CPST	Process CP statistics commands	1A00	6656	0	6656
$.CPV	Process CP vary commands	0700	1792	0	1792
$.DSFORM	Emulated PC disk management utility	2800	10240	0	10240
$.DSINIT	Initialize disk queue data sets	1700	5888	0	5888
$.HMU	Host management utility	1200	4608	1024	12544[2]
$.HMUOV1[1]	Shutdown and execute program	0F00	3840	0	3840
$.HMUOV2[1]	Send and receive data sets	1B00	6912	0	6912
$.HMUOV3[1]	Host management utility functions	1100	4352	0	4352
$.HSL	High-speed loader	0200	512	0	512
$.IO0AA8	PC Connect IOCP	2700	9984	6144	16128
$.IO0AB0	Local Communications Controller IOCP	1D00	7424	6144	13568
$.IO0AB8	X.25 IOCP	6400	25600	5120	30720

[1] Overlays
[2] Includes the size of the largest overlay
[3] Includes the size of the largest EDX overlay
[4] With LU-1 support
[5] Without LU-1 support
[6] Requires an additional 6144 decimal bytes in the same partition on a 2K boundary
[7] EDX Secondary SNA2 support

Figure 39 (Part 2 of 4). Module Sizes

Component Name	Function	Size without Dynamic Storage		Dynamic Storage Size (Bytes)	Total Size (Bytes)
		Hex	Decimal		
$.IO0AC0	3270 control IOCP	1300	4864	6144	11008
$.IO0AD0	Channel attach IOCP	1A00	6656	6144	12800
$.IO0AE0	3270 emulation IOCP	1500	5376	6144	11520
$.IO0A10	Series/1-to-Series/1 BSC IOCP	0C00	2816	0	2816
$.IO0A20	Point-to-point BSC IOCP	0C00	3072	6144	9216
$.IO06F0	3101 IOCP	5600	22016	5120	27136[6]
$.IO0670	4978/4980 data stream IOCP	2900	10496	6144	16640
$.IO0674	Printer IOCP	1900	6400	8192	14592
		2000	8192	8192	16384[4]
$.IO0680	7485 IOCP	5800	22528	8960	31488[6]
$.IO14E8	SNA IOCP	3300	13056	6144	19200[2]
$.IO14E8[7]	SNA IOCP	3B00	15104	6144	21248[2]
$.IO1CC8	Disk-Server IOCP	1200	4608	5376	9984
$.MIG212	Convert 3101F and printer stations	1900	6400	0	6400
$.MIGR20	Convert 3101F station	1900	6400	0	6400
$.MIGR211[7]	Convert SNA1 PU and LU stations	1800	6144	0	6144
$.OPAID	Process $.CONFIG assist commands	0300	768	0	768[5]
$.OPCH	Process $.CONFIG change commands	5C00	23552	0	23552
$.OPCH[7]	Process $.CONFIG change commands	5900	22784	0	22784
$.OPCOPY	Process $.CONFIG copy commands	1B00	6912	0	6912[5]
$.OPDEF	Process $.CONFIG define commands	6A00	27136	0	27136
$.OPDEF[7]	Process $.CONFIG define commands	6600	26112	0	26112
$.OPDISP	Process $.CONFIG display commands	2700	9984	0	9984
$.OPDISP[7]	Process $.CONFIG display commands	2700	9984	0	9984
$.OPEDIT	Process $.CONFIG edit commands	0F00	3840	0	3840
$.OPHELP	Process $.CONFIG help commands	0500	1280	0	1280
$.OPLINK	Process $.CONFIG link commands	1000	4096	0	4096
$.OPLIST	Process $.CONFIG list commands	2B00	11008	0	11008
$.OPLIST[7]	Process $.CONFIG list commands	2C00	11264	0	11264
$.OPSETX	Process $.CONFIG set exit commands	0400	1024	0	1024[5]
$.PANEL	Panel design aid	4D00	19712	512	20224
$.PD	Program dispatcher	2A00	10752	1280	12032
$.PDBSTS	Volume protection utility	0900	2304	0	2304

[1] Overlays
[2] Includes the size of the largest overlay
[3] Includes the size of the largest EDX overlay
[4] With LU-1 support
[5] Without LU-1 support
[6] Requires an additional 6144 decimal bytes in the same partition on a 2K boundary
[7] EDX Secondary SNA2 support

Figure 39 (Part 3 of 4). Module Sizes

Component Name	Function	Size without Dynamic Storage		Dynamic Storage Size (Bytes)	Total Size (Bytes)
		Hex	Decimal		
$.PDHELP	Display PD commands	1200	4608	0	4608
$.PDINIT	Program dispatcher initialization	1900	6400	0	12032[2]
$.PDINPF[1]	Prefind transaction-processing programs	1600	5632	0	5632
$.PDIPL	Remote IPL	2600	11264	0	11264
$.PDSCHD	Schedule transactions	0700	1792	256	2048
$.PD<IO>	Remote disk I/O	500	1280	0	1280
$.PNLPRT	Print panel description for $.PANEL	2100	8448	0	8448
$.PNLUT1	Panel utility	1E00	7680	2560	10240
$.PR14E8[1]	Send SNA logon prompts	0700	1792	0	1792
$.PR14E8[1,7]	Send SNA logon prompts	0700	1792	0	1792
$.RMU	Remote management utility	1000	4096	1024	10496[3]
$.SETSTG	Modify storage for message buffer	0D00	3328	0	3328
$.UT1	Diagnostic aid utility	1200	4608	0	4608
$.UT2	Report and purge messages	2100	8448	512	8904
$.UT2DS	Open disk queues for $.UT2	0F00	3840	0	3840
$.WSC	Work session controller	2800	10240	2304	12544
$.WSCUT1	Catalog work session controller images	3B00	15104	0	15104
$.WSMENU	Work session controller sample program	0B00	2816	0	2816

[1] Overlays
[2] Includes the size of the largest overlay
[3] Includes the size of the largest EDX overlay
[4] With LU-1 support
[5] Without LU-1 support
[6] Requires an additional 6144 decimal bytes in the same partition on a 2K boundary
[7] EDX Secondary SNA2 support

Figure 39 (Part 4 of 4). Module Sizes

Communications Facility Working Storage

The Communications Facility has various working storage areas whose size you should adjust to suit your requirements.

Most of these areas are dynamic storage, as defined by the STORAGE operand of PROGRAM statements. The size of these areas, as distributed, is shown in Figure 39 on page 216. Use the SS command of the EDX utility $DISKUT2 to set the sizes you need. The storage requirements are explained later in this chapter.

Some of the storage areas are *workspace pools*, areas that include information used to control the allocation of space from the pool. Each pool has an 8-byte *buffer reference block (BRB)*. Each element in the pool has an 8-byte *storage resource block (SRB)*. The length of each element is a multiple of 8 bytes. An element can be a work area or a buffer. Each buffer has a 10-byte buffer header.

When you calculate the size of a pool, you must allow for this overhead. Suppose, for example, you need a pool large enough for three 2500-byte buffers. The space required for each buffer is 2520 bytes (2500 plus 8 for the SRB and 10 for the buffer header, rounded to a multiple of 8). The required pool size is 7568 bytes (3 times 2520, plus 8 for the BRB). If the pool is in dynamic storage, its size is 7680 bytes (7568 rounded to a multiple of 256).

EDX Supervisor Storage Requirements

The chapter "Installing the Communications Facility" on page 251 explains which EDX modules and hardware definition statements are required to support the various types of Communications Facility stations. Refer to the EDX *Installation and System Generation Guide* for information about estimating the size of an EDX supervisor. Add the storage required for the Communications Facility, as follows, using the module sizes shown in Figure 39 on page 216. All Communications Facility supervisor components must reside in partition 1 or the EDX COMBASE partition.

Modules CSA, CSP, CSF, CSM, CSX, and CSXINIT are required as follows: CSA, CSP and CSXINIT reside in partition 1; modules CSF, CSM and CSX reside in the EDX COMBASE partition.

Module CSXSYS, the system storage pool (CSXPOOL) and the DCB pool (CSXDCBS) are required as follows: The DCB pool must reside in partition 1; CSXSYS and CSXPOOL are part of the common area (described under "Common Area Storage Requirements").

Modules CSPDIO and CSXRDINT are required for remote disk support in nodes that access disks attached to remote nodes. CSPDIO must reside in partition 1, in the static section of the supervisor.

Each multifunction attachment 3101 or 7485 that is defined to the Communications Facility but not to EDX (each CFTERM instruction) requires 4 bytes.

Common Area Storage Requirements

The common area is that part of the supervisor that may be mapped into other partitions. It must be mapped into all partitions in which Communications Facility programs, including user application programs, are executed. The maximum partition size available for Communications Facility programs in partition 1 is 64K minus the size of the supervisor. In all other partitions into which the common area is mapped, the maximum partition size available is 64K minus the size of the common area rounded to a multiple of 2K. For example, if the size of the common area is 5K, then the maximum partition size available for Communications Facility programs is 58K (64K minus 6K).

Components of the Common Area

The common area must include S$POOL (system pool storage) and module CSXSYS (Communications Facility system tables). If you choose to base the EDX common area in partition 1, it will also contain EDXSYS (EDX system tables). It can also include other common data ($SYSCOM) required for your installation, as described in the EDX *Installation and System Generation Guide*. Unless you base the common area in partition 1, your application programs may not assume that EDXSYS is in the common area, nor that S$POOL elements reside in partition 1.

The size of EDXSYS is approximately 1400 bytes. It is different for different releases of EDX. Look at an EDX supervisor link-edit map to determine its exact size. The size of CSXSYS is given in Figure 39 on page 216. The sizes of S$POOL and S$CFDCBS are discussed under "S$POOL Storage Requirements" on page 221 and "S$CFDCBS Storage Requirements" on page 222.

Once you have estimated the size of S$POOL and the common area, you may want to increase the size of S$POOL so that the common area is a multiple of 2K. For example, your initial calculation of the common area size may be:

Element	Bytes
EDXSYS	1382
CSXSYS	560
S$POOL BRB	8
S$POOL	4000
Total	5958

This leaves 186 bytes of unused common area (6144 minus 5958). If you are not certain that the size of S$POOL is adequate for your requirements, increase it to 4184 (a multiple of 8). This will increase the size of the supervisor by 184 bytes, but will not affect the amount of storage available for Communications Facility programs in other partitions.

After you have installed the Communications Facility, look at the EDX link-edit map to check that the size of the common area is what you intended. Entry point CSXEND shows the size.

S$POOL Storage Requirements

During the installation procedure, you will define the system storage pool, S$POOL. The Communications Facility uses S$POOL for station control blocks, the high-speed loader, and work areas.

To calculate the size of S$POOL, add the sizes of the elements you require, using the following values. Each value includes 8 bytes for the SRB.

S$POOL Element	Bytes
Message dispatcher station ($.DISP)	104
Each CA, CPU, LCC, PCC, PTPT, 3271C or 3271E line station	264
Each DCE or DTE line station	112
Each circuit station	128
Each terminal or node station	88
Each volume station	104
Each 4978 or printer device station	264
Each 3101F device station	160
Each user or message station	88
Each alias station	40
SNA physical unit station	128
Each EDX Secondary SNA1 logical unit station	280
Each EDX Secondary SNA2 logical unit station	120
Each remote (vector) station	32
Each work session controller terminal station	392[13]
Program dispatcher space requirements:	
Program dispatcher station ($.PD)	88
High-speed loader ($.HSL)	520
Each transaction-processing station	88
Remote disk support station ($.PD>IO<)	88
$.IO0670 work areas (one for every three or four active 4978s/4980s)	56

If your application programs will use work areas in S$POOL, add the space they require. If your use of S$POOL is volatile (that is, stations of different sizes are active at different times), allow some extra space for fragmentation.

When the Communications Facility is running, you can use the DQ command of the diagnostic aid utility ($.UT1) to monitor the usage of S$POOL. The command displays information about each element in the pool, including free space. $.UT1 is described in the *Operator's Guide*.

S$CFDCBS Storage Requirements

During the installation procedure, you will define the DCB pool, S$CFDCBS, and include it in partition 1. The Communications Facility uses S$CFDCBS to hold one DCB for each device station it manages without the use of EDX-provided I/O facilities (for example, device 3101F stations). It also holds executable code to invoke Communications Facility interrupt handling routines.

To calculate the size of S$CFDCBS, allow 72 bytes for each 3101F device station you define in your Communications Facility configuration. This value includes 8 bytes for the SRB. Add another 8 bytes for the BRB.

If your application programs manage devices that require a DCB and they do not use EDX-provided I/O facilities, you may want to include space in S$CFDCBS for

[13] Plus user work area (0-128)

your application DCBs and interrupt handler calls. The Series/1 assembler OIO and IO instructions are not EDX-provided I/O facilities.

$.CF Partition Storage Requirements

The $.CF partition must be large enough to contain:

- The common area
- The control program ($.CFS or $.CFD)
- The message buffer pool
- Command processor transients
- Optionally, the Series/1-to-Series/1 IOCP ($.IO0A10).

Each command processor program is a transient program—it is loaded by the control program, processes one command, and terminates. The space required is 6912 bytes, which is the size of the start command processor. You can run other programs in the $.CF partition, but you must be careful not to use up the space required for the command processor.

Message Buffer Pool (CFBUF) Requirements

The message buffer pool is the dynamic storage of module $.CFS or $.CFD. It contains storage-queued messages that have been sent but not yet received. To estimate how much space is required, you need to consider the size of messages, whether or not messages are queued on disk, and how long messages remain in the pool.

If you use the Series/1-to-Series/1 IOCP ($.IO0A10), add the space required for its line buffer as explained in the section "IOCP Buffer Requirements" on page 223.

If your system includes disk queuing of messages, add the space required for file control blocks (FCBs). An FCB contains information about a station that has a disk queue. Its size, including the SRB, is 48 bytes. $.CFD splits its dynamic storage into two pools, the message buffer pool and a pool for FCBs. As

distributed, it allows space for ten FCBs. Use utility program $.SETSTG, described in the *Operator's Guide*, to alter the size of either pool.

If your system does not include disk queuing of messages, use the EDX utility $DISKUT2 to set $.CFS's dynamic storage to the size you require for the message buffer pool.

Performance is degraded if the message buffer pool is too small, because Communications Facility programs that send messages wait until space is available if the pool is full. When the Communications Facility is running, you can use the CP Q CFBUF command to monitor usage of the message buffer pool and to set a warning level. You can also monitor usage of the message buffer pool with the DB command of the diagnostic aid utility ($.UT1). The command displays information about each element in the pool, including free space. The free space displayed is not the total free space. Since fragmentation of storage may occur, this cannot be used as a reliable indicator of remaining space. $.UT1 and the CP commands are described in the *Operator's Guide*.

IOCP Buffer Requirements

Each IOCP has a buffer pool from which it obtains buffers to process messages for the stations that it manages. This section explains how to determine the buffer size required for stations and the buffer pool size required for each IOCP.

You specify the buffer size for a device, line, or SNA physical unit station when you define it to the Communications Facility, using either the CP Define command or the $.CONFIG utility program. The size required varies according to the station type, as described in the following sections.

The buffer pools are, in most cases, the dynamic storage of the IOCPs. Figure 39 on page 216 shows their sizes, as distributed. When you calculate the size of a buffer pool, remember to allow space for the BRB, SRBs, and buffer headers, as explained under "Communications Facility Working Storage" on page 219. The buffer and work area sizes given in this section don't include the SRB or buffer header.

$.IO0AA8, PC Connect

When planning $.IO0AA8's storage requirements, consider these three factors:

- The amount of dynamic storage.
- The amount of mapped storage required by the IOCP to perform the host connection function.
- The amount of available mapped or unmapped processor storage for the host connection message buffers.

First, $.IO0AA8 uses its dynamic storage as its buffer pool. It requires a 578-byte work area and a line buffer to receive messages from the PC Connect for each active line. The size of the line buffer required is 20 bytes plus the buffer size specified for the PCC line station. The IOCP acquires the work area and the buffer when a line station is started and releases them when the station is stopped or halted.

$.IO0AA8 also requires a second line buffer to transmit messages. It acquires this transmit buffer when the need arises and releases it after the operation is completed. The IOCP requires space in its buffer pool for only one transmit buffer, even when it manages multiple lines. For improved performance, you should allow space for a transmit buffer for each active line.

The line buffer must be as large as the largest message transmitted between the Series/1 and any gateway PC connected to the Series/1. Allow 20 bytes for the message header. The buffer size for the Series/1 and the gateway PC communicating with each other should be the same. If the buffer size is not the same, the smaller size governs the amount of data that can be exchanged. The gateway PC buffer size is 33170 bytes (8192 hexadecimal).

When performing host connection function, $.IO0AA8 also acquires an 18-byte segmentation register table entry for each 2K of mapped or unmapped storage obtained. This 18-byte table entry contains the segmentation register value and other information associated with the 2K of mapped or unmapped storage. It is acquired or released at the same time as the 2K of mapped or unmapped storage.

Next, $.IO0AA8 requires a 2K block of mapped storage in the same partition as the IOCP. This block of storage is obtained when performing host connection function, and is released when the function is complete.

Finally, $.IO0AA8 keeps a message buffer in 2K of mapped or unmapped storage for each PC communicating with the host. This storage is acquired when the first segment message is sent from the PC to the host. It is released when the last segment message is sent.

The number of available 2K blocks of mapped or unmapped storage in your system must be equal to or greater than the number of PCs communicating with the host at the same time. The number of available unmapped storage blocks is reported by the system when you IPL. Assuming no other program in the system uses unmapped storage, this number may be sufficient for communicating with the host. If 2K blocks of unmapped storage is insufficient, the IOCP will try to obtain mapped storage starting in the highest partition. It will not obtain mapped storage in the same partition as $.CFS (or $.CFD).

The number of available mapped storage blocks is based on the amount of physical storage, the number of programs executing and their use of storage.

The maximum number of available mapped storage blocks is controlled by the MAXPROG value on the EDX SYSTEM statement. See the EDX *Installation and System Generation Guide*, for an explanation of the SYSTEM statement. If the MAXPROG value is insufficient, increase it so that more entries are available in the partition you choose.

$.IO0AB0, Local Communications Controller

The line buffer must be as large as the largest message transmitted between the local node and any remote node on the ring. Allow 20 bytes for the message header. Nodes that communicate with each other should have line buffers of the same size. When they don't, the smaller size governs the amount of data they can exchange.

When message traffic between two nodes is heavy, you can improve performance by defining line buffers large enough to hold several messages. $.IO0AB0 blocks messages when it can; it transmits messages when either the buffer is full or there are no more messages on a node station's queue.

$.IO0AB0 uses its dynamic storage as its buffer pool. It requires a line buffer, which is used to receive messages from the ring, and a 642-byte work area for each active line. The IOCP acquires the buffer and work area when a line station is started, and releases them when the station is stopped or halted.

$.IO0AB0 also requires a second line buffer, which it uses to transmit messages. It acquires this transmit buffer when the need arises and releases it when the operation is complete. The IOCP requires space in its buffer pool for only one transmit buffer, even when it manages multiple lines. For optimum performance, you should allow space for a transmit buffer for each active line.

You can also allow additional space to be used for recovery from errors. If a block of messages cannot be sent to a remote mode, the IOCP must save the block so that it can continue sending messages to other nodes attached to the Local Communications Controller. It does this only if there is enough space in the pool for an additional buffer for saving the block. If there isn't, it discards the block and logs message IO47, warning that messages may have been lost.

$.IO0AB8, X.25

$.IO0AB8 uses the line buffer size to acquire read and write buffers for DTE and DCE lines. The line buffer must be as large as the largest message sent or received by any circuit on the line. Allow additional bytes for the Communications Facility/X.25 data or control message headers. At present, these headers are 2 bytes long. If any circuits on the line have a usage type of CF, allow another 24 bytes for the Communications Facility message header.

When the line buffer size is larger than the line packet size, messages can be sent and received that are several packets long. The line's read task must wait for all packets in the message to arrive before another read can be issued. The arrival of a complete packet sequence can be delayed indefinitely because Recommendation X.25 does not specify a maximum time delay between packets in a sequence. This means the line's read task should wait indefinitely. For this reason, the line buffer size equal the line packet size when possible.

$.IO0AB8 uses its dynamic storage as a buffer pool and as a storage pool for station work areas. It requires a read buffer and a 1240-byte work area for each active line station. It requires a 304-byte work area for each active circuit station. This storage is acquired when the station is started and released when the station is stopped or halted.

$.IO0AB8 uses the rest of its storage area more dynamically. It acquires a write buffer to receive a message from a circuit station's queue and releases the buffer when the message is successfully written to XHCS. If the message is an X.25 control message, it frees the buffer when the XHCS control instruction completes successfully and the confirmation control message is sent back to the originating station, if required. The IOCP acquires a temporary 304-byte work area to open a circuit and releases the storage when the open has successfully completed. If message traffic is heavy on all the line's circuits, allow space for one line buffer per circuit. Otherwise, fewer line buffers are needed.

To reduce fragmentation of its dynamic storage area, $.IO0AB8 acquires storage in multiples of 312 bytes, including the SRB and buffer header. When you calculate how much space to allow for read and write buffers, add 18 bytes to the line buffer size for the SRB and buffer headers, and round the result to a multiple of 312. $.IO0AB8 only uses the line buffer size portion of the storage, even if more is acquired.

$.IO0AC0, 3270 Control Line

For 3270 control, the line buffer must be as large as the largest message for any of the terminal stations associated with the line. Allow 13 bytes for BSC and 3270 control data that may be appended to a message.

$.IO0AC0 uses its dynamic storage as its buffer pool. It acquires a line buffer to receive input from a terminal and releases the buffer when the operation is complete. It acquires a line buffer to send output to a terminal and releases the buffer when there is no more pending output for that terminal.

The IOCP requires space in its buffer pool for only one line buffer, even when it manages multiple lines. However, for optimal performance, you should allow space for one line buffer for each active line, especially when a line is used for communication between Series/1s. During communication between Series/1s, many messages may accumulate on a terminal's queue, and space in the buffer pool is not released until all the messages are sent. If you have multiple lines with different buffer sizes, you should allow extra space in the buffer pool for fragmentation.

$.IO0AD0, Channel Attach

For channel attach, the line buffer must be as large as the largest message for any of the terminal stations (ports) associated with the line. Allow 4 bytes for BSC control characters that may be appended to a message.

$.IO0AD0 uses its dynamic storage as its buffer pool. It requires a line buffer and a 300-byte work area for each active line. It acquires the buffer and work area when a line station is started, and releases them when the line station is stopped or halted.

$.IO0AE0, 3270 Emulation Line

For 3270 emulation, the line buffer must be as large as the largest message for any of the terminal stations associated with the line. Allow 7 bytes for BSC and 3270 control characters that may be appended to a message.

$.IO0AE0 uses its dynamic storage as its buffer pool. It acquires a line buffer to receive a message from the host and releases the buffer when the operation is complete. It acquires a line buffer to send messages to the host and releases the buffer when there is no more pending output from any terminal.

The IOCP requires space in its buffer pool for only one line buffer, even when it manages multiple lines. However, for optimal performance, allow space for one line buffer for each active line. If you do not, you will probably get time-outs when polled by the host. It is especially important that you allow space for one buffer per line when the line is used for communication between Series/1s or to transfer large messages to a host. In these cases, many messages may accumulate on the terminals' queues, and space in the buffer pool is not released until all the messages are sent. If you have multiple lines with different buffer sizes, you should allow extra space in the buffer pool for fragmentation.

$.IO0A10, Series/1-to-Series/1 Line

For Series/1-to-Series/1, the line buffer must be as large as the largest message transmitted between the two Series/1s, including 24 bytes for the message header.

$.IO0A10 does not have its own buffer pool. It uses the message buffer pool, CFBUF. It requires a line buffer for each active line. It acquires the line buffer when a line station is started, and releases the buffer when the line station is stopped or halted.

$.IO0A20, Point-to-Point Line

For a point-to-point line, the line buffer must be as large as the largest message transmitted between the Series/1 and the system to which it is connected. Allow 2 bytes for BSC control characters that may be appended to a message.

$.IO0A20 uses its dynamic storage as its buffer pool. It requires two line buffers for each active line. It acquires the buffers when a line station is started, and releases them when it is stopped or halted.

$.IO06F0, 3101 Device

When planning $.IO06F0's storage requirements, consider these three factors:

- The amount of mapped storage required by the IOCP itself regardless of the number of supported devices.
- The amount of dynamic storage.
- The amount of available mapped and/or unmapped processor storage for screen image workspaces.

First, $.IO06F0 requires a 6K block of mapped storage in the same partition as the IOCP. This block of storage creates the window to view device screen images in unmapped (or other mapped) storage.

Next, $.IO06F0 uses its dynamic storage as a pool to obtain a 232-byte work area for a task control block (TCB) and workspace for each started station. The work area is obtained when the station is started and is not released until the IOCP is halted. As shipped, $.IO06F0's dynamic storage area is 5120 bytes, which supports 16 3101s. To support more than 16 3101s, increase the dynamic storage of $.IO06F0 and, if necessary, provide more mapped storage entries as described below.

$.IO06F0 also acquires a 220-byte work area from its dynamic storage, if space is available, during output to a device. If no space is available, it can perform output to only one device at a time. For optimum performance, you should allow space for one of these work areas for every three to five devices.

Finally, $.IO06F0 keeps a screen image workspace in mapped and/or unmapped storage for each device. The size of the screen image workspace depends on the maximum number of fields on any screen image sent to the device. The size of the workspace must be 1990 bytes plus 2 bytes per field. In an unprotected field, two consecutive unprotected fields separated by one or more spaces are counted as three fields. The minimum size is 2048 bytes, which allows for 29 fields. The largest size you ever need specify is 3910 bytes, which allows for 960 fields. Estimate the amount of storage required by adding the screen image sizes for each active station. Divide the sum by 2K and subtract two to get the number of 2K blocks of of unmapped and/or mapped storage required to satisfy your screen image workspace requirements. (Approximately two workspaces will be obtained from the original 6K block.)

The number of available 2K blocks of unmapped and/or mapped storage in your system must be equal to or greater than the screen image workspace requirement to start all the devices concurrently. The number of available unmapped storage blocks is reported by the system as part of the IPL messages. Assuming no other program in the system uses unmapped storage, this number may be sufficient. If this number is insufficient, the IOCP will try to obtain mapped storage starting in the highest partition. It will not obtain mapped storage in the same partition as $.CFS (or $.CFD).

The number of available mapped storage blocks is based on the amount of physical storage, the number of currently executing programs and their storage usage.

The maximum number of available mapped storage blocks is controlled by the MAXPROG value on the EDX SYSTEM statement. (See the EDX *Installation and System Generation Guide*, for an explanation of the SYSTEM statement.) If the value on MAXPROG is insufficient, increase it so that more entries are available in the partition you choose.

Each of these IOCPs also has a 3100-byte buffer, defined in module S$IO0634, which is shared by all active devices. This buffer must be as large as the largest message sent to a device. If you need to change the size of this buffer, modify and assemble S$IO0634 and link it with the other modules that form the IOCP.

$.IO0670, 4978/4980 Data Stream Devices

For the 4978/4980 data stream IOCP, the device buffer must be as large as the largest message sent to or received from the device. The minimum size allowed is 2048 bytes.

$.IO0670 uses its dynamic storage as its buffer pool. The space in the pool is shared by all active devices. Allow space for one device buffer for every three or four active devices.

$.IO0670 also acquires a 176-byte work area from dynamic storage as each device station is started. This area is used for the device TCB and work area, and is released when the device station is stopped or halted. Allow one work area for each active device station.

$.IO0670 also uses the system storage pool, S$POOL. It acquires a 48-byte work area to process a message. Allow space for the same number of work areas as device buffers in the dynamic storage pool.

$.IO0674, Printer Devices

For the printer IOCP, the device buffer must be as large as the largest message sent to the printer. The minimum size allowed is 2048 bytes.

$.IO0674 uses its dynamic storage as its buffer pool. It acquires two buffers from the pool to process a message—a device buffer and a work buffer. For 3270 mode printer stations, the work buffer is 2054 bytes. For SCS mode printer stations, the work buffer size is the device buffer size plus 134 bytes. If you have only one printer station, allow space in the pool for one of each of these buffers. If you have more than one printer station and they are very busy, allow space for one 2054-byte buffer per printer and two device buffers. The device buffer size is specified when the printer device station is defined.

$.IO0674 also requires a 40-byte work area for each active printer station in SCS mode. It acquires the work area from the dynamic storage pool when the station is started and releases it when the station is stopped or halted.

$.IO0680, 7485 Devices

When planning $.IO0680's storage requirements, consider these three factors:

- The amount of mapped storage required by the IOCP itself regardless of the number of supported devices.
- The amount of dynamic storage.
- The amount of available mapped and/or unmapped processor storage for screen image workspaces.

First, $.IO0680 requires a 6K block of mapped storage in the same partition as the IOCP. This block of storage creates the window to view device screen images in unmapped (or other mapped) storage.

Next, $.IO0680 uses its dynamic storage as a pool to obtain a 232-byte work area for a task control block (TCB) and workspace for each started station. The work area is obtained when the station is started and is not released until the IOCP is halted. As shipped, $.IO0680's dynamic storage area is 5632 bytes, which supports 16 7485s. To support more than 16 7485s, increase the dynamic storage of $.IO0680 and, if necessary, provide more mapped storage entries as described below.

$.IO0680 also acquires a 356-byte work area from its dynamic storage, if space is available, during output to a device. If no space is available, it can perform output to only one device at a time. For optimum performance, you should allow space for one of these work areas for every three to five devices.

Finally, $.IO0680 keeps a screen image workspace in mapped and/or unmapped storage for each device. The size of the screen image workspace depends on the maximum number of fields on any screen image sent to the device. The size of the workspace must be 1990 bytes plus 2 bytes per field. In an unprotected field, two consecutive unprotected fields separated by one or more space is counted as three fields. The minimum size is 2048 bytes, which allows for 29 fields. The largest size you ever need specify is 3910 bytes, which allows for 960 fields. Estimate the amount of storage required by adding the screen image sizes for each active station. Divide the sum by 2K and subtract two to get the number of 2K blocks of unmapped and/or mapped storage required to satisfy your screen image workspace requirements. (Approximately two workspaces will be obtained from the original 6K block.)

The number of available 2K blocks of unmapped and/or mapped storage in your system must be equal to or greater than the screen image workspace requirement to start all the devices concurrently. The number of available unmapped storage blocks is reported by the system as part of the IPL messages. Assuming no other program in the system uses unmapped storage, this number may be sufficient. If this number is insufficient, the IOCP will try to obtain mapped storage starting in the highest partition. It will not obtain mapped storage in the same partition as $.CFS (or $.CFD).

The number of available mapped storage blocks is based on the amount of physical storage, the number of currently executing programs and their storage usage.

The maximum number of available mapped storage blocks is controlled by the MAXPROG value on the EDX SYSTEM statement. (See the EDX *Installation and System Generation Guide*, for an explanation of the SYSTEM statement.) If the value on MAXPROG is insufficient, increase it so that more entries are available in the partition you choose.

Each of these IOCPs also has a 3100-byte buffer, defined in module S$IO0634, which is shared by all active devices. This buffer must be as large as the largest message sent to a device. If you need to change the size of this buffer, modify and assemble S$IO0634 and link it with the other modules that form the IOCP.

$.IO1CC8, PC Connect Disk-Server

$.IO1CC8 uses its dynamic storage as a buffer pool. Each volume station requires a buffer pool large enough to contain the following:

- 4106 bytes for the send/receive buffer
- 416 bytes for the directory table
- 230 bytes for the task control block and the work area
- 74 bytes data set control block (DSCB) for each emulated PC disk.

$.IO1CC8's dynamic storage, as distributed, can support a volume station with five emulated PC disks. When you calculate the size of a buffer pool, remember to allow space for a BRB and an SRB for each element.

$.IO14E8, SNA Connection

The SNA physical unit buffer must be as large as the largest message for any of the SNA LU stations it controls. Allow 4 bytes for BSC control characters that may be appended to a message.

$.IO14E8 splits its dynamic storage into two pools. One pool is used during session starts and is 1120 bytes large. The remaining space is an I/O pool shared by all active SNA LU stations. Each LU acquires and releases buffers from this I/O pool. The buffer size each LU station uses for I/O is the buffer size you define for its controlling PU station. Allow space for one I/O buffer for every three or four active SNA LU stations.

For the EDX Secondary SNA2 support of the $.IO14E8 IOCP, each active LU station requires a 176-byte workspace for its TCB and work area. This workspace is acquired from the I/O pool when the LU station is started and is used until the station is stopped.

Other Dynamic Storage Requirements

This section describes the dynamic storage requirements of Communications Facility programs other than the control program and IOCPs. It explains what the dynamic storage is used for and why you may need to alter its size. The dynamic storage sizes, as distributed, are shown in Figure 39 on page 216.

$.CPCMCF, Communications Manager Command Processor

This program uses dynamic storage for responses to a Communications Manager DISPLAY command. The dynamic storage, as distributed, is 512 bytes; there is no reason to make it larger.

$.HMU and $.RMU, Remote Management Utilities

These programs use dynamic storage as a buffer for messages exchanged between the host and remote cells. The maximum message size is the smallest of $.HMU dynamic storage, $.RMU dynamic storage, and the $.PD transaction buffer in each cell. For data transfer operations, records are blocked if the maximum size allows it. See the discussion of record blocking and buffer size in the Remote Management Utility section of the EDX *Communications Guide.*

$.PANEL, Panel Design Aid

This program uses dynamic storage for work areas—one 116-byte area for each active user of the program. As distributed, the program can communicate with four users at a time.

$.PNLUT1, Panel Utility

This program uses dynamic storage as a panel buffer. The storage required is the size of the largest panel to be processed, plus 10 bytes. As distributed, the program can process 9-record panels.

$.PD, Program Dispatcher

The program dispatcher uses dynamic storage for some common data, the path table, the transaction table, a work buffer, and a transaction buffer. The common data requires 16 bytes. The path table requires 24 bytes per entry, including reserved entries, plus 2 bytes for an end-of-table entry. The transaction table requires 48 bytes per entry, including reserved entries, plus 2 bytes for an end-of-table entry. The remaining dynamic storage is the work buffer, and must be at least 410 bytes. The transaction buffer overlays part of the work buffer, and is approximately 100 bytes smaller.

The size of the transaction buffer governs the maximum size of a transaction that the program dispatcher can send or receive. The size appears as the XCODE field of message PD01, which is logged when the program dispatcher is started. The CP Q PARM command also displays this size, in the WORK column of the line for $.PD.

Here is an example calculation, using 1280 bytes of dynamic storage (as distributed), three path definitions, and six transaction definitions:

Common data	16
Path table	74
Transaction table	290
Total	380

Work buffer = 1280 - 380 = 900 bytes

Transaction buffer = approximately 800 bytes

$.PDSCHD, Transaction Scheduler

The transaction scheduler uses dynamic storage for information about transactions scheduled for later execution. The storage required is 32 bytes per entry. As distributed, the program can manage eight scheduled transactions.

$.UT2, Disk-Queued Message Utility

This program uses dynamic storage as a message buffer. A message larger than dynamic storage can be reported or purged, but it cannot be sent on.

$.WSC, Work Session Controller

The work session controller uses dynamic storage for a 2076-byte transaction buffer and a 132-byte work buffer. The minimum size, and the size as distributed, is 2304 bytes. There is no reason to make it larger.

Program Loading and Partition Use

The Communications Facility manages the loading of system programs and user application programs. Several facilities are available for performing the load function: the command processor start command, the program dispatcher high-speed loader, and the EDX $L command.

CP Start Command

A system or user application program is loaded when a CP Start command for a user station is issued.

An IOCP is loaded when the first CP Start command for a station managed by that program is issued.

The Start command processor uses the EDX transient loader ($LOADER) to load programs. When no partition is specified on the CP Start command, the Start command processor attempts to load programs as follows:

1. First it attempts to load the program into the partition just below the $.CF partition. If that fails or the partition is not mapped,

2. It attempts to load the program into any mapped partition, starting with partition 1. If that fails,

3. It logs an error message and does not start the station.

This process does not necessarily result in the most efficient use of partition storage. If you want more control over which programs are loaded into which partitions, several alternatives are available:

- Specify the mapped partition into which the program is to be loaded on the CP Start command. The Start command processor attempts to load the program into the specified partition. If that fails or if the partition is not mapped, it attempts to load the program into any mapped partition, starting with partition 1.

- Start an IOCP yourself, instead of letting it be loaded when you start a station of the type it manages. Define the IOCP as a user station and issue a Start command for it, specifying the mapped partition into which it is to be loaded.

- Start your stations in a planned sequence, so that the smallest program is loaded into a partition first, leaving room for $LOADER to load larger programs. For example, assume the remaining space in the partition just below the $.CF partition is 16K. Two stations need to be started. One is a 4K user program (smaller than $LOADER). The other is a 4978 device, which causes the $.IO0670 IOCP to be loaded. This IOCP is approximately 12K in size. If the 4978 is started first by a CP Start command, only 4K is left in that partition, which is not enough for $LOADER. But if the user station is started first, there is enough storage left for $LOADER to load the IOCP when the 4978 device is started.

If a program must run in an EDX static partition, you have two alternatives:

- You can specify S as the partition number. The start command processor attempts to load the program using the normal algorithm with the additional constraint that the partition must be static.

- You can specify the static partition into which you want to load your program.

It is important to note that the start command processor automatically loads the IOCPs which require static partitions ($.IO0AD0, $.IO0680, and $.IO06F0), into static partitions.

Program Dispatcher

The program dispatcher uses two methods to load transaction-processing programs:

1. It uses the CP Start command if the transaction-processing program is defined as a station in $.SYSNET. If the station type is user, loading is under control of the CP Start command processor, as already described.

2. Otherwise, it uses the high-speed loader ($.HSL).

You can specify in the transaction definition statement that the program is to be loaded into:

- A specific mapped partition
- Any mapped partition except a specific partition
- The $.CF partition
- Any mapped partition except the $.CF partition
- Any mapped partition (the default)

The program dispatcher validates specified partitions when it builds its transaction table during initialization. If a specified partition does not exist or is not mapped, the specification is changed to any mapped partition.

Be careful about specifying the $.CF partition or "any partition" for programs that will be loaded by the high-speed loader. The programs may use up the space in the $.CF partition that is required for command processor transients.

EDX $L Command

You can use the $L command to load some Communications Facility programs into the mapped partition you want. Most of the programs create their own station block during initialization if it doesn't already exist. Those that don't create their own station block terminate with an error message when you try to load them. Don't use this method when you have assigned a disk queue to a program's user station. It is the CP Start command processor that opens a station's disk queue.

If you load a Communications Facility program into an unmapped partition, a program check occurs when a Communications Facility instruction is executed. The PSW is 0802 (invalid function), and hardware register 1 contains the address of the Communications Facility instruction.

Coding Supervisor Definition Instructions

During EDX system generation, you need to define certain elements of your Communications Facility configuration to the EDX supervisor. Examples of how to use these instructions during the installation process are in the chapter "Installing the Communications Facility" on page 251. This chapter gives the details of the format and operands of these Communications Facility language extension instructions.

First, here's an explanation of the syntax notation that's used to describe the instructions in this chapter.

Syntax Notation

Each instruction format in this chapter appears in a box with three columns:

Name	Operation	Operand

Name Field. This field contains a symbolic label of up to 8 characters.

Operation Field. This field contains the instruction name.

Operand Field. This field contains the operands associated with the instruction.

The following conventions are used within the format descriptions:

- Words in **BOLD CAPITAL** letters must be coded exactly as shown. Commas, parentheses, and equal signs must also be coded exactly as shown.
- Values in **BOLD UNDERSCORED** letters are defaults.
- Words in *italics* are symbols for which you must substitute actual values.
- Brackets ([ ]) indicate that the operand is optional.
- Braces ({ }) indicate a group of mutually exclusive operands or values, of which you can code only one.
- A vertical bar (|) separates the mutually exclusive items within braces.

Instruction Format

The syntax of the language extensions is very similar to that of the other EDL instructions. For information about EDL syntax rules, see the EDX *Language Reference* manual.

The general format of the Communications Facility language extension instructions presented in this chapter is:

label **VERB MODIFIER,KEY1**=*value1*,**KEY2**=*value2*,. . .

where:

label
is the label to be given to the first word of the generated instruction. It is required only if other instructions will refer to this instruction.

VERB
is the name of the general operation to be performed (such as DEFINE).

MODIFIER
is an additional word (such as VOLUME) that further defines what the instruction is to do.

The modifier names as given in this chapter are the minimum names you can code to specify the instructions. You can code additional letters beyond those required if you want. For example, DEFINE BRB is the required form of the instruction that defines a buffer reference block, but you can code DEFINE BRBLOCK if you want.

KEY1, KEY2,. . .
are keyword operands, for which you supply values.

A space is required after the verb. No spaces are permitted between the modifier and the first operand, or between operands.

Operand Formats

The description of each operand in this chapter includes a notation of its valid syntax. The possibilities, for the instructions in this chapter, are:

term
A self-defining term. Self-defining terms are decimal constants such as the bufferpool size 4000 or hexadecimal constants such as the polling address X'C240'.

literal
A specific value, specified in the syntax of the instruction that requires its use. For example, a literal may be YES, NO, or LOCAL. A literal may also be alphanumeric such as the hexadecimal address 3C or the decimal number of error retries—8.

label
A symbolic name to represent the generated instruction.

Supervisor Definition Instructions

The rest of this chapter shows the format of each supervisor definition instruction, explains its operands, and gives an example of its use.

DEFINE BUFFERPOOL—Define the System Storage Pool and DCB Pool

This instruction defines the system storage pool (S$POOL) and the device control block pool (S$CFDCBS). (See "Planning Storage Requirements" on page 213 for a description of S$POOL and S$CFDCBS, and an explanation of how to calculate their sizes.) Note that you can code DEFINE WORK or DEFINE POOL instead of DEFINE BUFFERPOOL.

An alternative to the DEFINE BUFFERPOOL instruction is the DEFINE BRB instruction, which you can use to reduce the size of the supervisor.

DEFINE BUFFERPOOL Format

[*label*]	**DEFINE BUFFERPOOL** **DEFINE WORK** **DEFINE POOL**	**,SIZE=***size*

DEFINE BUFFERPOOL Operands

Operand	Syntax	Description
label	label	The label to be assigned to the pool. The default label is S$POOL. Code S$CFDCBS when defining the DCB pool.
SIZE=	term	The decimal size, in bytes, of the storage pool to be allocated. The maximum size is 32,767. If you specify 0 or a negative value, only a buffer reference block (BRB) is created; its size field contains 0 or the negative value.

DEFINE BUFFERPOOL Examples

```
DEFINE BUFFERPOOL,SIZE=4000
```

This instruction creates a system storage pool (S$POOL), of 4000 bytes.

```
S$CFDCBS   DEFINE BUFFERPOOL,SIZE=200
```

This instruction creates a DCB pool (S$CFDCBS) of 200 bytes.

DEFINE BRB—Define a Buffer Reference Block

Instead of using the DEFINE BUFFERPOOL instruction to create the system storage pool and the DCB pool, you may use the DEFINE BRB instruction to create a control block, called a *buffer reference block*, that defines the beginning of S$POOL or S$CFDCBS. Modules that will not be needed in storage after EDX system initialization may be located in the area following the buffer reference block. After initialization, this area becomes S$POOL or S$CFDCBS.

DEFINE BRB Format

[*label*]	**DEFINE BRB**	**,SIZE=***size*

DEFINE BRB Operands

Operand	Syntax	Description
label	label	The label to be assigned to the buffer reference block. The default label is S$POOL. Code S$CFDCBS when defining the DCB pool.
SIZE=	term	The decimal size, in bytes, of the storage area to be allocated (not including the 8 bytes for the BRB).

DEFINE BRB Example

```
DEFINE BRB,SIZE=3816
```

creates this buffer reference block:

Address	Contents	
1100[14]	1108	*Pointer to first word after BRB*
1102	0000 0000	*2 full words of 0*
1106	0EE8	SIZE
1108		*First word after BRB*

14 Example address only.

DEFINE DEVICE—Define Remote Disk Access

DEFINE DEVICE defines the information used to access one or more disks on another Series/1 as if they were on this Series/1. It is used in conjunction with one or more DISK statements that define remote disks. DEFINE DEVICE generates entry point and label $CFPDDB; only one DEFINE DEVICE per assembly is allowed.

The information supplied in DEFINE DEVICE is used to access remote disks only when the program dispatcher is not running in this Series/1. When the program dispatcher is running in this Series/1, access to remote disks is through the paths defined to the program dispatcher. DEFINE DEVICE is required even if remote disks are not accessed until the program dispatcher is running in this Series/1.

DEFINE DEVICE Format

	DEFINE DEVICE	**,{CUDA=***polladdr* **|** **RINGADR=***ringaddr***}** **,LINE=***lineaddr* **,CELL=***cellid* **,ADDRESS=***diskaddr*

DEFINE DEVICE Operands

Operand	Syntax	Description
CUDA=	term	A 3270 polling address, in the form of a fullword hexadecimal constant, or 0. When access to remote disks is over a multipoint BSC line specify the polling address of the terminal station that represents the path to the program dispatcher in the Series/1 where the real disks are located. When access to remote disks is over a point-to-point BSC line (Series/1-to-Series/1), specify 0.
RINGADR=	literal	The hexadecimal Local Communications Controller ring address of the Series/1 where the real disks are located.
LINE=	literal	The hexadecimal address of the line in this Series/1 that is used to access the remote disks—either a BSC line or subchannel 0 of a Local Communications Controller attachment.

Operand	Syntax	Description
CELL=	literal	The cell ID of this Series/1.
ADDRESS=	literal	The hexadecimal device address of the remote disk. This can also be a list of addresses in parentheses (for example, ADDRESS=(E1,E2,E3)). The address is the one specified on the DISK statement that defines the remote disk.

DEFINE DEVICE Examples

```
DISK DEVICE=4962-3,ADDRESS=E1,VOLNAME=(EDX003)
DEFINE DEVICE,CUDA=X'C240',LINE=9,CELL=S2,ADDRESS=E1
```

The DISK statement defines a remote disk with address E1. The device type must be the same as that of the corresponding real disk. The device address must be a fictitious address; there must be no real disk or other device attached at that address.

The DEFINE DEVICE instruction specifies that when the program dispatcher is not running in this Series/1, the corresponding real disk is accessed by means of a transaction sent from cell S2 on BSC line 9 using the polling address X'C240'. When the program dispatcher is running in this Series/1, the corresponding real disk is accessed by means of a transaction sent with the cell ID of the program dispatcher to the emulated terminal station with polling address X'C240'.

```
DEFINE DEVICE,RINGADR=4F,LINE=50,CELL=S2,ADDRESS=E1
```

This instruction specifies that when the program dispatcher is not running in this Series/1, the real disk corresponding to the one with address E1 is accessed by means of a transaction sent from cell S2 over the Local Communications Controller attachment at address 50 to the Series/1 with ring address 4F.

DEFINE VOLUME—Define a Remote Disk Volume

DEFINE VOLUME makes an entry for a remote disk volume in the volume descriptor cross-reference table. The first DEFINE VOLUME statement generates entry point and label $CFPVDE.

DEFINE VOLUME Format

	DEFINE VOLUME	**,CELL=***cellid* **,VOLNAME=(***local-name,remote-name***)** **[END={YES | NO}]**

DEFINE VOLUME Operands

Operand	Syntax	Description
CELL=	literal	The cell ID of the Series/1 where the real disk volume is located.
VOLNAME=	literal	The volume name used in this Series/1, followed by the real volume name.
END=	literal	YES if this is the end of the volume descriptor cross-reference table; NO otherwise.

DEFINE VOLUME Examples

```
DEFINE VOLUME,CELL=SJ,VOLNAME=(PSD002,EDX002)
DEFINE VOLUME,CELL=SJ,VOLNAME=(EDX003,EDX003),END=YES
```

These instructions define remote volumes located in cell SJ. The first one defines volume EDX002, which is known as PSD002 in this Series/1. The second one defines the volume known as EDX003 both in this Series/1 and in cell SJ.

The remote volumes and the names by which they are known in this Series/1 must be designated as performance volumes (specified in the VOLNAME operand of a DISK statement).

PCC—Define PC Connect

A PCC statement defines a Series/1-PC Connect Attachment subchannel to the EDX supervisor. All PCC instructions must appear together in the $EDXDEF data set, and must be grouped with LCC instructions. Each Series/1-PC Connect Attachment has three subchannels; all three subchannels must be defined for each Series/1-PC Connect Attachment.

PCC Format

	PCC	**ADDRESS=***chanaddr* **[,END=YES]**

PCC Operands

Operand	Syntax	Description
ADDRESS=	literal	The hexadecimal address of the Series/1-PC Connect Attachment subchannel.
END=	literal	This operand must be included in the last PCC instruction.

PCC Examples

```
PCC ADDRESS=B0
PCC ADDRESS=B1
PCC ADDRESS=B2,END=YES
```

The PCC instructions in this example define a PC Connect starting at address X'B0'.

CFTERM—Define Non-EDX 3101 or 7485 on Multifunction Attachment

CFTERM defines a 3101 or 7485 terminal, on a multifunction attachment, that is intended for Communications Facility usage only; it is not known to the EDX supervisor. One or more of these instructions are used in conjunction with the ADAPTER statement that defines the multifunction attachment.

CFTERM Format

label	**CFTERM**	**ADDRESS=***addr* **,LMODE={RS422 | LOCAL | SWITCHED | <u>PTTOPT</u>}** **[,PRTS=ON | <u>OFF</u>]**

CFTERM Operands

Operand	Syntax	Description
label	label	The name that identifies this terminal in the ADAPTER statement.
ADDRESS=	literal	The hexadecimal address of the multifunction attachment port to which the 3101 or 7485 is connected.
LMODE=	literal	The type of connection.
	RS422	For a terminal directly attached to any port of the multifunction attachment. For a 7485, you must define LMODE=RS422.
	LOCAL	For a terminal directly attached. The terminal must be attached on the base address only.
	SWITCHED	For a point-to-point switched connection. The terminal must be attached on the base address only.
	PTTOPT	For a point-to-point nonswitched connection. The terminal must be attached on the base address only.
PRTS=	literal	A permanent request to the send option. It applies to the base address of the multifunction attachment.

Operand	Syntax	Description
	ON	To set a permanent request at IPL time to the send function.
	OFF	To turn off the permanent request to send bit at IPL time.

Special Considerations

To turn on the permanent request to send, you must explicitly specify PRTS=ON. For example, you can specify either: LMODE=SWITCHED,PRTS=ON or LMODE=LOCAL,PRTS=ON.

CFTERM Examples

```
MFA3101 ADAPTER ADDRESS=58,TYPE=MFA,              CC
                DEVICES=(T3101A,T3101B,T3101C),  CC
                END=YES
T3101A  CFTERM  ADDRESS=58,LMODE=RS422
T3101B  CFTERM  ADDRESS=59,LMODE=RS422
T3101C  CFTERM  ADDRESS=5A,LMODE=RS422
```

These statements define a multifunction attachment with 3101 terminals attached to three ports.

Installing the Communications Facility

This chapter is a guide to installing the Communications Facility. It describes the machine-readable material you receive when you order the Communications Facility; explains what you need to do before you begin the installation procedure; and explains each step of the installation procedure.

Machine-Readable Material

The EDX Communications Facility Licensed Program, as released from the Information Systems Distribution (ISD), consists of machine-readable material on double-sided diskettes formatted with 256-byte records. These diskettes contain the Communications Facility modules and data sets. During the installation process, you add those modules and data sets to an installed Event Driven Executive system. The program directory that accompanies the diskettes lists the contents of each diskette.

The first five diskettes are basic material: They contain loadable program modules, assembler overlay programs, assembler macros, object modules, and data required to operate the Communications Facility. The remaining diskettes are optional material—they contain the source code for the Communications Facility. The optional material is not required for the installation and execution of the Communications Facility.

Data sets on the distribution diskettes are named according to the following conventions:

$. Data sets with names that begin with '$.' are loadable program modules (executable code) or data sets.

$4980 Data sets with names that begin with $4980 are the control store, image store, and microcode read access memory for 4980 terminals.

$PCC Data sets with names that begin with $PCC are the PC Connect microcode patch data sets.

A. Data sets with names that begin with 'A.' are assembler overlay programs that assemble the Communications Facility instructions for the $EDXASM assembler.

CS Data sets with names that begin with 'CS' are Communications Facility object modules for inclusion in the EDX supervisor.

S$ Data sets with names that begin with 'S$' are the source code for the Communications Facility modules or copy code for inclusion in user application programs.

O$ Data sets with names that begin with 'O$' are object modules for Communications Facility multimodule programs.

The macros, which assemble Communications Facility instructions for the $S1ASM assembler, are named according to their functions within the Communications Facility.

Installation Requirements

Before installing the Communications Facility, you must install EDX, as described in the EDX *Installation and System Generation Guide.* If you intend to use the SNA IOCP, you must also install the EDX Secondary SNA1 Licensed Program (Program Number 5719-SX1) or the EDX Secondary SNA2 Licensed Program (Program Number 5719-XX9). You must define your SNA configuration as described in the *EDX Secondary SNA1 Guide* or the *EDX Secondary SNA2 Guide.*

If you intend to use the X.25 IOCP, you must also install the EDX X.25/HDLC Communications Support (XHCS) licensed program (Program Number 5719-HD2). You must define your X.25 configuration as described in the EDX XHCS *Programming and Operating Reference* and the EDX XHCS *Program Directory.*

If you intend to use the PC Connect IOCP, you must configure your Series/1-PC Connect Attachment PC as described in the IBM *Series/1-PC Connect for the Event Driven Executive Communications Facility*, SC34-0698.

Several EDX utilities are required to install the Communications Facility:

- $DISKUT1 is required to allocate and delete data sets.
- $COPYUT1 is required to transfer the Communications Facility modules and data sets to disk volumes.
- $COPY is required to install Local Communications Controller microcode patches.
- $FSEDIT is required to edit data set modules.
- $INITDSK is required to initialize volumes and IPL text.
- $EDXASM, $EDXLINK, and $XPSLINK are required to assemble and link-edit the supervisor.

The EDX *Operator Commands and Utilities Reference* explains how to use these EDX utilities.

Installation Summary

The steps required to install the Communications Facility are:

1. Copy the Communications Facility basic material to disk.
2. Convert 3101F device station definitions.
3. Allocate data sets for editing, assembling, and link-editing the supervisor.
4. Edit the supervisor definition data set, $EDXDEF.
5. Create the Communications Facility system storage pool (S$POOL) and DCB pool (S$CFDCBS).
6. Edit the supervisor link control data set, $LNKCNTL, adding the EDX definition module, the system storage pool and DCB pool modules, and the required Communications Facility supervisor modules.
7. Edit the assembler ($EDXASM) control data set, $EDXL, adding references to the overlays that process the Communications Facility language extension instructions.
8. Rebuild the supervisor to include the Communications Facility support modules.
9. Rewrite the IPL text to include the new supervisor, and IPL the system.
10. Perform procedures, if necessary, to reduce the size of the supervisor.
11. Initialize data sets for disk queuing, if any are required.
12. Define the Communications Facility stations for your network.
13. Create the program dispatcher data set ($.SYSPD), if the program dispatcher is to be used.
14. Add IPL transactions to data set $.SYSLCC, if remote IPL through the Local Communications Controller is to be used.
15. Create the X.25 data set ($.SYSX25), if you plan to use X.25 switched virtual circuits.
16. Rename the SNA logon panels in data set $.SYSPNL, if you plan to communicate with an 8100 Distributed Processing Executive (DPPX) system.
17. Install EDX Secondary SNA1 or EDX Secondary SNA2 support, as needed.
18. Install LU-1 support, if you plan to use Series/1 printers for communication with SNA type 1 LUs or as PC LAN-shared printers.
19. Create the emulated PC disks, if you plan to use the disk-server.
20. Run the sample program $.CFMENU to verify that your installation is complete.

1. Transfer the Communications Facility Basic Material to Disk

Copy the ISD distribution diskettes to disk volumes, as explained in the program directory that accompanies the diskettes.

If you copied the data sets and load modules to a volume other than the IPL volume, you may need to copy some EDX modules to that volume.

If you are going to use 4978 terminals as Communications Facility terminals, copy:

$4978CS0
$4978IS0

If you are going to use the remote management utility program, copy:

CDRJP
CDROVCP
CDROV1
CDROV2
CDROV3
CDROV4
CDROV5
$DISKUT3

If you are going to use the Series/1-PC Connect Attachment, copy:

$PCCR0*n*

If you copied the Communications Facility data sets and load modules to a volume other than the IPL volume, you may also need to copy some Communications Facility modules to the IPL volume.

If you are going to use 4980 terminals as Communications Facility terminals, copy:

$4980CSA
$4980ISA
$4980ROA

2. Convert 3101F Device Station Definitions

The 3101F device station definitions for earlier releases (prior to Version 2.0) of the Communications Facility cannot be used with this release. If you have such definitions, you must convert them, as explained in the program directory that accompanies the distribution diskettes.

3. Allocate Data Sets

When you installed EDX, you had to create several data sets. Data sets EDITWORK and LINKWORK are probably not big enough to accommodate a supervisor that includes Communications Facility support. Use the EDX utility $DISKUT1 to delete them and reallocate them with the sizes shown below, and to allocate the data sets for the supervisor definition and system storage pool object modules:

```
> $L $DISKUT1

USING VOLUME EDX002

COMMAND (?): DE EDITWORK

COMMAND (?): DE LINKWORK

COMMAND (?): AL EDITWORK 300 D

COMMAND (?): AL LINKWORK 750 D

COMMAND (?): AL $EDXDEFO 200 D

COMMAND (?): AL CSXPOOL 50 D
```

4. Edit the Supervisor Definition Data Set ($EDXDEF)

Use any EDX text editor to modify $EDXDEF, adding statements that define the support required for your Communications Facility configuration. "Coding Supervisor Definition Instructions" on page 235 describes the Communications Facility supervisor definition instructions. The EDX *Installation and System Generation Guide* describes the EDX system configuration statements. See also the chapter "Sample Configurations" on page 269, which shows $EDXDEF data sets for several sample configurations.

EDX system generation requires a $SYSLOG statement in the $EDXDEF data set. Even though you can generate a Communications Facility system that has no terminals, you must include a $SYSLOG statement in $EDXDEF.

Add COMMON and INITMOD parameters to the SYSTEM statement.

The following example of the SYSTEM statement shows a common area based in partition 1:

```
$EDXDEF  CSECT
         SYSTEM  MAXPROG=(10,10,10,10),                    CC
                PARTS=(32,32,32,32),                       CC
                COMMON=(CSXEND,0,CSXEND,CSXEND),           CC
                INITMOD=(CSXINIT)
```

If you are going to support multiple 3101s or 7485s as 3101F stations or if you plan to use host communication with the Series/1-PC Connect Attachment, see "Planning Storage Requirements" on page 213 for a discussion on increasing the values on the MAXPROG parameter.

The COMMON parameter indicates that the supervisor is to be mapped up to the label CSXEND in partitions 1, 3, and 4. Label CSXEND is located in the Communications Facility system tables in module CSXSYS. You must specify CSXEND for each partition in which Communications Facility programs will run.

In the sample SYSTEM statement, Communications Facility programs can't be run in partition 2. The INITMOD parameter indicates that the Communications Facility initialization module CSXINIT will be executed during IPL.

In the following example, the common area is not included in partition 1:

```
$EDXDEF  CSECT
         SYSTEM  MAXPROG=(10,10,10,10),                    CC
                 PARTS=(32,32,32,32),                      CC
                 COMMON=(0,0,CSXEND,CSXEND),COMBASE=3,     CC
                 INITMOD=(CSXINIT)
```

The COMBASE parameter indicates that the common area is based in partition 3. The COMMON parameter indicates that the partitions 3 and 4 will be mapped up to the label CSXEND. Communications Facility programs may run in partitions 3 and 4 only.

Add statements for the support you require, as follows:

4978 Terminals and Printers: Use TERMINAL statements to define these to EDX. Be sure that the terminal name is different from the station name you use when you define the device to the Communications Facility.

4980 Terminals: Use TERMINAL statements to define these to EDX. Be sure that the terminal name is different from the station name you use when you define the device to the Communications Facility. Use the ADAPTER statement to define the SMIO card to which the 4980s are connected.

3101 Terminals, Feature Programmable Communications or Terminal/Host Adapter: If you are going to use these only as emulated 3277 terminals, you need not define them to EDX. If you are also going to use them as EDX terminals or as Communications Facility work session controller terminals, use TERMINAL statements to define them to EDX.

3101 Terminals, Multifunction Attachment: Use the ADAPTER statement to define the multifunction attachment. If you are going to use the terminals only as emulated 3277 terminals, use the Communications Facility CFTERM instruction to define them. For example, to define three 3101s on a multifunction attachment:

```
MFA3101 ADAPTER ADDRESS=58,TYPE=MFA,                    CC
                DEVICES=(T3101A,T3101B,T3101C),         CC
                END=YES
T3101A  CFTERM  ADDRESS=58,LMODE=RS422
T3101B  CFTERM  ADDRESS=59,LMODE=RS422
T3101C  CFTERM  ADDRESS=5A,LMODE=RS422
```

If you are also going to use the terminals as EDX terminals or as Communications Facility work session controller terminals, use TERMINAL statements instead of CFTERM instructions to define them.

7485 Terminals, Feature Programmable Communications or Terminal/Host Adapter: The terminals can be used only as emulated 3277 terminals. Do not define them to EDX.

7485 Terminals, Multifunction Attachment: The terminals can be used only as emulated 3277 terminals. Use the ADAPTER statement to define the multifunction attachment. Use the Communications Facility CFTERM instruction to define the terminals. For example, to define three 7485s on a multifunction attachment:

```
MFA7485 ADAPTER ADDRESS=58,TYPE=MFA,                     CC
                DEVICES=(T7485A,T7485B,T7485C),          CC
                END=YES
T7485A  CFTERM  ADDRESS=58,LMODE=RS422
T7485B  CFTERM  ADDRESS=59,LMODE=RS422
T7485C  CFTERM  ADDRESS=5A,LMODE=RS422
```

BSC Lines: Use the BSCLINE statement to define these to EDX. Specify TYPE=PT if the line is leased, TYPE=SM if the line is switched manual, or TYPE=SA if the line is switched auto-answer.

Local Communications Controller: Use EDX LCC statements to define the three subchannels of each Local Communications Controller attachment.

PC Connect: Use the Communications Facility PCC instruction to define the three subchannels of each Series/1-PC Connect Attachment. For example, to define one attachment:

```
PCC ADDRESS=58
PCC ADDRESS=59
PCC ADDRESS=5A,END=YES
```

If you want to support Local Communications Controller and PC Connect, you must define your PCC and LCC statements together. You must define all the PCC and LCC subchannels before specifying END=YES.

You must also modify the INITMOD parameter of the SYSTEM statement to include the PC Connect initialization module, CSPCCIPL. The PC Connect initialization module must be specified in this sequence:

```
INITMOD=(CSPCCIPL,CSXINIT)
```

Channel Attachment: The channel attachment IOCP uses the EXIO interface. Use the EXIODEV statement to define the channel attachment. For example:

```
EXIODEV ADDRESS=10,MAXDCB=1,RSB=6,END=YES
```

SNA Connection: The SNA IOCP requires the EDX-SNA licensed program, which uses the EXIO interface. Use the EXIODEV statement to define the SDLC attachment. For example:

```
EXIODEV ADDRESS=0A,MAXDCB=7,RSB=4,END=YES
```

The Communications Facility requirements for EDX-SNA are explained in "Series/1-to-Host SNA IOCP, $.IO14E8" on page 169.

X.25 Connection: The X.25 IOCP requires the EDX XHCS licensed program, which requires an XHCDEV statement in $EDXDEF for each XHCS line. For example:

```
LINK1A   XHCDEV   (1A,1B),ADAPTER=2080,END=YES
```

See the XHCS *Programming and Operating Reference* for a complete description of this statement.

You must also modify the INITMOD parameter of the SYSTEM statement to include the XHCS initialization module, CTZOII. The XHCS module must precede the Communications Facility initialization module.

```
INITMOD=(CTZOII,CSXINIT)
```

Remote Management Utility: The remote management utility, $.RMU, requires virtual terminal support. Use these statements to define it:

```
CDRVTA   TERMINAL DEVICE=VIRT,ADDRESS=CDRVTB,LINSIZE=254,SYNC=YES
CDRVTB   TERMINAL DEVICE=VIRT,ADDRESS=CDRVTA,LINSIZE=254
```

Remote Disk Support: Use the DISK statement to define each disk on a remote Series/1 that will be accessed as if it were attached to this Series/1. The device address must be a fictitious address; there must be no real disk or other device attached at that address. The device type must be the same as that of the corresponding real disk. Do *not* specify TASK=YES.

Use the Communications Facility DEFINE DEVICE instruction to define the path by which the remote disks are accessed when the program dispatcher is not running in this Series/1. This instruction is required even if the remote disks are not accessed until the program dispatcher is running.

Use the Communications Facility DEFINE VOLUME instruction to define the local and remote names of the volumes to be accessed.

For example, to define access over the Local Communications Controller to two volumes on a disk attached to the Series/1 known to the program dispatcher as cell SJ:

```
***
$EDXDEF  CSECT
           •
           •
*** DISK DEFINITIONS
***
         DISK DEVICE=4962-3,ADDRESS=E1,VOLNAME=(EDX002,LOCAL3),TASK=NO
           •
           •
*** COMMUNICATIONS FACILITY SUPPORT
***
           •
           •
         DEFINE DEVICE,LINE=50,CELL=S2,RINGADR=4F,ADDRESS=E1
         DEFINE VOLUME,CELL=SJ,VOLNAME=(EDX002,EDX002)
         DEFINE VOLUME,CELL=SJ,VOLNAME=(LOCAL3,EDX003),END=YES
```

When you have finished editing the supervisor definition data set, save it as $EDXDEF. In step 8, you will assemble it into an object module named $EDXDEFO and include it in the supervisor.

5. Create the System Storage Pool (S$POOL)

The Communications Facility requires a system storage pool, included in the EDX supervisor, as part of the common area for all partitions in which Communications Facility programs will run. This pool, named S$POOL, is where the station blocks for all started stations are stored.

The Communications Facility also requires a DCB pool (S$CFDCBS), resident in partition 1 of the supervisor. This pool contains Device Control Blocks (DCBs) for stations that the Communications Facility manages without the use of EDX-provided I/O facilities and executable code that calls the Communications Facility interrupt handler routines.

The chapter "Planning Storage Requirements" on page 213 explains how to calculate the size of S$POOL and S$CFDCBS. You can define these as shown in this step or use the alternative method described in step 10.

To define S$POOL, use any EDX text editor to create a source data set named S$POOL and enter the following statements:

```
CSXPOOL  CSECT
         DEFINE BUFFERPOOL,SIZE=nnnn
         END
```

where *nnnn* is the size of the pool in decimal. The DEFINE instruction generates the required pool name, S$POOL. In step 8, you will assemble the data set into an object module named CSXPOOL and include it in the supervisor.

To define S$CFDCBS, use any EDX text editor to create a source data set named S$CFDCBS and enter the following statements:

```
CSXDCBS   CSECT
S$CFDCBS  DEFINE BUFFERPOOL,SIZE=nnnn
          END
```

where *nnnn* is the size of the pool in decimal. The label S$CFDCBS must be coded as shown. In step 8, you will assemble the data set into an object module named CSXDCBS and include it in the supervisor.

6. Edit the Supervisor Link Control Data Set ($LNKCNTL)

Use any EDX text editor to modify $LNKCNTL, adding statements to include the supervisor definition module ($EDXDEFO), the EDX modules required to support the devices you defined, the Communications Facility system storage pool module (CSXPOOL), the Communications Facility DCB pool module (CSXDCBS), and the Communications Facility supervisor modules. CSXDCBS, CSA, CSPDIO, and CSP must be included in partition 1. The remaining Communications Facility supervisor modules must be included in the COMBASE partition. The modules cannot be in overlay segments.

The modules required for every Communications Facility system are:

- CSXPOOL, the system storage pool module. CSXPOOL must be included in the EDX common area. If the common area is based in partition 1, CSXPOOL must follow the EDX system table module, EDXSYS.

- CSXSYS, the Communications Facility system table. CSXSYS must be the last module included in the common area. CSXSYS contains label CSXEND which ends the EDX common area as defined by the COMMON= parameter on the SYSTEM statement.

- CSXDCBS, the DCB pool module. CSXDCBS must reside in partition 1, but not necessarily in the common area.

- $EDXDEFO, the supervisor definition module, and CSP, the supervisor Communications Facility parameter table. $EDXDEFO and CSP must be included in partition 1. Include $EDXDEFO and CSP after the CSXSYS module if you are basing the common area in partition 1. If not, the location of these modules in partition 1 is not important.

- CSA, the module that provides a universal attention capability for the command processor and program dispatcher commands. This module must be included in partition 1, outside the common area.

- CSP, the Communications Facility parameter table. This module must be included in partition 1, outside the common area.

- CSF, CSM, and CSX, the Communications Facility instruction support. These modules must be included in the CSXSYS (COMBASE) partition, after module CSXSYS.

- CSXINIT, the Communications Facility initialization module. Include CSXINIT in partition 1, before the $EDXSVCX module in the static portion of the supervisor.

The EDX *Installation and System Generation Guide* explains which EDX modules are required to support various devices and functions. Include the support you require, as follows:

4978 Terminals and Printers: Include EDX terminal support.

4980 Terminals: Include EDX terminal support, 4980 power-on read access memory support, and SMIO attachment support.

3101 Terminals, Feature Programmable Communications or Terminal/Host Adapter: Include EDX unmapped storage support. If you are going to use the terminals as EDX terminals or as Communications Facility work session controller terminals, include EDX terminal support.

3101 Terminals, Multifunction Attachment: Include EDX unmapped storage support and multifunction attachment support. If you are going to use the terminals as EDX terminals or as Communications Facility work session controller terminals, include EDX terminal support.

7485 Terminals, Feature Programmable Communications or Terminal/Host Adapter: Include EDX unmapped storage support.

7485 Terminals, Multifunction Attachment: Include EDX unmapped storage support and multifunction attachment support.

BSC Lines: Include EDX BSC support.

Local Communications Controller: Include EDX Local Communications Controller support.

PC Connect: Include EDX Local Communications Controller support. Include Communications Facility modules CSPCCINT, CSPCCIPL, CSPCCOPN, and CSPCCPTC. Modules CSPCCOPN, CSPCCPTC, and CSPCCINT may be in overlay segments (each in its own overlay segment). Module CSPCCIPL cannot be in an overlay segment.

Channel Attachment: Include EDX EXIO control support.

SNA Connection: Include EDX EXIO control support.

X.25 Connection: Include XHCS modules CTZOIA and CTZOII, as explained in the XHCS *Program Directory.*

Remote Management Utility: Include EDX virtual terminal support.

Configuration Processor and Panel Utility: Include EDX full message support.

Remote Disk Support: In a Series/1 that accesses remote disks, include EDX basic disk support. Include specific disk device support modules only for real disk devices, if any. Include Communications Facility modules CSPDIO and CSXRDINT. CSPDIO must be included before $EDXSVCX in the static portion of the supervisor.

$LNKCNTL Examples

Following are examples of $LNKCNTL data sets, showing where Communications Facility modules are included. The first example shows a system with the common area based in partition 1. The second example shows a system with the common area based in partition 3. Those marked with a single asterisk on the left are the optional remote disk support modules.

Example—Common Area Based in Partition 1

```
*** SUPERVISOR SUPPORT
 PART 1
 VOLUME XS4002                    DEFAULT VOLUME FOR INCLUDE
 INCLUDE EDXSYS          *1*      SYSTEM TABLES AND WORK AREAS
 INCLUDE CSXPOOL,EDX002  *CF*     C.F. SYSTEM STORAGE POOL
 INCLUDE CSXSYS,ASMLIB   *CF*     C.F. SYSTEM TABLE
*INCLUDE CSPDIO,ASMLIB   *CF*     C.F. REMOTE DISK SUPPORT
 INCLUDE CSXINIT,ASMLIB  *CF*     C.F. INITIALIZATION
 INCLUDE $EDXDEFO,EDX002 *1*      OUTPUT FROM USER SYSTEM GENERATION
     •
     •
*** COMMUNICATIONS FACILITY SUPPORT
*INCLUDE CSXRDINT,ASMLIB *CF*     REMOTE DISK INITIALIZATION
 INCLUDE CSX,ASMLIB      *CF*     C.F. LANGUAGE EXTENSION SUPPORT
 INCLUDE CSF,ASMLIB      *CF*     C.F. GET/PUT SUPPORT
 INCLUDE CSM,ASMLIB      *CF*     C.F. SEND/RECEIVE SUPPORT
 INCLUDE CSA,ASMLIB      *CF*     C.F. GLOBAL ATTENTION SUPPORT
 INCLUDE CSXDCBS         *CF*     C.F. DCB POOL
 INCLUDE CSP             *CF*     C.F. PARAMETER TABLE
     •
*** SYSTEM SUPPORT -- INITIALIZATION
     •
     •
 INCLUDE CSPCCINT,ASMLIB  *CF*  PCC INITIALIZATION
 INCLUDE CSPCCIPL,ASMLIB  *CF*  PCC INITIALIZATION CALL ROUTINE
 INCLUDE CSPCCOPN,ASMLIB  *CF*  PCC MICROCODE DATA SETS OPEN ROUTINE
 INCLUDE CSPCCPTC,ASMLIB  *CF*  PCC MICROCODE RAM LOAD ROUTINE
*** SUPERVISOR SUPPORT OUTSIDE PARTITION 1
     •
     •
 LINK $EDXNUCT,EDX002
```

Example—Common Area Based in Partition 3

```
*** SUPERVISOR SUPPORT
 PART 1
 VOLUME XS5202                    DEFAULT VOLUME FOR INCLUDE
 INCLUDE EDXSYS          *1*      SYSTEM TABLES AND WORK AREAS
 INCLUDE CSXINIT         *CF*     C.F. INITIALIZATION MODULE (PARTITION 1)
*INCLUDE CSPDIO          *CF*     REMOTE DISK SUPPORT (PARTITION 1)
 INCLUDE $EDXDEFO                 OUTPUT FROM USER SYSTEM GENERATION
     •
     •
*** COMMUNICATIONS FACILITY SUPPORT
*INCLUDE CSXRDINT,ASMLIB *CF*     REMOTE DISK INITIALIZATION
 INCLUDE CSXDCBS         *CF*     C.F. DCB POOL
 INCLUDE CSP             *CF*     C.F. PARAMETER TABLE
 INCLUDE CSA             *CF*     C.F. GLOBAL ATTENTION SUPPORT
 INCLUDE CSA,ASMLIB      *CF*     C.F. GLOBAL ATTENTION SUPPORT
     •
     •
PART 3
*** COMBASE PARTITION
 INCLUDE CSXPOOL           *1*     C.F. SYSTEM STORAGE POOL
 INCLUDE CSXSYS            *1*     C.F. SYSTEM TABLE
*** END COMMON AREA
     •
     •
     •
 INCLUDE CSX             *CF*     C.F. LANGUAGE EXTENSION SUPPORT
 INCLUDE CSF             *CF*     C.F. GET/PUT SUPPORT
 INCLUDE CSM             *CF*     C.F. SEND/RECEIVE SUPPORT
     •
     •
 LINK $EDXNUCT,EDX002
```

Save the edited version of $LNKCNTL on EDX002.

7. Edit the Assembler Control Data Set ($EDXL)

Use any EDX text editor to edit the $EDXL control data set on ASMLIB to include the overlays that compile the Communications Facility language extension instructions. Merge the data set A.EDXL (statements 10-50) between the last OVERLAY statement and the first COPYCOD statement, and before any EXTLIB statement.

If you need to assemble the X.25 IOCP or other programs which use XHCS instructions, you must include an EXTLIB statement in $EDXL for the data set $XHCL. ($XHCL is shipped with XHCS.)

If you need to assemble the SNA IOCP or other programs which use SNA instructions, you must include an EXTLIB statement in $EDXL for the data set $EDXLSNA.

A partial listing of the merged data sets in the workspace should show:

```
*OVERLAY A.CFID   ASMLIB CFID
*OVERLAY A.CFTERM ASMLIB CFTERM
*OVERLAY A.CSXDEF ASMLIB DEFINE
*OVERLAY A.GETPUT ASMLIB GET      PUT      MOV
*OVERLAY A.SENREC ASMLIB ACTIVATE FREE     LOCATE   RECEIVE  SEND
*COPYCOD ASMLIB
*COPYCOD EDX002
**STOP**
```

Save the edited version of $EDXL on ASMLIB. The PCC instruction will assemble without an entry in $EDXL.

8. Rebuild the Supervisor

Perform these steps to rebuild the supervisor:

1. Assemble the supervisor definition statements in $EDXDEF (which you saved in step 4) and store the object module in $EDXDEFO:

   ```
   > $L $EDXASM,ASMLIB $EDXDEF ASMWORK $EDXDEFO
   ```

2. Assemble S$POOL and S$CFDCBS (which you created in step 5) and store the object modules in CSXPOOL and CSXDCBS:

   ```
   > $L $EDXASM,ASMLIB S$POOL ASMWORK CSXPOOL
   > $L $EDXASM,ASMLIB S$CFDCBS ASMWORK CSXDCBS
   ```

3. Link-edit the supervisor, using the link control data set, $LNKCNTL, which you created in step 6:

```
> $L $XPSLINK LINKWORK
```

Enter $LNKCNTL when you are prompted for parameters. Check the link-edit printed output to be sure that there are no unresolved external references and that the supervisor is not larger than 64K in any partition. You'll get unresolved external references if you've defined remote disks with types that aren't attached to the Series/1, because you didn't include specific disk support modules. These errors are all right; the unresolved labels will never be referenced. If the supervisor is too large, compact it as explained in step 10.

9. Rewrite IPL Text and Re-IPL

Use the II (initialize IPL text) command of the EDX utility $INITDSK to rewrite IPL text:

```
> $L $INITDSK
COMMAND (?): II
NUCLEUS: $EDXNUCT
VOLUME: EDX002
IPL TEXT WRITTEN
```

Press the LOAD button to re-IPL the system with your new supervisor that supports the Communications Facility.

10. Compact the Supervisor

If the supervisor exceeds the 64K partition size limit, you can use one or more of the three methods described in this section to compact it.

The first method is to use the supervisor overlay support, which is described in the EDX *Installation and System Generation Guide*. You must place CSXINIT, CSPCCIPL (if PC Connect is used), and CSXRDINT (if remote disk support is used) before the INCLUDE statement for the overlay manager. These modules *cannot* be in overlay segments, even though they are user initialization modules.

The second method is to move part of the supervisor outside partition 1, as explained in the EDX *Installation and System Generation Guide*. Remember that you can move the EDX common area, and many associated Communications Facility modules, out of partition 1. This will free the storage that would have been reserved for S$POOL and CSXSYS in partition 1 for the rest of the supervisor.

The third method is to use the storage reserved in partition 1 for S$POOL or S$CFDCBS to contain supervisor modules during the early part of the IPL process. This method may be applied to S$POOL only if it is included in partition 1.

Because the CSXINIT module initializes S$POOL late in the IPL process, you can use the S$POOL storage for other purposes before CSXINIT receives control.

To use the S$POOL area in partition 1 before initialization, follow these steps:

1. Change the DEFINE BUFFERPOOL instruction in the S$POOL data set to:

   ```
   DEFINE BRB,SIZE=nnnn
   ```

   where *nnnn* is the size, in decimal, of S$POOL plus 8 bytes.

2. Reassemble the module.

3. Modify the $LNKCNTL data set by moving one or more INCLUDE statements for supervisor initialization modules between the INCLUDE CSXPOOL statement and the INCLUDE CSXSYS statement. You may use almost any of the initialization modules as long as their combined size *exceeds* the size specified in the DEFINE BRB instruction. If their size is less, S$POOL will overlay module CSXSYS.

   You can include Communications Facility initialization modules in this area (CSXINIT, CSXRDINT, CSPCCINT, CSPCCIPL, CSPCCOPN, and CSPCCPTC). Do *not* include EDXINIT in this area.

   Use the link-edit map from your initial link-edit to determine the sizes of the initialization modules.

4. Link-edit the supervisor.

5. Check the supervisor size. If it's acceptable, rewrite the IPL text and re-IPL the new supervisor.

Because S$CFDCBS is not initialized until Communications Facility programs are loaded into storage after IPL is complete, S$CFDCBS storage is available during the entire IPL process.

To use the S$CFDCBS area before initialization, follow these steps:

1. Change the DEFINE BUFFERPOOL instruction in the S$CFDCBS data set to:

   ```
   S$CFDCBS DEFINE BRB,SIZE=nnnn
   ```

   where *nnnn* is the size, in decimal, of the Communications Facility DCB pool

2. Reassemble the module.

3. Modify the $LNKCNTL data set by moving one or more INCLUDE statements for supervisor modules immediately after the INCLUDE CSXDCBS statement in partition 1. You may use almost any of the initialization modules as long as their combined size *exceeds* the size specified in the DEFINE BRB instruction. A significant size reduction in the supervisor is achieved only if the DCB pool is almost as large as the total size of the initialization modules you want to include in this area. As described before for using the S$POOL area, you may use Communications Facility initialization modules, but do *not* include EDXINIT in this area.

4. Link-edit the supervisor.

5. Check the supervisor size. If it's acceptable, rewrite the IPL text and re-IPL the new supervisor.

Note: This technique is appropriate only if you have defined a relatively large DCB pool.

11. Initialize Data Sets for Disk Queuing

If you use disk queuing of messages, you need to allocate and initialize disk-queue data sets. The chapter "Creating Disk-Queue Data Sets" on page 207 explains how to do this.

12. Define the Communications Facility Stations

Define the stations required to represent your hardware and software configuration.

The chapter "Defining Stations" on page 45 explains how to define stations and gives worksheets to use to record your station definitions.

Fill out those worksheets, and then use the DEFINE and LINK commands of $.CONFIG or the command processor to build the network configuration data set, $.SYSNET.

The chapter "Sample Configurations" on page 269 gives many examples of station definition worksheets.

13. Create the Program Dispatcher Data Set ($.SYSPD)

If you are going to use the program dispatcher, you must set up a $.SYSPD data set, which contains commands, path definitions, transaction definitions, and transactions that are to be processed when the program dispatcher is started. The chapter "Creating $.SYSPD" on page 187 explains how to do this.

The chapter "Sample Configurations" on page 269 shows several sample $.SYSPD data sets.

14. Create the IPL Transaction Data Set ($.SYSLCC)

If you plan to process hardware IPL requests initiated at remote nodes on a Local Communications Controller ring, you must create a $.SYSLCC data set that contains the appropriate IPL transactions. The section "Remote IPL" on page 37 explains the usage of $.SYSLCC.

" Example 9: Local Communications Controller Connection with Transaction Processing" on page 354 in the chapter "Sample Configurations" includes a sample $.SYSLCC data set.

15. Create the X.25 Call ID Data Set ($.SYSX25)

If you plan to use the X.25 I/O control program ($.IO0AB8) with switched virtual circuits, you must create the $.SYSX25 data set that contains the circuits' call IDs matched with the appropriate X.25 network address. The section "X.25 Call ID Data Set ($.SYSX25)" on page 184 explains how to do this. See "Example 13: X.25 Connection Between Series/1s through an X.25 Network" on page 338 or "Example 14: X.25 Connection between a Series/1 and DTEs" on page 352 a $.SYSX25 data set.

You need not modify the supplied version of $.SYSX25 for use with PVC circuits.

16. Customize Logon Prompt Panel for SNA IOCP

If you plan to use the Communications Facility to connect to an 8100 Distributed Processing Executive (DPPX) system, you must use an alternate SNA logon prompt screen. Rename members SNALOGON and SNALOG02 in the $.SYSPNL library using the EDX utility $DIUTIL:

>$L $DIUTIL $.SYSPNL

```
COMMAND (?): RE SNALOGON SNALOG01
RENAME COMPLETED
```

```
COMMAND (?): RE SNALOG02 SNALOGON
RENAME COMPLETED
```

The alternate screen has separate fields for USERID and PASSWORD. Other host systems use the USER DATA field for these parameters.

17. Install EDX Secondary SNA1 or SNA2 Support

Relink the SNA IOCP ($.IO14E8) with $NETCMD, if you plan to use EDX Secondary SNA1. Link $.IO14E8 with $NETCMD and $NETPACT if you plan to use EDX Secondary SNA2 and $.IO14E8 with multiple PU support.

18. Install LU-1 Support

If you plan to use Series/1 printers for communication with SNA type 1 LUs or as PC LAN-shared printers, you must add LU-1 support to the printer I/O control program ($.IO0674).

Link these modules in the order listed to create the load module $.IO0674:

- O$IO0674
- O$IO0675
- O$IO0676
- O$IO0677

19. Create Emulated PC Disks

If you plan to use the disk-server support, you must use the emulated PC disk management utility, $.DSFORM, to create the emulated PC disks you need on a Series/1 volume. See the *Operator's Guide* for more information on $.DSFORM.

20. Use $.CFMENU to Verify Installation

You can verify that you have installed the Communications Facility correctly and test certain communications functions by using the sample program $.CFMENU.

The functions you can test are:

- Communication with 3270 display stations.
- Communication with 4978, 4980, 3101, or 7485 terminals being managed as if they were 3277s.
- Communication with a host using 3270 emulation, Series/1-to-host channel attachment, or Series/1-to-host SNA.

To use the sample program, you must first define it to the Communications Facility as a user station. Use $.CONFIG to define the program as a user station, with station name $.CFMENU and a network address appropriate to your configuration.

Issue one of these commands to start the Communications Facility:

```
> $L $.CFS
> $L $.CFD
```

Issue this command to start the sample program:

```
> CP S $.CFMENU
```

Issue commands to link one or more terminals to the program. For example, to link T4978 to the program, you would enter:

```
> CP LINK T4978 $.CFMENU
```

If the terminal hasn't been started, issue a command to start it:

```
> CP S T4978
```

Begin communication with $.CFMENU by pressing the ENTER key of a terminal linked to it. The menu shown in Figure 40 on page 267 will be displayed on the screen. When the display appears, you have successfully tested communication with the terminal.

```
******************************************************************************

                EVENT DRIVEN EXECUTIVE COMMUNICATIONS FACILITY

                               SAMPLE PROGRAM

******************************************************************************

                    CONNECT TO HOST:            (ENTER HOST TERMINAL NAME)

           CONNECT TO APPLICATION PGM:          (ENTER PROGRAM NAME)

               LOAD $FSEDIT UNDER EDX:          (ENTER WORKFILE NAME)

             DISCONNECT THIS TERMINAL:          (ENTER ANY CHARACTER)


===> TAB TO FIELD AND ENTER DATA
```

Figure 40. $.CFMENU Menu

To test communication with a host, you must first define the appropriate line and terminal stations, or the SNA physical and logical unit stations, and start them. For example, to start a 3270 emulation line and an emulated control unit and terminal, enter this command at a terminal that is *not* linked to $.CFMENU:

```
> CP S HOST CU71 T770
```

Then enter the name of the emulated terminal station in the sample program menu and press ENTER. A screen image from the host program should be displayed on the terminal's screen.

You can use the sample program to link to another Communications Facility application program, such as $.PANEL, the panel design aid program. You must first define the program, specifying a network address appropriate to your configuration, and start it:

```
> CP DEF $.PANEL 0188 USER
> CP S $.PANEL
```

You can load the EDX full-screen editor from a 4978 terminal by entering the name of an edit work file in the sample program menu. Note that the sample program is not intended to be used as a session manager. It does not allow you to specify where the editor is loaded or a work file volume name.

To end a single session with $.CFMENU and return the terminal to control of EDX, enter any character in the fourth field of the menu.

To end the use of $.CFMENU, issue a command to stop it:

```
> CP P $.CFMENU
```

Sample Configurations

This chapter presents sample Communications Facility configurations:

1. BSC connection to a host, with a 4978 terminal, a 4980 terminal, and a printer.
2. Channel attach connection to a host, with 3101 terminals on a multifunction attachment.
3. EDX Secondary SNA1 connection to a host, with 3270 terminals and printer.
4. EDX Secondary SNA2 connection to a host, with 3270 terminals and printer.
5. Local Communications Controller connection between Series/1s, with 3101 terminals on a feature programmable communications adapter.
6. BSC point-to-point connection between Series/1s, with 3101 terminals on a feature programmable communications adapter.
7. BSC multipoint connection between Series/1s.
8. BSC multipoint connection with transaction processing. See "Remote IPL" on page 37 for restrictions.
9. Local Communications Controller connection with transaction processing. See "Remote IPL" on page 37 for restrictions.
10. BSC point-to-point connection with transaction processing.
11. Multiple Local Communications Controller and BSC connections.
12. X.25 leased line connection between Series/1s with a permanent virtual circuit.
13. X.25 network connection between Series/1 with switched virtual circuits.
14. X.25 connection between Series/1s and through an X.25 network with packet devices.
15. LAN with Series/1-PC Connect Attachment and SNA connection.
16. PC Connect with inter-LAN services.

Each example includes a diagram of the configuration and an explanation of the message flow between stations. The diagram shows the hardware elements and their device addresses, the Communications Facility stations, and the links between stations. Stations are shown as boxes within the Series/1 box. Vector stations, those that represent elements in a remote node, are shown as dotted boxes. Links between stations are shown as dashed lines with arrows that indicate the direction of the link.

Each example also includes the station definitions and EDX definitions. The EDX definitions contain only the statements required to define and support the hardware and functions used by the elements of the sample configuration. When you define your configuration, you will need to add statements for basic EDX functions and for other hardware, such as EDX terminals, disks, and diskettes.

Where applicable, the example also includes other definitions required to support the configuration, such as EDX-SNA definitions, host system definitions, and program dispatcher data sets.

You can use the sample configurations, individually or combined, as models when you define your configuration. For example, if your configuration includes Series/1s connected by a Local Communications Controller with an SNA connection from one of the Series/1s to a host system, combine the information presented in examples 3, 4, and 5.

Example 1: BSC Connection to a Host

Figure 41 shows a configuration in which a Series/1 is connected to a host computer by a BSC multipoint line. The Series/1 appears to the host as a remote 3271 control unit with two 3277 display stations and a 3286 printer. The real devices are a 4978 terminal, a 4980 terminal, and a 4974 printer.

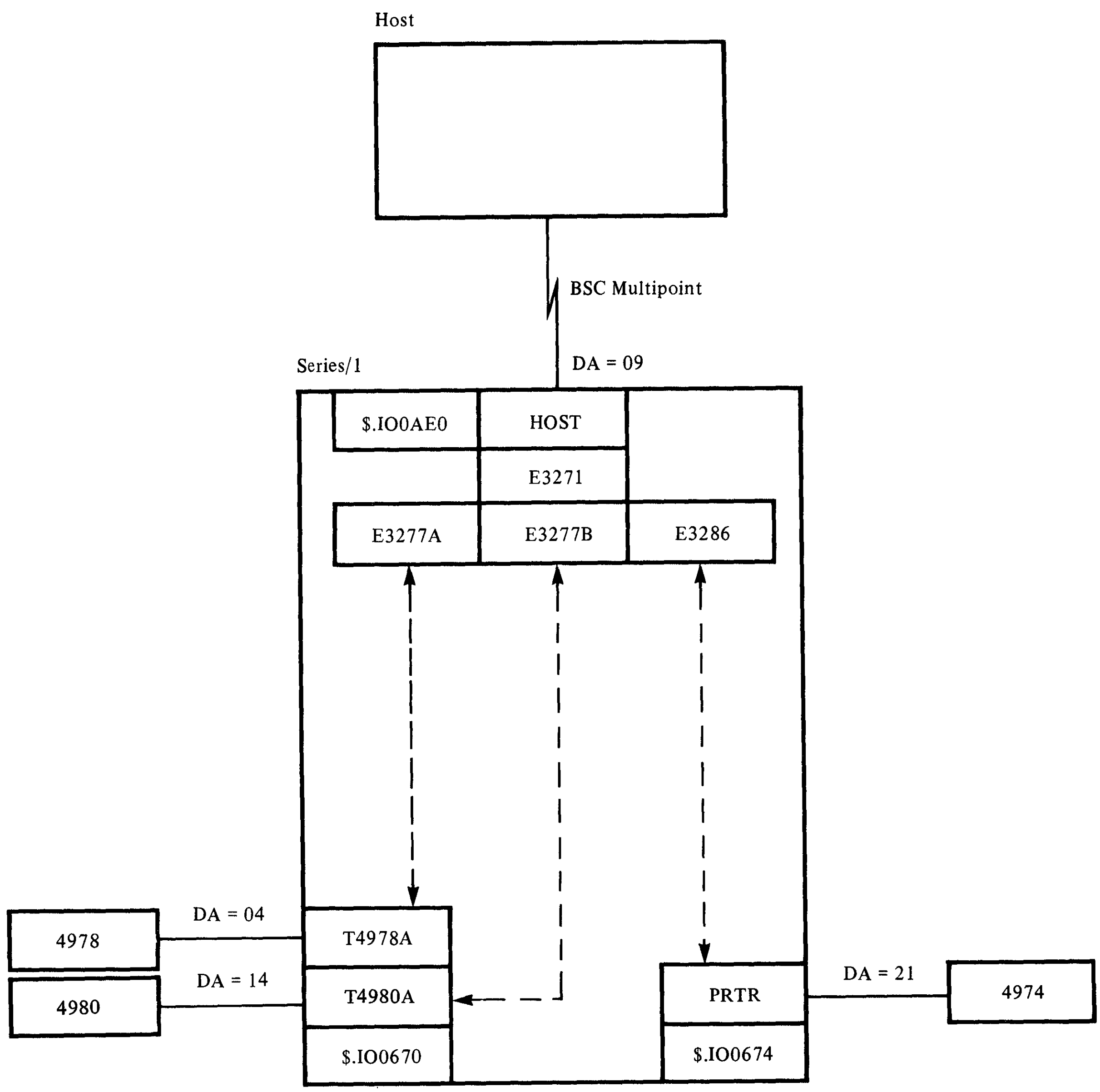


Figure 41. Example 1: BSC Connection to a Host

Message Flow

To achieve the flow of messages between the Series/1 devices and the host, each device station and its associated emulated terminal or device station are linked to each other. When an operator enters data at the 4978 represented by station T4978A, the 4978/4980 data stream IOCP, $.IO0670, reads the data, converts it to a 3270 data stream, and sends it to T4978A's link, station E3277A. Subsequently, the 3270 emulation IOCP, $.IO0AE0, retrieves the message from E3277A's queue and writes it to the associated BSC line, station HOST.

$.IO0AE0 also reads data from the host, and disposes of it according to the 3270 polling and selection address that you defined for the emulated terminal stations. For example, data with selection address X'6040' is intended for station E3277A, so it is sent to that station's link, station T4978A. Subsequently, $.IO0670 retrieves the message from T4978A's queue, decodes the 3270 data stream, and writes the data to the 4978.

Notice station E3271, which represents the emulated 3271 control unit. Although this station does not send or receive messages, it is required for communication over a BSC multipoint line.

Station Definitions

Figure 42, Figure 43 on page 274, and Figure 44 on page 275 show the station definitions. In Figure 43 on page 274, notice that a partitioned data set member has been specified for the printer station. As a result, a member named PRTR will be allocated in data set $.SYSPRT, and the printer can be supported as a buffered device.

User/Message/Alias/Volume Configuration Worksheet

System: Example 1: BSC Connection to Host
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
3270 Emulation IOCP	$.IO0AE0	01AA	USER	
4978/4980 Data Stream IOCP	$.IO0670	01AB	USER	
Printer IOCP	$.IO0674	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

3 User/Msg Stations	× 88 = 264 bytes
_____Alias Stations	× 40 = _____ bytes
_____Remote Stations	× 32 = _____ bytes
_____Volume Stations	× 104 = _____ bytes

Figure 42. Example 1: User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 1: BSC Connection to Host
Node Assignment: 01

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
4978	T4978A	0104	DEVICE	4978	2500		4040	0000		
4980	T4980A	0114	DEVICE	4978	2500		4040	0000		
Printer	PRTR	0121	DEVICE	PRINTER	2500	Y	4040	0000		

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485's, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

______3101F Devices	× 160 = ______ bytes
3 4978/Printer Devices	× 264 = 792 bytes
______3101 Devices	× 304 = ______ bytes
______Remote Stations	× 32 = ______ bytes

Figure 43. Example 1: Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 1: BSC Connection to Host
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC line to host	HOST	0109	LINE	3271E	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Emulated control unit	E3271	01E1	TERM	3271		HOST		407F	0000
Emulated terminal	E3277A	01E2	TERM	3277		HOST		4040	6040
Emulated terminal	E3277B	01E3	TERM	3277		HOST		40C1	60C1
Emulated printer	E3286	01E4	TERM	3286		HOST		40C2	60C2

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
4 Terminals/Nodes	× 88 = 352 bytes
_____Remote Stations	× 32 = ______ bytes

Figure 44. Example 1: Line/Terminal/Node Configuration Worksheet

EDX Definitions

Figure 45 shows the EDX definitions for example 1. You must define the 4978, 4980, and the printer to EDX even though you might use them only as Communications Facility terminals. Notice the definition of PRINT2, the printer to be used as if it were a 3286. The definition is based on the assumption that the host system controls pagination with NL (new line) control orders in the data streams that it sends to the printer. If the host system controls pagination with FF (forms feed) control orders, you should change the definitions of the top and bottom margins.

```
*** EXAMPLE 1
***
$EDXDEF   CSECT
          SYSTEM   MAXPROG=(10,10),PARTS=(32,32),              X
               COMMON=(CSXEND,CSXEND),                          X
               INITMOD=(CSXINIT)                                X
            •
            •
SMIO01    ADAPTER  ADDRESS=80,TYPE=SMIO,DEVICES=(TERM2),END=YES
TERM1     TERMINAL DEVICE=4978,ADDRESS=04,HDCOPY=$SYSPRTR
TERM2     TERMINAL DEVICE=4980,ADAPTER=SMIO,ADDRESS=80,         X
               SECADDR=01,PORT=0
$SYSPRTR  TERMINAL DEVICE=4973,ADDRESS=01
PRINT2    TERMINAL DEVICE=4974,ADDRESS=21,                      X
               PAGSIZE=66,TOPM=0,BOTM=65,END=YES
            •
            •
          BSCLINE  ADDRESS=09,TYPE=MT,END=YES   3271 EMULATION
          END
```

Figure 45. Example 1: EDX Definitions

Example 2: Channel Attach Connection to a Host

Figure 46 on page 279 shows a configuration in which a Series/1 is connected to an IBM System/370 computer by a channel attachment. The Series/1 appears to the host as a locally-attached 3272 control unit with two 3277 display stations. The real devices are a 3101 and a 3161 terminal attached to the Series/1 by a multifunction attachment.

Message Flow

To achieve the flow of messages between the Series/1 devices and the host, each device station and its associated emulated terminal station are linked to each other. A user application program (station APROG) is included in this example to illustrate how you can intercept all messages sent by an IOCP.

When an operator enters data at a 3101 or a 3161 terminal, the 3101 IOCP, $.IO06F0, reads the data, converts it to a 3270 data stream, and sends it to the IOCP's link, station APROG. The application program can do whatever you require with the messages it receives; for example, it might check them for validity or write them to a data set as an audit trail. The origin of the messages is the input device station (T3101A or T3101B), not the I/O control program. Therefore, the application program can send messages on to the host by sending them to the origin station's link (PORTA or PORTB). The channel attach IOCP, $.IO0AD0, retrieves messages from PORTA's and PORTB's queues and writes them to the associated channel, station CHAN.

$.IO0AD0 disposes of data sent by the host according to the port number that you defined for the emulated terminal stations. For example, data for port 0 is intended for station PORTA, so the IOCP sends it to PORTA's link, station T3101A. Notice that with a channel attachment, there is no station that represents the emulated 3272 control unit.

In this example, there is no program to intercept messages from the host. You could intercept them by linking $.IO0AD0 to APROG or to some other application program. See the *Programmer's Guide* for information about writing programs to intercept messages from I/O control programs.

Station Definitions

Figure 47 on page 280, Figure 48 on page 281, and Figure 49 on page 282 show the station definitions.

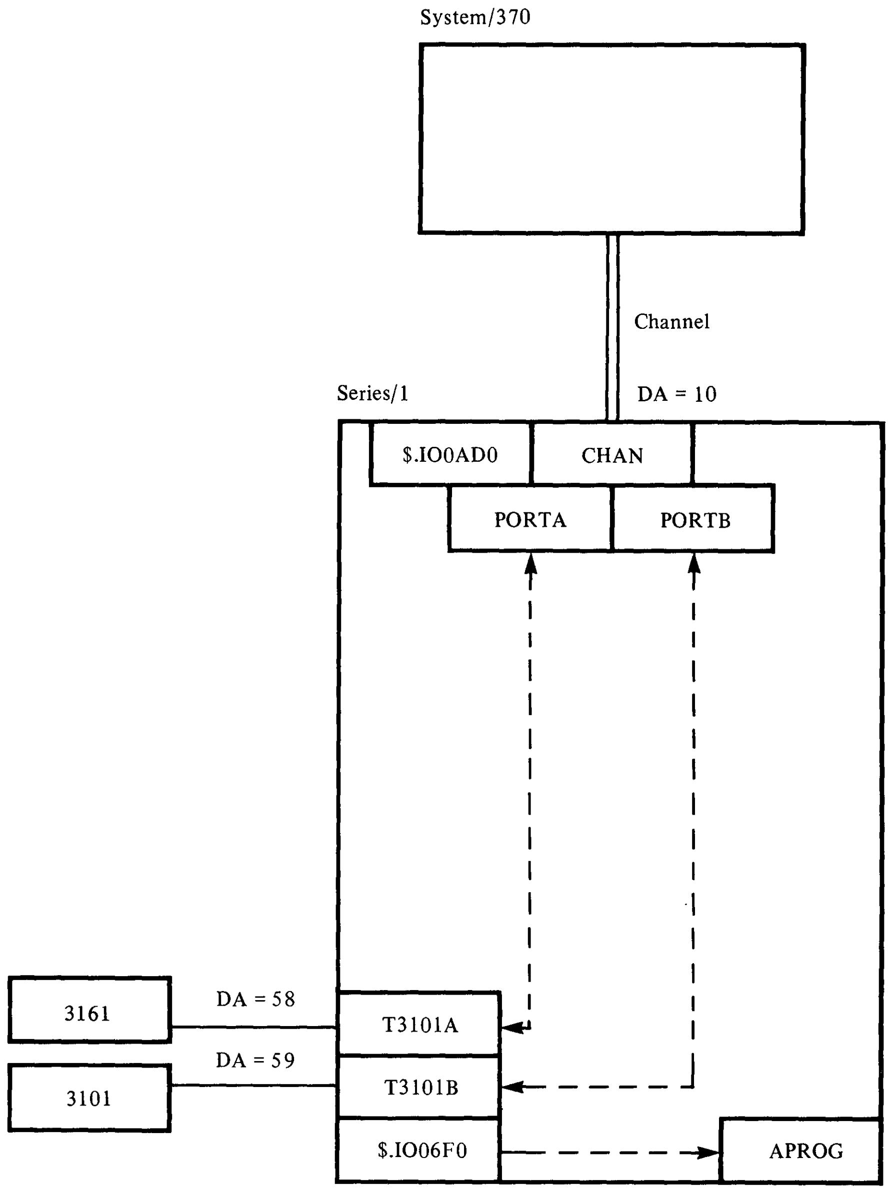


Figure 46. Example 2: Channel Attach Connection to a Host

User/Message/Alias/Volume Configuration Worksheet

System: Example 2: Channel Attach Connection to a Host
Node Assignment: 01

Description	**Name**	**NA**	**Type**	**Alias**[1]
Channel attach IOCP	$.IO0AD0	01AA	USER	
3101-MFA IOCP	$.IO06F0	01AB	USER	
User application program	APROG	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

3 User/Msg Stations × 88 = 264 bytes
_____Alias Stations × 40 = _______ bytes
_____Remote Stations × 32 = _______ bytes
_____Volume Stations × 104 = _______ bytes

Figure 47. Example 2: User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 2: Channel Attach Connection to a Host
Node Assignment: 01

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3161	T3101A	0158	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
3101	T3101B	0159	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485's, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

_____3101F Devices	× 160 = _____ bytes
_____4978/Printer Devices	× 264 = _____ bytes
2 3101 Devices	× 304 = 608 bytes
_____Remote Stations	× 32 = _____ bytes

Figure 48. Example 2: Device Configuration Worksheet

EDX Definitions

Figure 50 on page 283 shows the EDX definitions for example 2.. Notice that the CFTERM instruction is used to define the 3101 and 3161, so you can use them only as Communications Facility terminals. If you also want to use them as EDX terminals, use the TERMINAL statement to define them instead.

Line/Terminal/Node Configuration Worksheet

System: Example 2: Channel Attach Connection to Host
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Channel attachment	CHAN	0110	LINE	CA	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Emulated 3277	PORTA	01E1	TERM	PORT		CHAN		0	
Emulated 3277	PORTB	01E2	TERM	PORT		CHAN		1	

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
2 Terminals/Nodes	× 88 = 176 bytes
_____Remote Stations	× 32 = ______ bytes

Figure 49. Example 2: Line/Terminal/Node Configuration Worksheet

```
*** EXAMPLE 2
***
$EDXDEF    CSECT
           SYSTEM MAXPROG=(10,10),PARTS=(32,32),                X
                COMMON=(CSXEND,CSXEND),                          X
                INITMOD=(CSXINIT)
             •
             •
$SYSPRTR   TERMINAL DEVICE=4973,ADDRESS=01,END=YES
MFA01      ADAPTER  ADDRESS=58,TYPE=MFA,DEVICES=(TERMA,TERMB),END=YES
TERMA      CFTERM   ADDRESS=58,LMODE=RS422
TERMB      CFTERM   ADDRESS=59,LMODE=RS422
             •
             •
           EXIODEV  ADDRESS=10,MAXDCB=1,RSB=6,END=YES   CHANNEL
           END
```

Figure 50. Example 2: EDX Definitions

The EDX terminal support modules shown in the figure are those required to support the 3101 hard copy device, $SYSPRTR. No EDX modules are required to support the 3101s as Communications Facility terminals.

Example 3: EDX Secondary SNA1 Connection to a Host

Figure 51 on page 285 shows a configuration in which a Series/1 is connected to a host computer by an SNA1 connection. The Series/1 appears to the host as an SNA physical unit, two SNA type 2 logical units (terminals), one SNA type 1 logical unit (printer), and one SNA type 3 logical unit (printer). The real devices are two 3277 display stations, one 4974 printer, and one 3286 printer, respectively.

Message Flow

To achieve the flow of messages between the Series/1 terminals and the host, each terminal or device station and its associated SNA logical unit station are linked to each other. When an operator enters data at the terminal represented by station C3277A, the 3270 control IOCP, $.IO0AC0, reads the data and sends it to C3277A's link, station LCFS01. Subsequently, the SNA IOCP, $.IO14E8, retrieves the message from LCFS01's queue and calls on EDX-SNA to write it to the SDLC line.

$.IO14E8 also receives data from the host, and disposes of it according to the local address that you defined for the SNA logical unit stations. For example, data for logical unit 2 is intended for station LCFS02, so it is sent to that station's link, C3277B.

The 3270 system consists of devices 0-2 on control unit 0. These are the attachment numbers that you use to determine the 3270 polling and selection addresses of the terminals. Station C3271, which represents the 3271 control unit, is required for communication with a 3270 system even though it doesn't send or receive messages.

Station Definitions

Figure 52 on page 286, Figure 53 on page 287, Figure 54 on page 288, and Figure 55 on page 289 show the station definitions. Notice the terminal type of the SNA logical units in Figure 55. It must correspond to the type of station that the logical unit is linked to, either a real 3270 terminal or a Series/1 device being used as if it were a 3277, a 3286, or a 3287.

Notice the logon information in Figure 55 on page 289. Three host applications are specified: #1 is for TSO/terminal communication, #2 is for IMS/printer communication, and #3 is for CICS/printer communication. An operator at the 3277 terminal linked to station LCFS01 is automatically connected to TSO when the terminal and LU station are started.

An operator at the 3277 terminal linked to station LCFS02 is presented with a logon prompt screen when the terminal and LU stations are started. ("Series/1-to-Host SNA IOCP, $.IO14E8" on page 169 includes a picture of a logon screen.) The screen shows the applications specified in the PU. The terminal user can select option 1 to log on to TSO or option 4 to log on to any other SNA application defined in the host.

The 3286 printer linked to station LCFS03 is automatically connected to CICS when the printer and LU stations are started. The 4974 printer linked to station LCFS04 is automatically connected to IMS when the printer and LU stations are started. IMS uses SNA type 1 logical units for printer support, so the printer

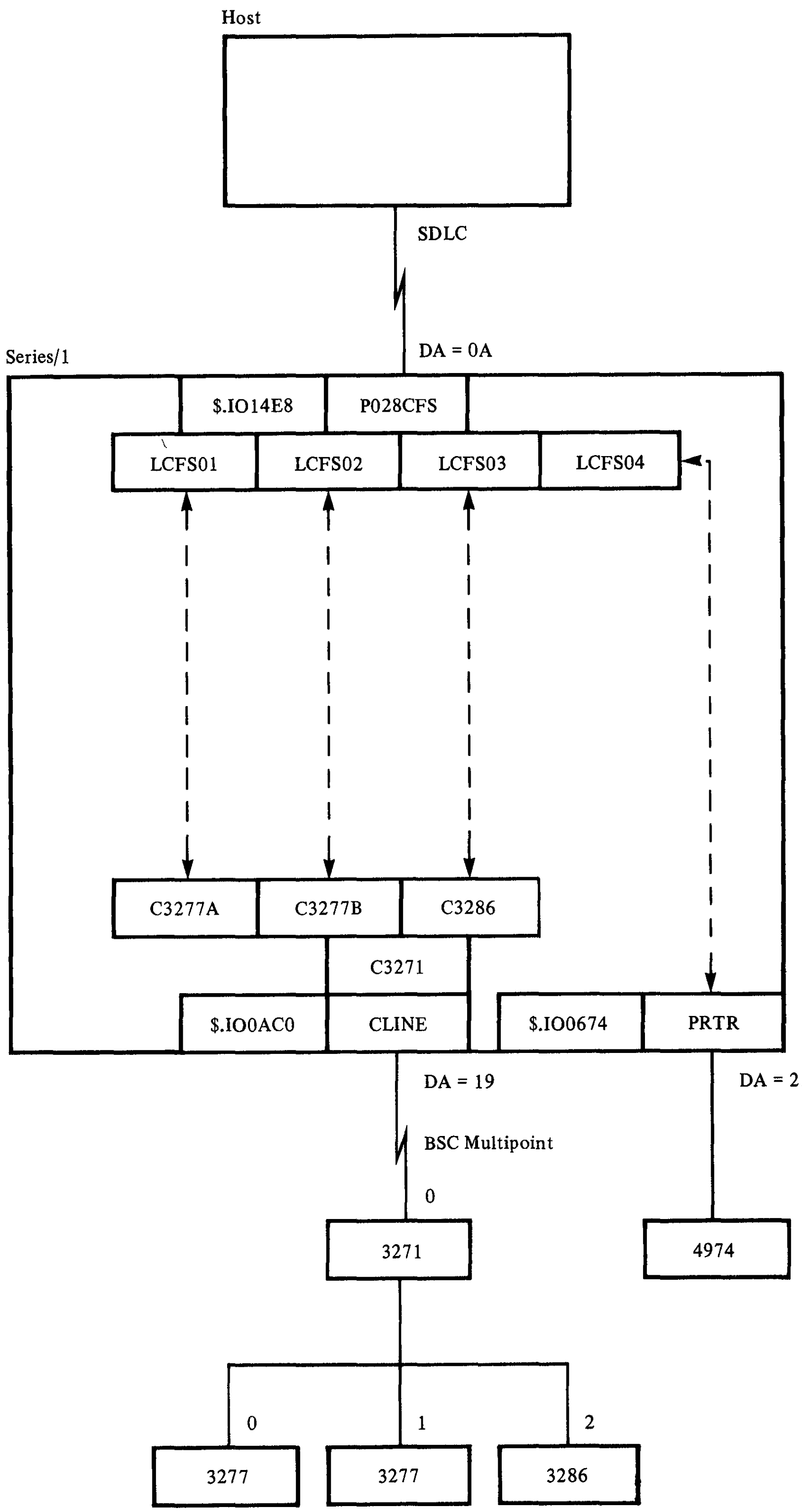


Figure 51. Example 3: EDX Secondary SNA1 Connection to a Host

station will receive SNA character string (SCS) data. You must include LU-1 support in $.IO0674 and set the printer station to SCS mode.

User/Message/Alias/Volume Configuration Worksheet

System: Example 3: SNA Connection to a Host
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
SNA IOCP	$.IO14E8	01AA	USER	
3270 Control IOCP	$.IO0AC0	01AB	USER	
Printer IOCP	$.IO0674	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations × 88 = 176 bytes
_____Alias Stations × 40 = _____ bytes
_____Remote Stations × 32 = _____ bytes
_____Volume Stations × 104 = _____ bytes

Figure 52. Example 3: User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 3: SNA Connection to a Host
Node Assignment: 01

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
Printer	PRTR	0121	DEVICE	Printer	2500	N	4040	00		

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485's, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

______3101F Devices	× 160 = ______ bytes
1 4978/Printer Device	× 264 = 264 bytes
______3101 Devices	× 304 = ______ bytes
______Remote Stations	× 32 = ______ bytes

Figure 53. Example 3: Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 3: SNA Connection to a Host
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC line to 3270	CLINE	0119	LINE	3271	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Control Unit	C3271	01C1	TERM	3271		CLINE		407F	0000
Terminal	C3277A	01C2	TERM	3277		CLINE		4040	6040
Terminal	C3277B	01C3	TERM	3277		CLINE		40C1	60C1
Printer	C3286	01C4	TERM	3286		CLINE		40C2	60C2

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

______HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
4 Terminals/Nodes	× 88 = 352 bytes
______Remote Stations	× 32 = ______ bytes

Figure 54. Example 3: Line/Terminal/Node Configuration Worksheet

SNA Configuration Worksheet

System: Example 3: EDX Secondary SNA1 Connection to a Host
Node Assignment: 01

PU Definition

Description	Name	NA	Type	Buffer Size	PU #[1]	Host Application	Host Mode
SNA physical unit	P028CFS	010A	PU	2500		#1: TSO	LU2256
						#2: IMS	STD32741
						#3: CICS	STD32743

LU Definitions

Description	Name	NA	Type	Term Type	PU Name	Local Address	Logon ID	User Data
Emulated terminal	LCFS01	01E1	LU	3277	P028CFS	1	1	
Emulated terminal	LCFS02	01E2	LU	3277	P028CFS	2		
Emulated printer	LCFS03	01E3	LU	3286	P028CFS	3	3	
Emulated printer	LCFS04	01E4	LU	3287	P028CFS	4	2	

[1] Valid only for EDX Secondary SNA2

S$POOL Storage

1 Physical Unit	1 × 128 = 128 bytes
4 Logical Units	× 280 = 1120 bytes
______Remote Stations	× 32 = ______ bytes

Figure 55. Example 3: SNA Configuration Worksheet

EDX Definitions

Figure 56 on page 290 shows the EDX definitions for example 3. You may not want to map the common area into all partitions, as shown in the example. The partition in which EDX-SNA runs need not be mapped, unless Communications Facility programs also run in that partition.

```
*** EXAMPLE 3
***
$EDXDEF    CSECT
           SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),         X
                COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                  X
                INITMOD=(CSXINIT)
             •
             •
           PRINT2   TERMINAL DEVICE=4974,ADDRESS=21,PAGSIZE=66,        X
                TOPM=0,BOTM=65,END=YES
             •
           BSCLINE  ADDRESS=19,TYPE=MC,END=YES    3271 CONTROL
           EXIODEV  ADDRESS=0A,MAXDCB=7,RSB=4,END=YES    SNA
           END
```

Figure 56. Example 3: EDX Definitions

EDX Secondary SNA1 and Host System Definitions

This configuration must also be defined to EDX-SNA, to the host system access method (VTAM in this example), and to the host subsystem (CICS in this example).

Note: The sample definitions are not compatible with every SNA environment; you can use them as models to develop your customized definitions. Parameters have not been optimized, and there may be instances of values that restrict usage or degrade performance.

The examples show definitions of SNA 3274-SDLC and associated terminals to the network. No special parameters are needed in the network for the Communications Facility; you can use standard 3274 definitions. When you define your configuration, refer to the 3274 and SNA publications listed in "About This Book" on page iii and to the publications for your host system and subsystem.

Several parameters in the various definitions must agree with each other. The corresponding values defined in the figures in this section, the values defined to EDX in the EXIODEV statement (shown in the EDX definitions), and the values defined for Communications Facility SNA stations (shown on the SNA configuration worksheet) are as follows:

1 EDX-SNA SNAPU ADDRESS and EXIODEV ADDRESS. (This address has also been used as the PU station network address, which is recommended but not required.)

2 EDX-SNA SNAPU ENCODE and NCP/VTAM NRZI.

3 EDX-SNA SNAPU BUFSIZ must be greater than or equal to NCP/VTAM MAXDATA.

4 EDX-SNA SNAPU DCBNO, NCP/VTAM MAXOUT, and EXIODEV MAXDCB.

5 EDX-SNA SNALU number, NCP/VTAM LOCADDR, and LU station local address.

6 NCP VTAM MODETAB and VTAM mode table name.

7 NCP VTAM PU ADDR and secondary station address strapped on SDLC attachment card.

8 NCP/VTAM LU name and CICS NETNAME. (This name has also been used as the LU station name, and the NCP/VTAM PU name has been used as the PU station name, which is recommended but not required.)

9 VTAM LOGMODE name and Communications Facility SNA application mode (HOST MODE).

10 VTAM APPL name and Communications Facility SNA application name (HOST APPL).

```
NET1 CSECT ,             DEFINE THE NETWORK
     SNAPU RETRY=3,       NUMBER OF I/O RETRIES                X
           ADDRESS=0A,    DEVICE ADDRESS                       X 1

           CNCTYPE=PP,    CONNECTION TYPE                      X
           NORING=NO,     INCLUDE RING SUPPORT                 X
           RATE=FULL,     MODEM SPEED                          X
           ENCODE=NRZ,    MODEM ENCODING                       X 2

           PAD=NO,        SEND PAD CHAR ON INITIAL XMISSION    X
           TODTR=0,       DATA-TERMINAL-READY TIMEOUT          X
           TODSR=0,       DATA-SET-READY TIMEOUT               X
           TOCTS=0,       CLEAR-TO-SEND TIMEOUT                X
           TOHLA=0,       HOLD-LINE-ACTIVE TIMEOUT             X
           THRESH=4,      MINIMUM BUFFS TO SEND RECEIVE-READY X
           BUFSIZ=256,    SIZE OF LARGEST USER MESSAGE         X 3

           BUFNO=4,       NUMBER OF BUFFERS AT ACTIVATION      X
           DCBNO=7,       MAXIMUM FRAMES TO SEND OR RECEIVE    X 4

           STAXID=S00000 ID TO SEND IN RESPONSE TO XID CMD
*
     SNALU LU=1,          LOGICAL UNIT NUMBER                  X 5

           SENDBUF=1,     NUMBER OF SEND BUFFERS FOR LU        X
           RECVBUF=1,     NUMBER OF RECEIVE BUFFERS FOR LU     X
           CTE=1,         NUMBER OF CORRELATION TABLE ENTRIES X
           END=NO
*
     SNALU LU=2,          LOGICAL UNIT NUMBER                  X
           SENDBUF=1,     NUMBER OF SEND BUFFERS FOR LU        X
           RECVBUF=1,     NUMBER OF RECEIVE BUFFERS FOR LU     X
           CTE=1,         NUMBER OF CORRELATION TABLE ENTRIES X
           END=NO
*
     SNALU LU=3,          LOGICAL UNIT NUMBER                  X
           SENDBUF=1,     NUMBER OF SEND BUFFERS FOR LU        X
           RECVBUF=1,     NUMBER OF RECEIVE BUFFERS FOR LU     X
           CTE=1,         NUMBER OF CORRELATION TABLE ENTRIES X
           END=NO
*
     SNALU LU=4,          LOGICAL UNIT NUMBER                  X
           SENDBUF=1,     NUMBER OF SEND BUFFERS FOR LU        X
           RECVBUF=1,     NUMBER OF RECEIVE BUFFERS FOR LU     X
           CTE=1,         NUMBER OF CORRELATION TABLE ENTRIES X
           END=YES        LAST LOGICAL UNIT TO GENERATE
     END
```

Figure 57. Example 3: EDX Secondary SNA1 Definitions

EDX Secondary SNA1 Definitions

Figure 57 shows the EDX Secondary SNA1 definitions for example 3. The definitions support four logical units using 256-byte buffers. Only two buffers per logical unit are allocated; you may want more for best performance. See the *EDX Secondary SNA1 Guide* for information about how buffers and related parameters affect performance.

Figure 58 shows the definitions of physical and logical units to the network control program (NCP) and VTAM. They are defined as a PU type 2 (cluster controller) and its associated LUs. No information about LU type is specified to the NCP. VTAM unformatted session services table (USSTAB) specifications are not included, because the Communications Facility can't use this VTAM service. The SNA IOCP has its own logon technique that replaces the USS screen and facilities. If USSTAB is specified for your system, the Communications Facility won't use it, but it shouldn't cause any problem.

```
GCFSDLC GROUP LNCTL=SDLC,        SDLC LINE                      X
              POLLED=YES,        3274 ALWAYS POLLED             X
              NRZI=NO            NRZI                             2
*
LINE028 LINE  ADDRESS=028,       LINE ADDRESS IN 3705           X
              DISCNT=NO,         VTAM ONLY OPERAND              X
              MODETAB=CFTBL,     VTAM MODE TABLE                X 6

              DLOGMOD=LU2256,    DEF LOGON MODE FROM MODETAB X
              SPEED=4800,        LINE SPEED (SEE NCP DOCS)      X
              RETRIES=(5,1,20),  RECOVERY ATTEMPTS & PAUSE      X
              PACING=0,          PACING NOT NORMALLY USED       X
              VPACING=0,         FOR DISPLAYS OR PRINTERS       X
              ISTATUS=ACTIVE     LINE COMES UP WITH NCP
*
        SERVICE ORDER=(P028CFS)  ONLY 1 STATION HERE
*
P028CFS PU    ADDR=01,           MATCH SDLC CARD STRAPPING      X 7

              PUTYPE=2,          CLUSTER CONTROLLER             X
              MAXDATA=256,       SEGMENT SIZE                   X 3

              PASSLIM=7,         MAX CONSEC RU'S TO THIS PU     X
              MAXOUT=6           MAX FRAMES BEFORE RESPONSE       4

*
LCFS01  LU    LOCADDR=1  5   8

LCFS02  LU    LOCADDR=2
LCFS03  LU    LOCADDR=3
LCFS04  LU    LOCADDR=4
```

Figure 58. Example 3: Definitions to NCP and VTAM

Figure 59 on page 294 shows the definition of a VTAM mode table. The SNA application modes used with the Communications Facility correspond to SNA session parameters that are defined in SNA bind commands. A given mode name refers to an image of a specific bind command stored on the host. You must define all modes used with the Communications Facility. The sample mode table includes entries for LU types 1, 2, and 3, as suggested in the *3274 Guide*. The restriction the Communications Facility places on mode values is that only 24 × 80 screen size is supported for both default and alternate screens. For subtype 3286 and 3287 LU3 stations, and for subtype 3278 and 3279 LU2 devices, the Communications Facility will accept the bind, and assume the real device can handle the alternate size. See the *3274 Guide* for details of valid bind images.

```
CFTBL    MODETAB                                                      6

*                     STANDARD 3274 LU1 FROM 3274 PUBLICATION
STD32741 MODEENT LOGMODE=STD32741,                                  X 9

               FMPROF=X'03',                                        X
               TSPROF=X'03',                                        X
               PRIPROT=X'B1',                                       X
               SECPROT=X'90',                                       X
               COMPROT=X'3080',                                     X
               RUSIZES=X'8587',                                     X
               PSERVIC=X'010000000000000000000000'
*                     STANDARD 3274 LU3 FROM 3274 PUBLICATION
STD32743 MODEENT LOGMODE=STD32743,                                  X
               FMPROF=X'03',                                        X
               TSPROF=X'03',                                        X
               PRIPROT=X'B1',                                       X
               SECPROT=X'90',                                       X
               COMPROT=X'3080',                                     X
               RUSIZES=X'8587',                                     X
               PSERVIC=X'030000000000000000000000'
*                      LU2 WITH 256-BYTE RU SIZE
LU2256   MODEENT LOGMODE=LU2256,                                    X
               FMPROF=X'03',                                        X
               TSPROF=X'03',                                        X
               PRIPROT=X'B1',                                       X
               SECPROT=X'90',                                       X
               COMPROT=X'3080',                                     X
               RUSIZES=X'8585',                                     X
               PSERVIC=X'020000000000000000000200'
         MODEEND
         END
```

Figure 59. Example 3: VTAM Mode Table

Figure 60 shows the definition of application programs to VTAM. You must define all SNA application names used with the Communications Facility.

```
IMS      APPL  AUTH=(ACQ,PASS)
*
CICS     APPL  AUTH=(ACQ,VPACE)        10

*
TSO      APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=10
TSO0001  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0002  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0003  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0004  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0005  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0006  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0007  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0008  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0009  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0010  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
```

Figure 60. Example 3: Definition of Applications to VTAM

Figure 61 shows a CICS terminal control table with entries for the two type 2 logical units and the type 3 logical unit used in this sample configuration.

```
LC01     DFHTCT TYPE=TERMINAL,                                        X
                TRMTYPE=LUTYPE2,        TERMINAL                      X
                TRMMODL=2,                                            X
                TRMIDNT=LC01,                                         X
                BUFFER=0,                                             X
                RUSIZE=256,                                           X
                TIOAL=(256,2048),                                     X
                NETNAME=LCFS01,                                       X 8

                CHNASSY=YES,                                          X
                BRACKET=YES,                                          X
                TRMSTAT=(TRANSCEIVE),                                 X
                ACCMETH=VTAM,                                         X
                GMMSG=YES,                                            X
                RELREQ=(YES,YES)
*
LC02     DFHTCT TYPE=TERMINAL,                                        X
                TRMTYPE=LUTYPE2,        TERMINAL                      X
                TRMMODL=2,                                            X
                TRMIDNT=LC02,                                         X
                BUFFER=0,                                             X
                RUSIZE=256,                                           X
                TIOAL=(256,2048),                                     X
                NETNAME=LCFS02,                                       X
                CHNASSY=YES,                                          X
                BRACKET=YES,                                          X
                TRMSTAT=(TRANSCEIVE),                                 X
                ACCMETH=VTAM,                                         X
                GMMSG=YES,                                            X
                RELREQ=(YES,YES)
*
*
LC03     DFHTCT TYPE=TERMINAL,                                        X
                TRMTYPE=LUTYPE3,        PRINTER                       X
                TRMMODL=2,                                            X
                TRMIDNT=LC03,                                         X
                BUFFER=0,                                             X
                NETNAME=LCFS03,                                       X
                TRMSTAT=(TRANSCEIVE),                                 X
                ACCMETH=VTAM
```

Figure 61. Example 3: CICS Terminal Control Table

Example 4: EDX Secondary SNA2 Connection to a Host

Figure 62 on page 297 shows a configuration in which a Series/1 is connected to a host computer by an EDX Secondary SNA2 connection. The Series/1 appears to the host as two SNA physical units, two SNA type 2 logical units (terminals), and one SNA type 1 logical unit (printer). The real devices are two 4980 display stations and one 4974 printer.

Message Flow

To achieve the flow of messages between the Series/1 terminals and the host, each terminal or device station and its associated SNA logical unit station are linked to each other. When an operator enters data at the terminal represented by station T4980A, the 4978/4980 data stream IOCP, $.IO0670, reads the data and sends it to T4980A's link, station LU0101. Subsequently, the SNA IOCP, $.IO14E8, retrieves the message from LU0101's queue and calls on EDX Secondary SNA2 to write it to the SDLC line.

$.IO14E8 also receives data from the host through the SDLC lines, and disposes of it according to the local address and PU that you defined for the SNA logical unit stations. For example, data for logical unit 2 on the SDLC line with device address X'0A', is intended for station LU0102, so it is sent to that station's link, T4980B.

Station Definitions

Figure 63 on page 298, Figure 64 on page 299, and Figure 65 on page 300 show the station definitions.

Notice the terminal type of the SNA logical units in Figure 65 on page 300. It must correspond to the type of station that the logical unit is linked to, either a real 3270 terminal or a Series/1 device being used as if it were a 3277, a 3286, or a 3287.

Notice the PU definitions in Figure 65 on page 300. The PU number specified in each station must correspond to the PU defined to EDX Secondary SNA2 for that station. For example, SNA PUs 1 and 2 defined in Figure 67 on page 303 become PU01 and PU02 respectively.

Notice also the logon information in Figure 65 on page 300. Three host applications are specified: #1 is for TSO/terminal communication, #2 is for IMS/printer communication, and #3 is for CICS/printer communication. An operator at the 4980 display device linked to station LU0101 is automatically connected to TSO when the terminal and LU station are started.

An operator at the 4980 display device linked to station LU0102 is presented with a logon prompt screen when the terminal and LU stations are started. ("Series/1-to-Host SNA IOCP, $.IO14E8" on page 169 includes a picture of a logon screen.) The screen shows the applications specified in the PU. The terminal user can select option 1 to log on to TSO or option 4 to log on to any other SNA application defined in the host.

The 4974 printer linked to station LU0102 is automatically connected to IMS when the printer and LU stations are started. IMS uses SNA type 1 logical units for printer support, so the printer station will receive SNA character string (SCS) data.

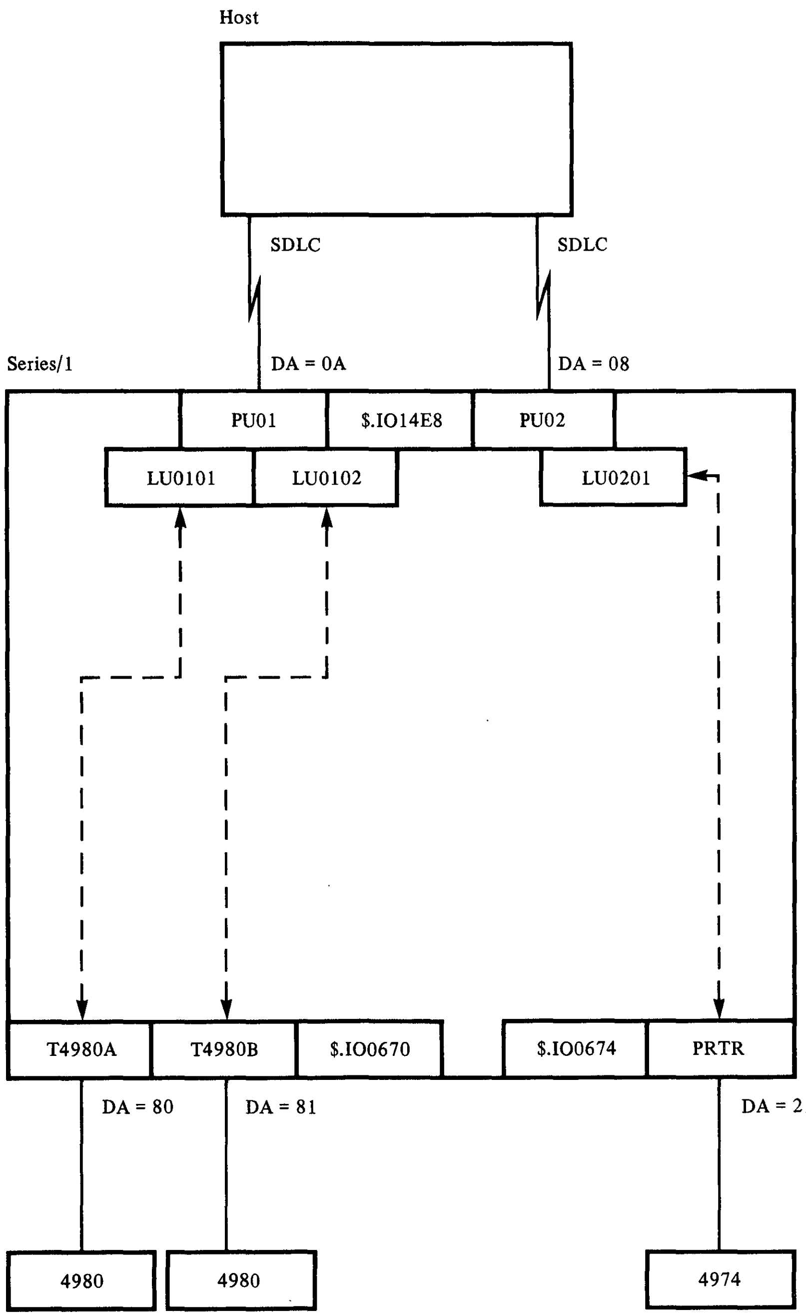


Figure 62. Example 4: EDX Secondary SNA2 Connection to a Host

You must include LU-1 support in $.IO0674 and set the printer station to SCS mode.

User/Message/Alias/Volume Configuration Worksheet

System: Example 4: EDX Secondary SNA2 Connection to a Host
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
SNA IOCP	$.IO14E8	01AA	USER	
4978/4980 Data Stream IOCP	$.IO0670	01AB	USER	
Printer IOCP	$.IO0674	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
____Alias Stations	× 40 = ______ bytes
____Remote Stations	× 32 = ______ bytes
____Volume Stations	× 104 = ______ bytes

Figure 63. Example 4: User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 4: EDX Secondary SNA2 Connection to a Host
Node Assignment: 01

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
Printer	PRTR	0121	DEVICE	Printer	2500	N	4040	0000		
4980	T4980A	0180	DEVICE	4978	2500		4040	0000		
4980	T4980B	0181	DEVICE	4978	2500		4040	0000		

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485's, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

______3101F Devices	× 160 = ______ bytes
1 4978/Printer Device	× 264 = 264 bytes
______3101 Devices	× 304 = ______ bytes
______Remote Stations	× 32 = ______ bytes

Figure 64. Example 4: Device Configuration Worksheet

SNA Configuration Worksheet

System: Example 4: EDX Secondary SNA2 Connection to a Host
Node Assignment: 01

PU Definition

Description	Name	NA	Type	Buffer Size	PU #[1]	Host Application	Host Mode
SNA physical unit	PU01	010A	PU	2500	1	#1: TSO	LU2256
						#2: IMS	STD32741
						#3: CICS	STD32743
SNA physical unit	PU02	0108	PU	2500	2	#1: TSO	LU2256
						#2: IMS	STD32741
						#3: CICS	STD32743

LU Definitions

Description	Name	NA	Type	Term Type	PU Name	Local Address	Logon ID	User Data
Emulated terminal	LU0101	01E1	LU	3277	PU01	1	1	
Emulated terminal	LU0102	01E2	LU	3277	PU01	2		
Emulated printer	LU0201	01E4	LU	3287	PU02	1	2	

[1] Valid only for EDX Secondary SNA2

S$POOL Storage

1 Physical Unit	1 × 128 = 128 bytes
4 Logical Units	× 120 = 480 bytes
______Remote Stations	× 32 = ______ bytes

Figure 65. Example 4: SNA Configuration Worksheet

EDX Definitions

Figure 66 on page 301 shows the EDX definitions for example 4. You may not want to map the common area into all partitions, as shown in the example. The partition in which EDX-SNA runs does not need to be mapped, unless Communications Facility programs also run in that partition.

```
*** EXAMPLE 4
***
$EDXDEF   CSECT
          SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),         X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                  X
               INITMOD=(CSXINIT)
            •
            •
          PRINT2   TERMINAL DEVICE=4974,ADDRESS=21,PAGSIZE=66,        X
               TOPM=0,BOTM=65,END=YES
            •
          BSCLINE  ADDRESS=19,TYPE=MC,END=YES   3271 CONTROL
          EXIODEV  ADDRESS=0A,MAXDCB=7,RSB=4,END=NO    SNA
          EXIODEV  ADDRESS=08,MAXDCB=7,RSB=4,END=YES   SNA
          END
```

Figure 66. Example 4: EDX Definitions

EDX Secondary SNA2 and Host System Definitions

This configuration must also be defined to EDX Secondary SNA2, to the host system access method (VTAM in this example), and to the host subsystem (CICS in this example).

Note: The sample definitions are not compatible with every SNA environment; you can use them as models to develop your customized definitions. Parameters have not been optimized, and there may be instances of values that restrict usage or degrade performance.

The examples show definitions of SNA 3274-SDLC and associated terminals to the network. No special parameters are needed in the network for the Communications Facility; you can use standard 3274 definitions. When you define your configuration, refer to the 3274 and SNA publications listed in "About This Book" on page iii and to the publications for your host system and subsystem.

Several parameters in the various definitions must agree with each other. The corresponding values defined in the figures in this section, the values defined to EDX in the EXIODEV statement (shown in the EDX definitions), and the values defined for Communications Facility SNA stations (shown on the SNA configuration worksheet) are as follows:

1 EDX-SNA SNAPU ADDRESS and EXIODEV ADDRESS. (This address has also been used as the PU station network address, which is recommended but not required.)

2 EDX-SNA SNAPU ENCODE and NCP/VTAM NRZI.

3 EDX-SNA SNAPU BUFSIZ must be greater than or equal to NCP/VTAM MAXDATA.

4 EDX-SNA SNAPU DCBNO, NCP/VTAM MAXOUT, and EXIODEV MAXDCB.

5 EDX-SNA SNALU number, NCP/VTAM LOCADDR, and LU station local address.

6 NCP VTAM MODETAB and VTAM mode table name.

7 NCP VTAM PU ADDR and secondary station address strapped on SDLC attachment card.

8 NCP/VTAM LU name and CICS NETNAME.

9 VTAM LOGMODE name and Communications Facility SNA application mode (HOST MODE).

10 VTAM APPL name and Communications Facility SNA application name (HOST APPL).

11 EDX SNA PU number and the PU station's PU number. (NET1 and NET2 indicate that these SNA definitions are for PUs 1 and 2, respectively.)

```
NET1 CSECT,                 DEFINE THE NETWORK                     1
     SNAPU RETRY=3,         NUMBER OF I/O RETRIES                X
           ADDRESS=0A,      DEVICE ADDRESS                       X 1

           CNCTYPE=PP,      CONNECTION TYPE                      X
           NORING=NO,       INCLUDE RING SUPPORT                 X
           RATE=FULL,       MODEM SPEED                          X
           ENCODE=NRZ,      MODEM ENCODING                       X 2

           PAD=NO,          SEND PAD CHAR ON INITIAL XMISSION    X
           TODTR=0,         DATA-TERMINAL-READY TIMEOUT          X
           TODSR=0,         DATA-SET-READY TIMEOUT               X
           TOCTS=0,         CLEAR-TO-SEND TIMEOUT                X
           TOHLA=0,         HOLD-LINE-ACTIVE TIMEOUT             X
           THRESH=4,        MINIMUM BUFFS TO SEND RECEIVE-READY  X
           BUFSIZ=256,      SIZE OF LARGEST USER MESSAGE         X 3

           BUFNO=4,         NUMBER OF BUFFERS AT ACTIVATION      X
           DCBNO=7,         MAXIMUM FRAMES TO SEND OR RECEIVE    X 4

           STAXID=S00000 ID TO SEND IN RESPONSE TO XID CMD
*
     SNALU LU=1,            LOGICAL UNIT NUMBER                  X 5

           SENDBUF=1,       NUMBER OF SEND BUFFERS FOR LU        X
           RECVBUF=1,       NUMBER OF RECEIVE BUFFERS FOR LU     X
           CTE=1,           NUMBER OF CORRELATION TABLE ENTRIES  X
           END=NO
*
     SNALU LU=2,            LOGICAL UNIT NUMBER                  X
           SENDBUF=1,       NUMBER OF SEND BUFFERS FOR LU        X
           RECVBUF=1,       NUMBER OF RECEIVE BUFFERS FOR LU     X
           CTE=1,           NUMBER OF CORRELATION TABLE ENTRIES  X
           END=NO
*
     SNALU LU=3,            LOGICAL UNIT NUMBER                  X
           SENDBUF=1,       NUMBER OF SEND BUFFERS FOR LU        X
           RECVBUF=1,       NUMBER OF RECEIVE BUFFERS FOR LU     X
           CTE=1,           NUMBER OF CORRELATION TABLE ENTRIES  X
           END=NO
*
     SNALU LU=4,            LOGICAL UNIT NUMBER                  X
           SENDBUF=1,       NUMBER OF SEND BUFFERS FOR LU        X
           RECVBUF=1,       NUMBER OF RECEIVE BUFFERS FOR LU     X
           CTE=1,           NUMBER OF CORRELATION TABLE ENTRIES  X
           END=YES          LAST LOGICAL UNIT TO GENERATE
     END
```

Figure 67. Example 4: EDX Secondary SNA2 Definitions

```
NET2 CSECT,              DEFINE THE NETWORK                     11
     SNAPU RETRY=3,      NUMBER OF I/O RETRIES                 X
           ADDRESS=08,   DEVICE ADDRESS                        X 1

           CNCTYPE=PP,   CONNECTION TYPE                       X
           NORING=NO,    INCLUDE RING SUPPORT                  X
           RATE=FULL,    MODEM SPEED                           X
           ENCODE=NRZ,   MODEM ENCODING                        X 2

           PAD=NO,       SEND PAD CHAR ON INITIAL XMISSION     X
           TODTR=0,      DATA-TERMINAL-READY TIMEOUT           X
           TODSR=0,      DATA-SET-READY TIMEOUT                X
           TOCTS=0,      CLEAR-TO-SEND TIMEOUT                 X
           TOHLA=0,      HOLD-LINE-ACTIVE TIMEOUT              X
           THRESH=4,     MINIMUM BUFFS TO SEND RECEIVE-READY   X
           BUFSIZ=256,   SIZE OF LARGEST USER MESSAGE          X 3

           BUFPOOL=YES,  BUFFER POOLING                        X
           STKNUM=5,     NUMBER OF STACKS FOR LUS              X
           SBUFNO=4      CONCURRENT SEND BUFFERS               X

           BUFNO=4,      NUMBER OF BUFFERS AT ACTIVATION       X
           DCBNO=7,      MAXIMUM FRAMES TO SEND OR RECEIVE     X 4

           STAXID=S00000,ID TO SEND IN RESPONSE TO XID CMD
*
     SNALU LU=1,         LOGICAL UNIT NUMBER                   X 5

           SENDBUF=1,    NUMBER OF SEND BUFFERS FOR LU         X
           RECVBUF=1,    NUMBER OF RECEIVE BUFFERS FOR LU      X
           CTE=1,        NUMBER OF CORRELATION TABLE ENTRIES   X
           HOLDBND=YES,  HOLD BIND UNTIL NETINIT ISSUED        X
           END=NO
*
     SNALU LU=2,         LOGICAL UNIT NUMBER                   X
           SENDBUF=1,    NUMBER OF SEND BUFFERS FOR LU         X
           RECVBUF=1,    NUMBER OF RECEIVE BUFFERS FOR LU      X
           CTE=1,        NUMBER OF CORRELATION TABLE ENTRIES   X
           END=YES
     END
```

Figure 68. Example 4: EDX Secondary SNA2 Definitions

EDX Secondary SNA2 Definitions

Figure 67 on page 303 and Figure 68 show the EDX Secondary SNA2 definitions for example 4. The definitions support four logical units using 256-byte buffers. Only two buffers per logical unit are allocated; you may want more for best performance. See the *EDX Secondary SNA2 Guide* for information about how buffers and related parameters affect performance.

Figure 69 and Figure 70 on page 306 show the definitions of physical and logical units to the network control program (NCP) and VTAM. They are defined as a PU type 2 (cluster controller) and its associated LUs. No information about LU type is specified to the NCP. VTAM unformatted session services table (USSTAB) specifications are not included, because the Communications Facility can't use this VTAM service. The SNA IOCP has its own logon technique that replaces the USS screen and facilities. If USSTAB is specified for your system, the Communications Facility won't use it, but it shouldn't cause any problem.

```
GCFSDLC1 GROUP LNCTL=SDLC,       SDLC LINE                  X
               POLLED=YES,       3274 ALWAYS POLLED         X
               NRZI=NO           NRZI                         2

*
LINE037  LINE  ADDRESS=037,      LINE ADDRESS IN 3705       X
               DISCNT=NO,        VTAM ONLY OPERAND          X
               MODETAB=CFTBL,    VTAM MODE TABLE            X 6

               DLOGMOD=TSO32742, DEF LOGON MODE FROM MODETAB X
               SPEED=4800,       LINE SPEED (SEE NCP DOCS)  X
               RETRIES=(5,1,20), RECOVERY ATTEMPTS & PAUSE  X
               PACING=0,         PACING NOT NORMALLY USED   X
               VPACING=0,        FOR DISPLAYS OR PRINTERS   X
               ISTATUS=ACTIVE    LINE COMES UP WITH NCP
*
        SERVICE ORDER=(P037CFS) ONLY 1 STATION HERE
*
P028CFS PU     ADDR=0D,          MATCH SDLC CARD STRAPPING  X 7

               PUTYPE=2,         CLUSTER CONTROLLER         X
               MAXDATA=1024,     SEGMENT SIZE               X 3

               PASSLIM=55,       MAX CONSEC RU'S TO THIS PU X
               MAXOUT=7          MAX FRAMES BEFORE RESPONSE   4

*
L37S01  LU     LOCADDR=1 5  8

L37S02  LU     LOCADDR=2
L37S03  LU     LOCADDR=3
L37S04  LU     LOCADDR=4
```

Figure 69. Example 4: Definitions to NCP and VTAM

```
GCFSDLC1 GROUP LNCTL=SDLC,        SDLC LINE                     X
               POLLED=YES,        3274 ALWAYS POLLED            X
               NRZI=NO            NRZI                            2

*
LINE037  LINE  ADDRESS=037,       LINE ADDRESS IN 3705          X
               DISCNT=NO,         VTAM ONLY OPERAND             X
               MODETAB=CFTBL,     VTAM MODE TABLE               X 6

               DLOGMOD=TSO32742,  DEF LOGON MODE FROM MODETAB   X
               SPEED=4800,        LINE SPEED (SEE NCP DOCS)     X
               RETRIES=(5,1,20),  RECOVERY ATTEMPTS & PAUSE     X
               PACING=0,          PACING NOT NORMALLY USED      X
               VPACING=0,         FOR DISPLAYS OR PRINTERS      X
               ISTATUS=ACTIVE     LINE COMES UP WITH NCP
*
        SERVICE ORDER=(P037CFS) ONLY 1 STATION HERE
*
P028CFS PU     ADDR=0D,           MATCH SDLC CARD STRAPPING     X 7

               PUTYPE=2,          CLUSTER CONTROLLER            X
               MAXDATA=1024,      SEGMENT SIZE                  X 3

               PASSLIM=55,        MAX CONSEC RU'S TO THIS PU    X
               MAXOUT=7           MAX FRAMES BEFORE RESPONSE      4

*
L37S01   LU    LOCADDR=1 5   8

L37S02   LU    LOCADDR=2
   •              •
   •              •
```

Figure 70. Example 4: Definitions to NCP and VTAM

Figure 71 on page 307 shows the definition of a VTAM mode table. The SNA application modes used with the Communications Facility correspond to SNA session parameters that are defined in SNA bind commands. A given mode name refers to an image of a specific bind command stored on the host. You must define all modes used with the Communications Facility. The sample mode table includes entries for LU types 1, 2, and 3, as suggested in the *3274 Guide*. The restriction the Communications Facility places on mode values is that only 24 by 80 screen size is supported for both default and alternate screens. For subtype 3286 and 3287 LU3 stations, and for subtype 3278 and 3279 LU2 devices, the Communications Facility will accept the bind, and assume that the real device can handle the alternate size. See the *3274 Guide* for details of valid bind images.

```
CFTBL    MODETAB                                                    6
*                    STANDARD 3274 LU1 FROM 3274 PUBLICATION
STD32741 MODEENT LOGMODE=STD32741,                                X 9
               FMPROF=X'03',                                      X
               TSPROF=X'03',                                      X
               PRIPROT=X'B1',                                     X
               SECPROT=X'90',                                     X
               COMPROT=X'3080',                                   X
               RUSIZES=X'8587',                                   X
               PSERVIC=X'010000000000000000000000'
*                    STANDARD 3274 LU3 FROM 3274 PUBLICATION
STD32743 MODEENT LOGMODE=STD32743,                                X
               FMPROF=X'03',                                      X
               TSPROF=X'03',                                      X
               PRIPROT=X'B1',                                     X
               SECPROT=X'90',                                     X
               COMPROT=X'3080',                                   X
               RUSIZES=X'8587',                                   X
               PSERVIC=X'030000000000000000000000'
*                     LU2 WITH 256-BYTE RU SIZE
LU2256   MODEENT LOGMODE=LU2256,                                  X
               FMPROF=X'03',                                      X
               TSPROF=X'03',                                      X
               PRIPROT=X'B1',                                     X
               SECPROT=X'90',                                     X
               COMPROT=X'3080',                                   X
               RUSIZES=X'8585',                                   X
               PSERVIC=X'020000000000000000000200'
         MODEEND
         END
```

Figure 71. Example 4: VTAM Mode Table

Figure 72 shows the definition of application programs to VTAM. You must define all SNA application names used with the Communications Facility.

```
IMS      APPL   AUTH=(ACQ,PASS)
*
CICS     APPL   AUTH=(ACQ,VPACE)        10

*
TSO      APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=10
TSO0001  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0002  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0003  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0004  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0005  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0006  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0007  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0008  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0009  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0010  APPL   AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
```

Figure 72. Example 4: Definition of Applications to VTAM

Figure 73 on page 308 shows a CICS terminal control table with entries for the two type 2 logical units used in this sample configuration.

```
LC01     DFHTCT TYPE=TERMINAL,                                      X
               TRMTYPE=LUTYPE2,        TERMINAL                     X
               TRMMODL=2,                                           X
               TRMIDNT=LC01,                                        X
               BUFFER=0,                                            X
               RUSIZE=256,                                          X
               TIOAL=(256,2048),                                    X
               NETNAME=LCFS01,                                      X

               CHNASSY=YES,                                         X
               BRACKET=YES,                                         X
               TRMSTAT=(TRANSCEIVE),                                X
               ACCMETH=VTAM,                                        X
               GMMSG=YES,                                           X
               RELREQ=(YES,YES)
*
LC02     DFHTCT TYPE=TERMINAL,                                      X
               TRMTYPE=LUTYPE2,        TERMINAL                     X
               TRMMODL=2,                                           X
               TRMIDNT=LC02,                                        X
               BUFFER=0,                                            X
               RUSIZE=256,                                          X
               TIOAL=(256,2048),                                    X
               NETNAME=LCFS02,                                      X
               CHNASSY=YES,                                         X
               BRACKET=YES,                                         X
               TRMSTAT=(TRANSCEIVE),                                X
               ACCMETH=VTAM,                                        X
               GMMSG=YES,                                           X
               RELREQ=(YES,YES)
*
*
LC03     DFHTCT TYPE=TERMINAL,                                      X
               TRMTYPE=LUTYPE2,        TERMINAL                     X
               TRMMODL=2,                                           X
               TRMIDNT=LC03,                                        X
               BUFFER=0,                                            X
               RUSIZE=256,                                          X
               TIOAL=(256,2048),                                    X
               NETNAME=LCFS03,                                      X
               CHNASSY=YES,                                         X
               BRACKET=YES,                                         X
               TRMSTAT=(TRANSCEIVE),                                X
               ACCMETH=VTAM,                                        X
               GMMSG=YES,                                           X
               RELREQ=(YES,YES)
```

Figure 73. Example 4: CICS Terminal Control Table

Example 5: Local Communications Controller Connection between Series/1s

Figure 74 on page 311 shows a configuration in which three Series/1s are attached to a Local Communications Controller. The application program in node 1 communicates with users at 3101 terminals in any of the three nodes. The 3101 terminals are attached to the Series/1s by feature programmable adapters.

Message Flow

To achieve the flow of messages between the terminals in node 1 and the application program, the device stations are linked to the program station. There can't be a link from the program station to the device stations, since a station can be linked to only one station. A program that communicates with multiple users must be written to respond to a user by sending a message to the station from which it received a message. Sample program $.CFMENU, which is described in the *Programmer's Guide,* illustrates this kind of coding.

To understand the flow of messages between the terminals in nodes 2 and 3 and the application program in node 1, first look at the station definition worksheets for the three nodes. Notice that each node has a unique node assignment.

In node 1, all stations that the application program will communicate with are defined on the device configuration worksheet; the network address distinguishes the local 3101s from the remote 3101s. If you prefer, you can define the remote 3101s as MSG stations. When a remote station is started, a vector station block is created; it contains only the station's name, network address, and link.

In nodes 2 and 3, the remote application program is defined on the user/message/alias configuration worksheet; its network address identifies it as a remote station.

In all three nodes, the remote nodes to which messages will be sent are defined on the line/terminal/node configuration worksheet. Notice the network addresses on those worksheets. The second byte of the line station network address is irrelevant (so long as it is unique within the node). It is the device address that specifies the Local Communications Controller subchannel 0 address. The second byte of the node station network addresses must be 00. As defined, nodes 2 and 3 can't communicate with each other. To make that communication possible, define node 3 in node 2 and vice versa.

When these stations have been defined, linked as shown in the diagram of the configuration, and started, a display of the active stations in node 2 will show this information:

STATION	TYPE	NA	LINK
$.DISP	0000	0200	0000
LCCDEV	0AB0	02C0	0000
NODE1	1040	0100	0000
TERM2A	0630	0210	01AC
TERM2B	0630	0220	01AC
APROG	0800	01AC	0000

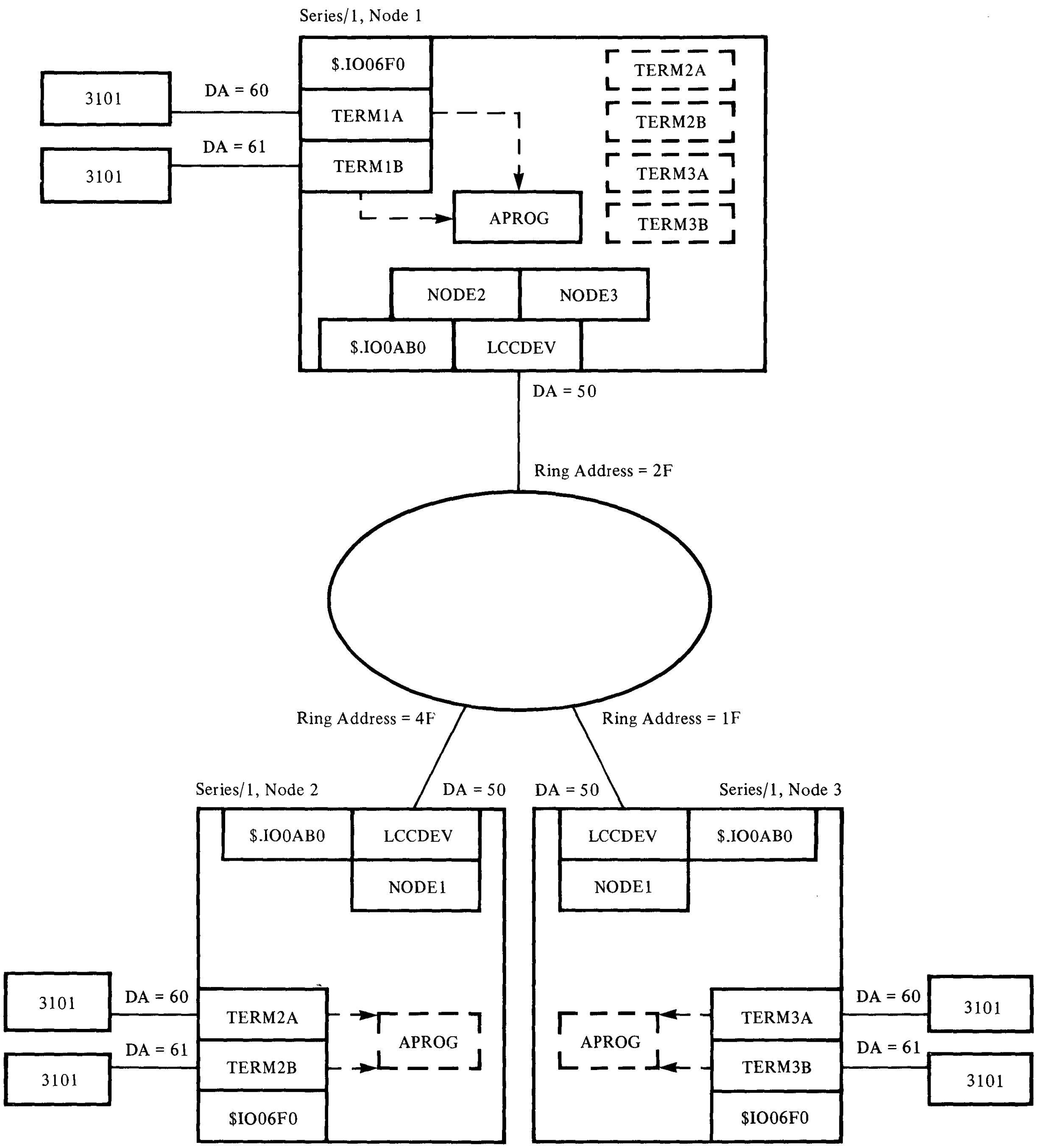


Figure 74. Example 5: Local Communications Controller Connection between Series/1s

Notice the following:

- The network address of $.DISP (the system station) shows that the node assignment is 02. This is the result of issuing a CP SET NODE command.
- The remote program station (APROG) has a vector station block (type 08). This is the result of starting a station whose type is other than NODE and whose network address is not in this node.

Now let's see how a message from TERM2A gets routed to the application program in node 1. As always, the I/O control program sends the message to the origin station's link, APROG, whose network address is 01AC. The message dispatcher determines that the address is not in this node, sets the second byte to 00, and looks for a station with that network address (0100). Station NODE1 satisfies the search, so the message dispatcher puts the message on NODE1's queue.

The method of routing messages between nodes accounts for the restriction that you can't have more than one active BSC point-to-point connection, Local Communications Controller connection, or X.25 connection through circuit stations with usage CF between two Series/1s. For all these connections, a station with network address *nn*00 is required to route messages to node *nn*, and you can't start more than one station with a given network address.

Subsequently, the Local Communications Controller IOCP, $.IO0AB0, retrieves the message from NODE1's queue and sends it over to the Local Communications Controller ring managed by the associated line station, LCCDEV. The ring address determines which node receives the message. In this case, station NODE1 has ring address 2F, so $.IO0AB0 in node 1 receives the message. $.IO0AB0 sends the message to the message dispatcher. The message carries its original routing information showing that the origin is 0210 and the destination is 01AC. The message dispatcher determines that the destination network address is in this node, so it puts the message on APROG's queue. The response from APROG to TERM2A is routed in a similar manner.

In summary, to enable communication between nodes on a Local Communications Controller, you must:

- Give each node a unique node assignment.
- In each node, define the remote stations to which local stations will send messages. Be sure that the network address is the same as that of the corresponding real station in the remote node. The station names need not be the same.
- In each node, define a NODE station for each node to which local stations will send messages.
- When you start the Communications Facility, issue a CP SET NODE command to set the node assignment before starting any remote stations.

Station Definitions

Figure 75 through Figure 83 on page 319 show the station definitions for the three nodes in this configuration.

User/Message/Alias/Volume Configuration Worksheet

System: Example 5: Local Communications Controller, Node 1
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
3101-MFA/FPCA IOCP	$.IO06F0	01AA	USER	
Local Communications Controller IOCP	$.IO0AB0	01AB	USER	
Application program	APROG	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

3 User/Msg Stations	× 88 = 264 bytes
_____Alias Stations	× 40 = _____ bytes
_____Remote Stations	× 32 = _____ bytes
_____Volume Stations	× 104 = _____ bytes

Figure 75. Example 5: Node 1 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 5: Local Communications Controller, Node 1
Node Assignment: 01

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM1A	0110	DEVICE	3101F	2500		4040	$SYSPRTR	9600	N
3101	TERM1B	0120	DEVICE	3101F	2500		4040	$SYSPRTR	9600	N
Remote 3101	TERM2A	0210	DEVICE	3101F	2500		4040	$SYSPRTR	9600	N
Remote 3101	TERM2B	0220	DEVICE	3101F	2500		4040	$SYSPRTR	9600	N
Remote 3101	TERM3A	0310	DEVICE	3101F	2500		4040	$SYSPRTR	9600	N
Remote 3101	TERM3B	0320	DEVICE	3101F	2500		4040	$SYSPRTR	9600	N

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

_____3101F Devices	× 160 = ______ bytes
_____4978/Printer Devices	× 264 = ______ bytes
2 3101 Devices	× 304 = 608 bytes
4 Remote Stations	× 32 = 128 bytes

Figure 76. Example 5: Node 1 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 5: Local Communications Controller, Node 1
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller ring	LCCDEV	01C0	LINE	LCC	2500	50		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 2	NODE2	0200	NODE	CF	4F	LCCDEV			
Node 3	NODE3	0300	NODE	CF	1F	LCCDEV			

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

____HDLC Lines	× 112 = _____ bytes
1 Other Line	× 264 = 264 bytes
2 Terminals/Nodes	× 88 = 176 bytes
____Remote Stations	× 32 = _____ bytes

Figure 77. Example 5: Node 1 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 5: Local Communications Controller, Node 2
Node Assignment: 02

Description	Name	NA	Type	Alias[1]
3101-MFA/FPCA IOCP	$.IO06F0	02AA	USER	
Local Communications Controller IOCP	$.IO0AB0	02AB	USER	
Remote application program	APROG	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
____Alias Stations	× 40 = ____ bytes
1 Remote Station	× 32 = 32 bytes
____Volume Stations	× 104 = ____ bytes

Figure 78. Example 5: Node 2 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 5: Local Communications Controller, Node 2
Node Assignment: 02

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM2A	0210	DEVICE	3101	2500		4040	$SYSPRTR		
3101	TERM2B	0220	DEVICE	3101	2500		4040	$SYSPRTR		

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

____3101F Devices	× 160 = ____ bytes
____4978/Printer Devices	× 264 = ____ bytes
2 3101 Devices	× 304 = 608 bytes
____Remote Stations	× 32 = ____ bytes

Figure 79. Example 5: Node 2 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 5: Local Communications Controller, Node 2
Node Assignment: 02

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller ring	LCCDEV	02C0	LINE	LCC	2500	50		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 1	NODE1	0100	NODE	CF	2F	LCCDEV			

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = _____ bytes
1 Other Line	× 264 = 264 bytes
1 Terminal/Node	× 88 = 88 bytes
_____Remote Stations	× 32 = _____ bytes

Figure 80. Example 5: Node 2 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 5: Local Communications Controller Node 3
Node Assignment: 03

Description	Name	NA	Type	Alias[1]
3101-MFA/FPCA IOCP	$.IO06F0	03AA	USER	
Local Communications Controller IOCP	$.IO0AB0	03AB	USER	
Remote application program	APROG	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations × 88 = 176 bytes
_____Alias Stations × 40 = ______ bytes
1 Remote Station × 32 = 32 bytes
_____Volume Stations × 104 = ______ bytes

Figure 81. Example 5: Node 3 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 5: Local Communications Controller Node 3
Node Assignment: 03

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM3A	0310	DEVICE	3101F	2500		4040	$SYSPRTR	9600	N
3101	TERM3B	0320	DEVICE	3101F	2500		4040	$SYSPRTR	9600	N

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

_____3101F Devices × 160 = ______ bytes
_____4978/Printer Devices × 264 = ______ bytes
2 3101 Devices × 304 = 608 bytes
_____Remote Stations × 32 = ______ bytes

Figure 82. Example 5: Node 3 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 5: Local Communications Controller, Node 3
Node Assignment: 03

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller ring	LCCDEV	03C0	LINE	LCC	2500	50		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 1	NODE1	0100	NODE	CF	2F	LCCDEV			

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = _____ bytes
1 Other Line	× 264 = 264 bytes
1 Terminal/Node	× 88 = 88 bytes
_____Remote Stations	× 32 = _____ bytes

Figure 83. Example 5: Node 3 Line/Terminal/Node Configuration Worksheet

EDX Definitions

Figure 84 on page 320 shows the EDX definitions for any of the three nodes in this example. Notice that the 3101s are not defined to EDX, so they can be used only as Communications Facility terminals. If you also want to use them as EDX terminals, use TERMINAL statements to define them.

```
*** EXAMPLE 5: NODE 1, 2, OR 3
***
$EDXDEF   CSECT
          SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                   X
               INITMOD=(CSXINIT)
            •
            •
$SYSPRTR  TERMINAL DEVICE=4973,ADDRESS=01,END=YES
            •
            •
          LCC      ADDRESS=50                    SUBCHANNEL 0
          LCC      ADDRESS=51                    SUBCHANNEL 1
          LCC      ADDRESS=52,END=YES            SUBCHANNEL 2
          END
```

Figure 84. Example 5: EDX Definitions

Example 6: BSC Point-to-Point Connection between Series/1s

Figure 85 shows a configuration in which two Series/1s are connected by a BSC point-to-point line. The application program in node 1 communicates with users at 3101 terminals in either node. The 3101 terminals are attached to the Series/1s by feature programmable communications adapter.

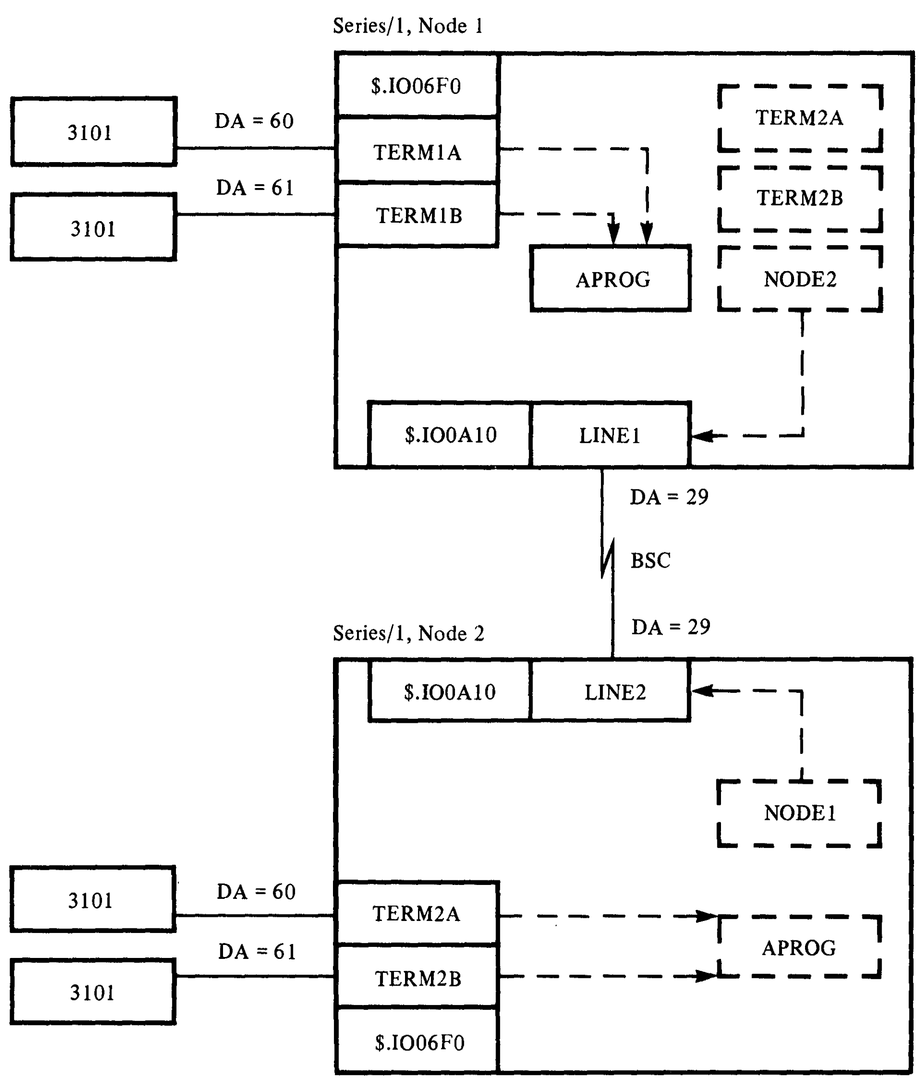


Figure 85. Example 6: BSC Point-to-Point Connection between Series/1s

Message Flow

The flow of messages between the terminals in node 1 and the application program is as described for example 4. The flow of messages between nodes 1 and 2 is similar to that described for example 4, but there are some important differences.

Look at the station definition worksheets for example 6. As for example 4, each node has a unique node assignment, and there are definitions of all remote stations to which local stations will send messages. The differences appear in the line/terminal/node configuration worksheets for nodes 1 and 2. In each node, the remote node is defined as a node station with no ring address or line name. (If you prefer, you can define the remote node as a CPU line station, with the same network address as the other end of the line. The effect of these two definitions is identical when the station is started.)

When these stations have been defined, linked as shown in the diagram of the configuration, and started, a display of the active stations in node 2 will show this information:

STATION	TYPE	NA	LINK
$.DISP	0000	0200	0000
NODE1	0800	0100	0229
LINE2	0A10	0229	0000
TERM2A	06F0	0260	01AC
TERM2B	06F0	0261	01AC
APROG	0800	01AC	0000

Notice the following:

- The network address of $.DISP (the system station) shows that the node assignment is 02. This is the result of issuing a CP SET NODE command.
- The remote program station (APROG) has a vector station block (type 08). This is the result of starting a station whose type is other than NODE and whose network address is not in this node.
- The remote node station (NODE1) has a vector station block (type 08). This is the result of starting a station whose type is NODE, but which doesn't have an associated line station.

A message from TERM2A is routed to the application program in node 1 in almost the same way as in example 4. The 3101 IOCP, $.IO06F0, sends the message to the origin station's link, APROG, whose network address is 01AC. The message dispatcher determines that the address is not in this node, sets the second byte to 00, and looks for a station with that network address (0100). Station NODE1 satisfies the search, so the message dispatcher puts the message on the queue of the station to which NODE1 is linked. That station is the one with network address 0229 (LINE2).

It's this last step that is different from example 4; the difference arises because of the type of the station that has network address 0100. In example 4, it was a node station (type 10), so the message dispatcher put the message on its queue. Here it's a vector station (type 08), so the message dispatcher put the message on its link's queue.

The method of routing messages between nodes accounts for the restriction that you can't have more than one active BSC point-to-point connection, Local

Communications Controller connection, or X.25 connection through circuit stations with usage CF between two Series/1s. For all these connections, a station with network address *nn*00 is required to route messages to node *nn*, and you can't start more than one station with a given network address.

Subsequently, the Series/1-to-Series/1 point-to-point IOCP, $.IO0A10, retrieves the message from LINE2's queue and sends it over the BSC line to node 1. $.IO0A10 in node 1 reads the message and sends it to the message dispatcher. The message carries its original routing information showing that the origin is 0260 and the destination is 01AC. The message dispatcher determines that the destination network address is in this node, so it puts the message on APROG's queue. The response from APROG to TERM2A is routed in a similar manner.

In summary, to enable communication between nodes over a point-to-point line, you must:

- Give each node a unique node assignment.
- In each node, define the remote stations to which local stations will send messages. Be sure that the network address is the same as that of the corresponding real station in the remote node. The station names need not be the same.
- In each node, define the local line station and the remote node station. Link the remote node station to the local line station.
- When you start the Communications Facility, issue a CP SET NODE command to set the node assignment before starting any remote stations.

Station Definitions

Figure 86 on page 325 through Figure 91 on page 330 show the station definitions for the two nodes in this configuration.

User/Message/Alias/Volume Configuration Worksheet

System: Example 6: Series/1-Series/1, Node 1
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
3101-FPCA IOCP	$.IO06F0	01AA	USER	
Series/1-Series/1 IOCP	$.IO0A10	01AB	USER	
Application program	APROG	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

3 User/Msg Stations	× 88 = 264 bytes
_____Alias Stations	× 40 = _____ bytes
_____Remote Stations	× 32 = _____ bytes
_____Volume Stations	× 104 = _____ bytes

Figure 86. Example 6: Node 1 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 6: Series/1-Series/1, Node 1
Node Assignment: 01

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM1A	0160	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
3101	TERM1B	0161	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
Remote 3101	TERM2A	0260	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
Remote 3101	TERM2B	0261	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

______3101F Devices	× 160 = ______ bytes
______4978/Printer Devices	× 264 = ______ bytes
2 3101 Devices	× 304 = 608 bytes
2 Remote Stations	× 32 = 64 bytes

Figure 87. Example 6: Node 1 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 6: Series/1-Series/1, Node 1
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC line	LINE1	0129	LINE	CPU	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 2	NODE2	0200	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = _____ bytes
1 Other Line	× 264 = 264 bytes
_____Terminals/Nodes	× 88 = _____ bytes
1 Remote Station	× 32 = 32 bytes

Figure 88. Example 6: Node 1 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 6: Series/1-Series/1 Node 2
Node Assignment: 02

Description	Name	NA	Type	Alias[1]
3101-FPCA IOCP	$.IO06F0	02AA	USER	
Series/1-Series/1 IOCP	$.IO0A10	02AB	USER	
Remote application program	APROG	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
_____Alias Stations	× 40 = ______ bytes
1 Remote Station	× 32 = 32 bytes
_____Volume Stations	× 104 = ______ bytes

Figure 89. Example 6: Node 2 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 6: Series/1-Series/1, Node 2
Node Assignment: 02

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM2A	0260	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
3101	TERM2B	0261	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

______3101F Devices	× 160 = ______ bytes	
______4978/Printer Devices	× 264 = ______ bytes	
2 3101 Devices	× 304 = 608 bytes	
______Remote Stations	× 32 = ______ bytes	

Figure 90. Example 6: Node 2 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 6: Series/1-Series/1, Node 2
Node Assignment: 02

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC line	LINE2	0229	LINE	CPU	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 1	NODE1	0100	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
_____Terminals/Nodes	× 88 = ______ bytes
1 Remote Station	× 32 = 32 bytes

Figure 91. Example 6: Node 2 Line/Terminal/Node Configuration Worksheet

EDX Definitions

Figure 92 on page 331 shows the EDX definitions for either node in this example. Notice that the 3101s are not defined to EDX, so they can be used only as Communications Facility terminals. If you also want to use them as EDX terminals, use TERMINAL statements to define them.

```
*** EXAMPLE 6,  NODE 1 OR 2
***
$EDXDEF   CSECT
          SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                   X
               INITMOD=(CSXINIT)
            •
            •
$SYSPRTR  TERMINAL DEVICE=4973,ADDRESS=01,END=YES
            •
            •
          BSCLINE  ADDRESS=29,TYPE=PT,END=YES   SERIES/1-SERIES/1
          END
```

Figure 92. Example 6: EDX Definitions

Example 7: BSC Multipoint Connection between Series/1s

Figure 93 on page 333 shows a configuration in which three Series/1s are connected by a BSC multipoint line. The application program in node 1 communicates with users in all three nodes.

The 3270 emulation IOCP, $.IO0AE0, manages the BSC line at nodes 2 and 3. Each of these Series/1s appears to be a 3270 system—a 3271 control unit with two 3277 display stations. The 3270 control IOCP, $.IO0AC0, manages the BSC line at node 1. It communicates with the two emulated 3270 systems just as if it were communicating with two real 3270 systems.

Message Flow

The routing of messages between nodes in this configuration is done according to the 3270 polling and selection addresses of the 3271 control and emulation terminal stations. Look at the line/terminal/node configuration worksheets for the three nodes. Notice that the polling and selection addresses of the control terminals in node 1 match the polling and selection addresses of the emulation terminals in nodes 2 and 3.

A message from one of the 4978s in node 2 is routed to the application program in node 1 as follows. The message is entered at station TERM2A and sent to its link, station ETERM2A. Subsequently, when $.IO0AE0 is polled by $.IO0AC0, it retrieves the message from ETERM2A's queue, inserts ETERM2A's polling address (C240), and writes the message to the BSC line. $.IO0AC0 receives the message in node 1 and disposes of it according to the polling address. Address C240 identifies station CTERM2A, so the message is sent to its link, station APROG.

When APROG receives the message, it no longer has its original routing information. The origin station is CTERM2A, not TERM2A. When APROG sends a response to the message, the response is routed back over the same path—from CTERM2A to ETERM2A and then to its link, TERM2A. A separate path between each remote device and the application program is required because the original routing information is lost. If the messages flowed in only one direction, from node 2 to node 1, all messages from node 2 could be routed through a single pair of emulation/control terminal stations. But if responses were routed back over the single path, there would be no way to determine which node 2 station should receive the message.

In this example, each node has been given a unique node assignment. This is not required in this case, because there are no remote stations (that is, there is no station whose network address identifies it as being in a remote node). You can issue CP SET NODE commands to set node assignments, but you don't have to. If you don't, the node assignment is 00, and the message dispatcher treats all stations as local stations, whatever their network address.

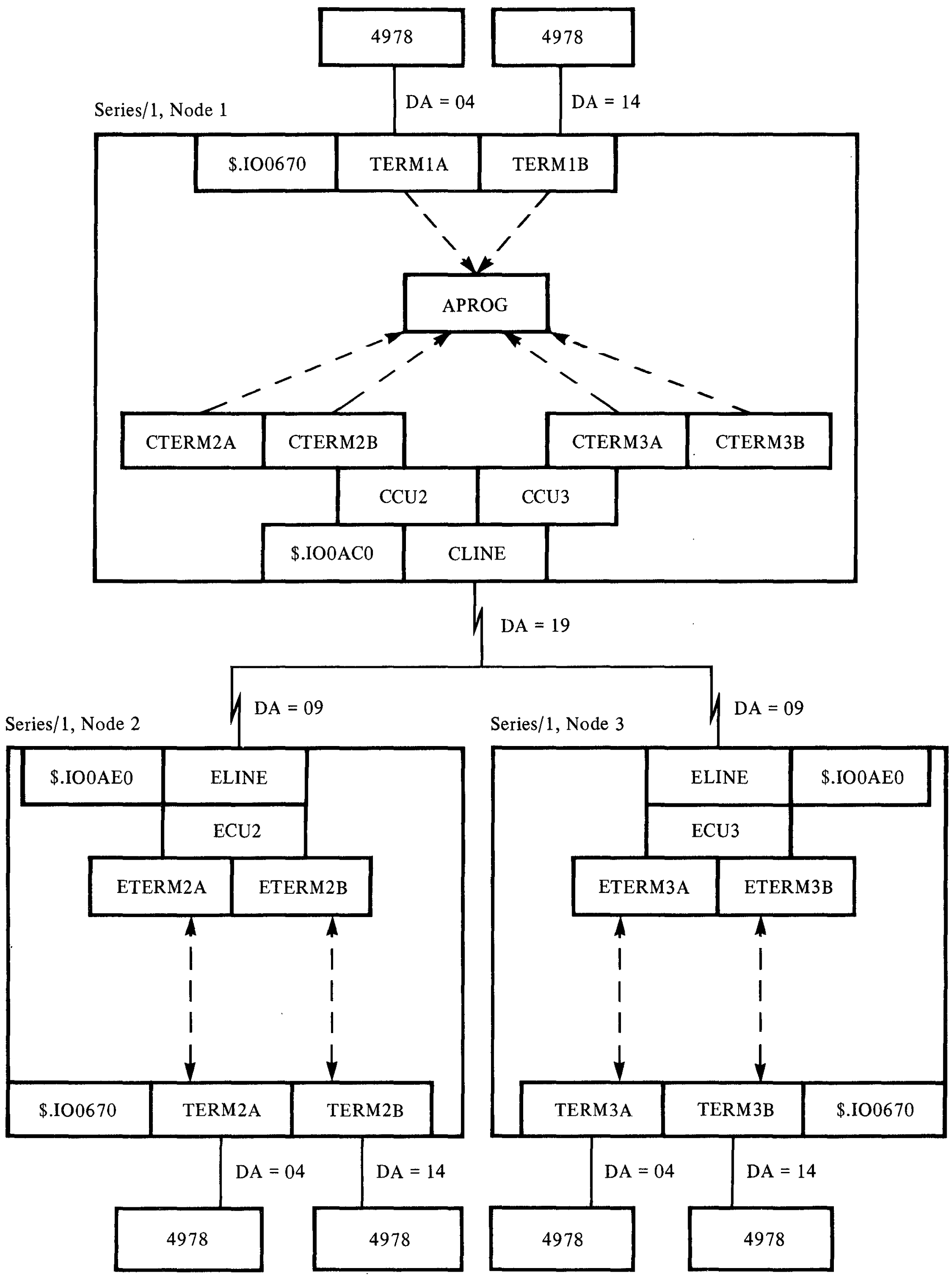


Figure 93. Example 7: BSC Multipoint Connection between Series/1s

In summary, what you must do to enable communication between nodes over a BSC multipoint line is:

- Designate one of the nodes as the control node and the other nodes as tributary (emulation) nodes.
- In each tributary node, define:
  - One emulated 3271 station.
  - One emulated 3277 station for each station that will send messages to a station in the control node.
- In the control node, define:
  - One control 3271 station for each tributary node, with the same polling address as the emulated 3271 station.
  - The same number of control 3277 stations as emulated 3277 stations, with matching polling and selection addresses.

Station Definitions

Figure 94 through Figure 102 on page 340 show the station definitions for this example.

User/Message/Alias/Volume Configuration Worksheet

System: Example 7: 3270 Emulation/Control, Node 1
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
4978 IOCP	$.IO0670	01AA	USER	
3270 control IOCP	$.IO0AC0	01AB	USER	
Application program	APROG	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

3 User/Msg Stations	× 88 = 264 bytes
______Alias Stations	× 40 = ______ bytes
______Remote Stations	× 32 = ______ bytes
______Volume Stations	× 104 = ______ bytes

Figure 94. Example 7: Node 1 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 7: 3270 Emulation/Control, Node 1
Node Assignment: 01

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
4978	TERM1A	0104	DEVICE	4978	2500		4040	0000		
4978	TERM1B	0114	DEVICE	4978	2500		4040	0000		

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

_____3101F Devices	× 160 = ______ bytes
2 4978/Printer Devices	× 264 = 528 bytes
_____3101 Devices	× 304 = ______ bytes
_____Remote Stations	× 32 = ______ bytes

Figure 95. Example 7: Node 1 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 7: 3270 Emulation/Control, Node 1
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC line	CLINE	0119	LINE	3271C	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Control unit	CCU2	01C1	TERM	3271		CLINE		C27F	0000
Terminal	CTERM2A	01C2	TERM	3277		CLINE		C240	E240
Terminal	CTERM2B	01C3	TERM	3277		CLINE		C2C1	E2C1
Control unit	CCU3	01C4	TERM	3271		CLINE		C37F	0000
Terminal	CTERM3A	01C5	TERM	3277		CLINE		C340	E340
Terminal	CTERM3B	01C6	TERM	3277		CLINE		C3C1	E3C1

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = _______ bytes
1 Other Line	× 264 = 264 bytes
6 Terminals/Nodes	× 88 = 528 bytes
_____Remote Stations	× 32 = _______ bytes

Figure 96. Example 7: Node 1 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 7: 3270 Emulation/Control, Node 2
Node Assignment: 02

Description	Name	NA	Type	Alias[1]
4978 IOCP	$.IO0670	02AA	USER	
3270 emulation IOCP	$.IO0AE0	02AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
______Alias Stations	× 40 = ______ bytes
______Remote Stations	× 32 = ______ bytes
______Volume Stations	× 104 = ______ bytes

Figure 97. Example 7: Node 2 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 7: 3270 Emulation/Control, Node 2
Node Assignment: 02

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
4978	TERM2A	0204	DEVICE	4978	2500		4040	0000		
4978	TERM2B	0214	DEVICE	4978	2500		4040	0000		

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

______3101F Devices	× 160 = ______ bytes
2 4978/Printer Devices	× 264 = 528 bytes
______3101 Devices	× 304 = ______ bytes
______Remote Stations	× 32 = ______ bytes

Figure 98. Example 7: Node 2 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 7: 3270 Emulation/Control, Node 2
Node Assignment: 02

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC tributary line	ELINE	0209	LINE	3271E	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Control unit	ECU2	02E1	TERM	3271		ELINE		C27F	0000
Terminal	ETERM2A	02E2	TERM	3277		ELINE		C240	E240
Terminal	ETERM2B	02E3	TERM	3277		ELINE		C2C1	E2C1

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = _____ bytes
1 Other Line	× 264 = 264 bytes
3 Terminals/Nodes	× 88 = 264 bytes
_____Remote Stations	× 32 = _____ bytes

Figure 99. Example 7: Node 2 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 7: 3270 Emulation/Control, Node 3
Node Assignment: 03

Description	Name	NA	Type	Alias[1]
4978 IOCP	$.IO0670	03AA	USER	
3270 emulation IOCP	$.IO0AE0	03AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
_____Alias Stations	× 40 = _____ bytes
_____Remote Stations	× 32 = _____ bytes
_____Volume Stations	× 104 = _____ bytes

Figure 100. Example 7: Node 3 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 7: 3270 Emulation/Control, Node 3
Node Assignment: 03

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
4978	TERM3A	0304	DEVICE	4978	2500		4040	0000		
4978	TERM3B	0314	DEVICE	4978	2500		4040	0000		

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

_____3101F Devices	× 160 = _____ bytes
2 4978/Printer Devices	× 264 = 528 bytes
_____3101 Devices	× 304 = _____ bytes
_____Remote Stations	× 32 = _____ bytes

Figure 101. Example 7: Node 3 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 7: 3270 Emulation/Control, Node 3
Node Assignment: 03

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC tributary line	ELINE	0309	LINE	3271E	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/Port#[4]	Select CUDA[3]
Control unit	ECU3	03E1	TERM	3271		ELINE		C37F	0000
Terminal	ETERM3A	03E2	TERM	3277		ELINE		C340	E340
Terminal	ETERM3B	03E3	TERM	3277		ELINE		C3C1	E3C1

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
3 Terminals/Nodes	× 88 = 264 bytes
_____Remote Stations	× 32 = ______ bytes

Figure 102. Example 7: Node 3 Line/Terminal/Node Configuration Worksheet

EDX Definitions

Figure 103 shows the EDX definitions for the control node.

```
*** EXAMPLE 7, NODE 1
***
$EDXDEF   CSECT
          SYSTEM    MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                     X
               INITMOD=(CSXINIT)
            •
            •
TERM1     TERMINAL DEVICE=4978,ADDRESS=04,HDCOPY=$SYSPRTR
TERM2     TERMINAL DEVICE=4978,ADDRESS=14,HDCOPY=$SYSPRTR
$SYSPRTR  TERMINAL DEVICE=4973,ADDRESS=01,END=YES
            •
            •
          BSCLINE   ADDRESS=19,TYPE=MC,END=YES    3271 CONTROL
          END
```

Figure 103. Example 7: Node 1 EDX Definitions

Figure 104 shows the EDX definitions for either of the tributary nodes.

```
*** EXAMPLE 7, NODE 2 OR 3
***
$EDXDEF   CSECT
          SYSTEM    MAXPROG=(10,10),PARTS=(32,32),                      X
               COMMON=(CSXEND,CSXEND),                                   X
               INITMOD=(CSXINIT)
            •
            •
TERM1     TERMINAL DEVICE=4978,ADDRESS=04,HDCOPY=$SYSPRTR
TERM2     TERMINAL DEVICE=4978,ADDRESS=14,HDCOPY=$SYSPRTR
$SYSPRTR  TERMINAL DEVICE=4973,ADDRESS=01,END=YES
            •
            •
          BSCLINE   ADDRESS=09,TYPE=MT,END=YES    3271 EMULATION
          END
```

Figure 104. Example 7: Node 2 or 3 EDX Definitions

Example 8: BSC Multipoint Connection with Transaction Processing

Figure 105 on page 343 shows a configuration in which three Series/1s and a host computer are connected by BSC multipoint lines. This example includes the program dispatcher with support for remote disk, remote IPL, and remote management functions. Node 1 is considered the central site. It has the disks and the definitions that allow it to do remote management of nodes 2 and 3 and IPL them. Nodes 2 and 3 have definitions that allow them to access volumes EDX003 and USRVOL on node 1's disks.

The example doesn't include Communications Facility terminals; you could, of course, add them to such a configuration.

Message Flow

The routing of messages between node 1 and the host is as explained in example 1. The routing of messages between nodes is as explained in example 7. Be sure you understand the requirements, given in example 7, for communication between nodes over a BSC multipoint line.

Look at Figure 106 on page 344, the program dispatcher data sets. The path stations are the control and emulation terminal stations by which messages are routed from one node to another. A link from each path station to the program dispatcher is required, but you don't have to define them. The program dispatcher sets the links when it's started.

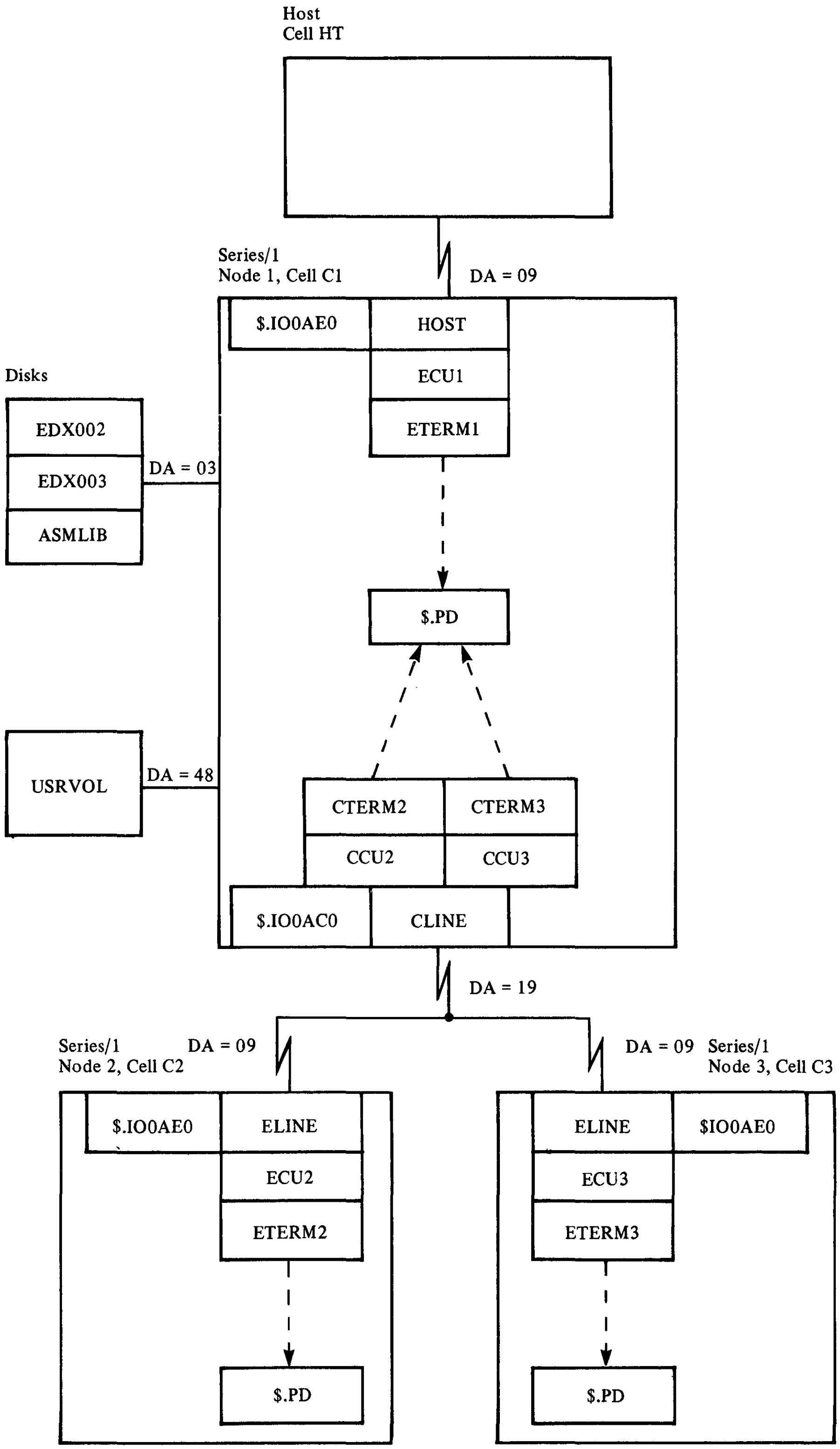


Figure 105. Example 8: BSC Multipoint Connection with Transaction Processing

```
*** EXAMPLE 8, CELL C1
SET LOG $SYSPRTR
S HOST ECU1
S CLINE CCU2 CCU3
/*
CELL C1
PATH ETERM1,HT,C P
PATH CTERM2,C2
PATH CTERM3,C3
/*
TID <IO> $.PD<IO>,,,N 32
TID IPL $.PDIPL 22
TID HMU $.HMU 22
/*
SYST**       MS********CELL C1 IS OPERATIONAL
/*

*** EXAMPLE 8, CELL C2
SET LOG $SYSPRTR
S ELINE ECU2
/*
CELL C2
PATH ETERM2,C1 P
/*
TID RMU $.RMU 22
/*
SYST         SPE1C1
SYST**       MS********CELL C2 IS OPERATIONAL
SYSTC1       CP PD T C2
/*

*** EXAMPLE 8, CELL C3
SET LOG $SYSPRTR
S ELINE ECU3
/*
CELL C3
PATH ETERM3,C1 P
/*
TID RMU $.RMU 22
/*
SYST         SPE1C1
SYST**       MS********CELL C3 IS OPERATIONAL
SYSTC1       CP PD T C3
/*
```

Figure 106. Example 8: Program Dispatcher Data Sets

Let's see what happens to a transaction with primary cell identifier C1 that originates in cell C2:

1. The program dispatcher in cell C2 receives the transaction, determines that the path to cell C1 is station ETERM2, and sends the transaction to that station.

2. Subsequently, $.IO0AE0 retrieves the transaction from ETERM2's queue, inserts ETERM2's polling address (C240), and sends the transaction over the BSC line.

3. $.IO0AC0 in cell C1 reads the data and disposes of it according to the polling address. Address C240 identifies station CTERM2, so the transaction is sent to its link, $.PD.

4. The program dispatcher retrieves the transaction from its queue, determines that it is to be processed in this cell, and routes it according to the primary transaction identifier.

Program Dispatcher Data Sets

Figure 106 on page 344 shows the program dispatcher data set ($.SYSPD) for each of the three cells. The commands in the first part of each data set assign $SYSPRTR as the Communications Facility log and start the line stations and the 3271 terminal stations. You don't have to start the path stations; the program dispatcher does that when it's started. You can add commands to start other stations in your configuration, or you can start them independently of the program dispatcher.

The second part of each data set defines the local cell and the paths to remote cells. In cell C1, the path to the host system is the preferred path. The definition specifies data conversion, so binary data can be transmitted to and from the host. In cells C2 and C3, only the path to C1 is defined, but it provides a path to all cells because it is the preferred path. You could define paths to each of the other cells (they could all use the same path station), but the effect would be the same.

The third part of cell C1's data set defines the transactions that support remote disk, remote IPL, and the host management utility program. The third part of cell C2's and C3's data set defines the transaction for the remote management utility program.

The SYST MS broadcast transaction in the fourth part of cell C1's data set notifies all cells that C1 is operational. The message goes to all EDX terminals in all cells. The transaction also goes to the host system. If this causes a problem with the host transaction-processing program, route a SYST MS transaction to each of the Series/1 cells instead of using a broadcast transaction.

The SYST SP transaction in C2's and C3's data set causes remote disk support (CSPDIO) in those cells to release control of the BSC line so that $.IOOAE0 can use it. The SYST CP transactions cause cell C1 to send its system time and date to cells C2 and C3.

IPL Transaction Data Set

An IPL transaction data set ($.SYSLCC) is required in cell C1, because there is a definition of transaction IPL. The data set won't be used (it's used only for Local Communications Controller hardware IPL requests), so it can be a small empty data set.

Station Definitions

Figure 107 on page 346 through Figure 112 on page 351 show the station definitions for the three nodes in this configuration.

User/Message/Alias/Volume Configuration Worksheet

System: Example 8: BSC Multipoint with Transaction Processing, Node 1
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
3270 emulation IOCP	$.IO0AE0	01AA	USER	
3270 control IOCP	$.IO0AC0	01AB	USER	
Program dispatcher	$.PD	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

3 User/Msg Stations	× 88 = 264 bytes
______Alias Stations	× 40 = ______ bytes
______Remote Stations	× 32 = ______ bytes
______Volume Stations	× 104 = ______ bytes

Figure 107. Example 8: Node 1 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 8: BSC Multipoint with Transaction Processing, Node 1
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC tributary line	HOST	0109	LINE	3271E	2500			
BSC control line	CLINE	0119	LINE	3271C	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Control unit	ECU1	01E1	TERM	3271		HOST		407F	0000
Terminal	ETERM1	01E2	TERM	3277		HOST		4040	6040
Control unit	CCU2	01C1	TERM	3271		CLINE		C27F	0000
Terminal	CTERM2	01C2	TERM	3277		CLINE		C240	E240
Control unit	CCU3	01C3	TERM	3271		CLINE		C37F	0000
Terminal	CTERM3	01C4	TERM	3277		CLINE		C340	E340

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = _______ bytes
2 Other Lines	× 264 = 528 bytes
6 Terminals/Nodes	× 88 = 528 bytes
_____Remote Stations	× 32 = _______ bytes

Figure 108. Example 8: Node 1 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 8: BSC Multipoint with Transaction Processing, Node 2
Node Assignment: 02

Description	Name	NA	Type	Alias[1]
3270 emulation IOCP	$.IO0AE0	02AA	USER	
Program dispatcher	$.PD	02AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
_____Alias Stations	× 40 = ______ bytes
_____Remote Stations	× 32 = ______ bytes
_____Volume Stations	× 104 = ______ bytes

Figure 109. Example 8: Node 2 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 8: BSC Multipoint with Transaction Processing, Node 2
Node Assignment: 02

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC tributary line	ELINE	0209	LINE	3271E	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Control unit	ECU2	02E1	TERM	3271		ELINE		C27F	0000
Terminal	ETERM2	02E2	TERM	3277		ELINE		C240	E240

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = _____ bytes
1 Other Line	× 264 = 264 bytes
2 Terminals/Nodes	× 88 = 176 bytes
_____Remote Stations	× 32 = _____ bytes

Figure 110. Example 8: Node 2 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 8: BSC Multipoint with Transaction Processing, Node 3
Node Assignment: 03

Description	Name	NA	Type	Alias[1]
3270 emulation IOCP	$.IO0AE0	03AA	USER	
Program dispatcher	$.PD	03AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations × 88 = 176 bytes
_____Alias Stations × 40 = ______ bytes
_____Remote Stations × 32 = ______ bytes
_____Volume Stations × 104 = ______ bytes

Figure 111. Example 8: Node 3 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 8: BSC Multipoint with Transaction Processing, Node 3
Node Assignment: 03

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC tributary line	ELINE	0309	LINE	3271E	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Control unit	ECU3	03E1	TERM	3271		ELINE		C37F	0000
Terminal	ETERM3	03E2	TERM	3277		ELINE		C340	E340

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = _____ bytes
1 Other Line	× 264 = 264 bytes
2 Terminals/Nodes	× 88 = 176 bytes
_____Remote Stations	× 32 = _____ bytes

Figure 112. Example 8: Node 3 Line/Terminal/Node Configuration Worksheet

EDX Definitions

Figure 113 on page 352 shows the EDX definitions for node 1.

```
*** EXAMPLE 8, NODE 1
***
$EDXDEF    CSECT
           SYSTEM    MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
                COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                    X
                INITMOD=(CSXINIT)
             •
             •
           DISK      DEVICE=4962-3,ADDRESS=03,                           X
                VOLNAME=(EDX002,EDX003,ASMLIB),TASK=YES
           DISK      DEVICE=4963-29,ADDRESS=48,                          X
                VOLNAME=(USRVOL),TASK=YES,END=YES
             •
             •
           BSCLINE   ADDRESS=09,TYPE=MT,END=YES    3271 EMULATION
           BSCLINE   ADDRESS=19,TYPE=MC,END=YES    3271 CONTROL
           END
```

Figure 113. Example 8: Node 1 EDX Definitions

Figure 114 shows the EDX definitions for either node 2 or node 3.

```
*** EXAMPLE 8,  NODE 2 OR 3
***
$EDXDEF    CSECT
           SYSTEM    MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
                COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                    X
                INITMOD=(CSXINIT)
             •
             •
           DISK      DEVICE=4962-3,ADDRESS=E1,                           X
                VOLNAME=(EDX003),TASK=NO
           DISK      DEVICE=4963-29,ADDRESS=E2,                          X
                VOLNAME=(REMVOL),TASK=NO,END=YES
             •
             •
CDRVTA     TERMINAL DEVICE=VIRT,ADDRESS=CDRVTB,LINSIZE=254,SYNC=YES
CDRVTB     TERMINAL DEVICE=VIRT,ADDRESS=CDRVTA,LINSIZE=254,END=YES
             •
             •
           BSCLINE   ADDRESS=09,TYPE=MT,END=YES    3271 EMULATION
           DEFINE    DEVICE,CUDA=X'C240',LINE=9,CELL=C2,ADDRESS=(E1,E2)15
           DEFINE    VOLUME,CELL=C1,VOLNAME=(EDX003,EDX003)
           DEFINE    VOLUME,CELL=C1,VOLNAME=(REMVOL,USRVOL),END=YES
           END
```

Figure 114. Example 8: Node 2 or 3 EDX Definitions

Notice the definitions for remote disks and for the virtual terminals required by the remote management utility program.

[15] CELL=C3 for node 3 and CUDA=X'C340' for cell C3.

Example 9: Local Communications Controller Connection with Transaction Processing

Figure 115 on page 355 shows a configuration in which three Series/1s are connected to a Local Communications Controller. This example includes the program dispatcher with support for remote disks, remote IPL, and remote management functions. Node 1 is considered the central site. It has the disks and the definitions that allow it to do remote management of nodes 2 and 3 and IPL them. Nodes 2 and 3 have definitions that allow them to access volumes EDX002 and EDX003 on node 1's disk.

The example doesn't include Communications Facility terminals or communication with a host system; you could, of course, add them to such a configuration.

Message Flow

The routing of messages between nodes on a Local Communications Controller is as explained in example 4. Be sure you understand that example before you study this one.

Look at Figure 116 on page 356, the program dispatcher data sets. The path stations are those that represent the remote program dispatchers. The path and station definitions allow transactions to be routed directly between any two cells. Although cell C1 is considered the central site, transactions can be routed between cells C2 and C3 without going through C1.

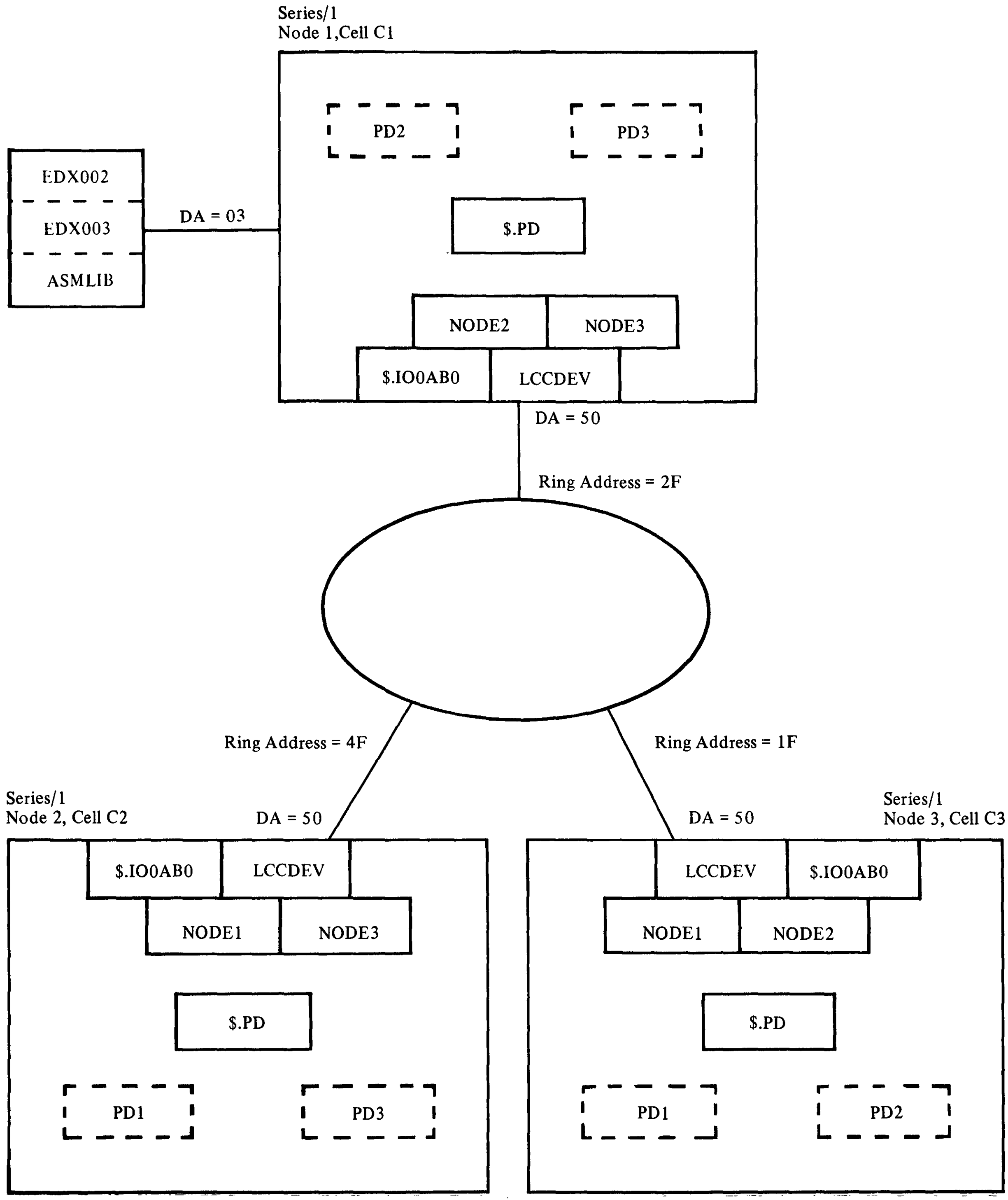


Figure 115. Example 9: Local Communications Controller Connection with Transaction Processing

```
*** EXAMPLE 9, CELL C1
SET LOG $SYSPRTR
SET NODE 0100
S LCCDEV NODE2 NODE3
/*
CELL C1
PATH PD2,C2
PATH PD3,C3
/*
TID <IO> $.PD<IO>,,,N 32
TID IPL $.PDIPL 22
TID HMU $.HMU 22
/*
SYST**       MS********CELL C1 IS OPERATIONAL
/*

*** EXAMPLE 9, CELL C2
SET LOG $SYSPRTR
SET NODE 0200
S LCCDEV NODE1 NODE3
/*
CELL C2
PATH PD1,C1 P
PATH PD3,C3
/*
TID RMU $.RMU 22
/*
SYST         SPE1C1
SYST**       MS********CELL C2 IS OPERATIONAL
SYSTC1       CP PD T C2
/*

*** EXAMPLE 9, CELL C3
SET LOG $SYSPRTR
SET NODE 0300
S LCCDEV NODE1 NODE2
/*
CELL C3
PATH PD1,C1 P
PATH PD2,C2
/*
TID RMU $.RMU 22
/*
SYST         SPE1C1
SYST**       MS********CELL C3 IS OPERATIONAL
SYSTC1       CP PD T C3
/*
```

Figure 116. Example 9: Program Dispatcher Data Sets

A transaction with primary cell identifier C3 that originates in cell C2 is routed as follows:

1. The program dispatcher in cell C2 receives the transaction, determines that the path to cell C3 is station PD3, and sends the transaction to that station.

2. PD3 is a remote station (its network address, 03AB, is not in this node), so node 2's message dispatcher puts the transaction on the queue of station NODE3—the station whose network address is 0300.

3. Subsequently, $.IO0AB0 retrieves the message from NODE3's queue and sends it over the Local Communications Controller ring.

4. Station NODE3 has ring address 1F, so $.IO0AB0 in node 3 receives the data. The IOCP sends the message with its header to the message dispatcher.

5. The header shows that the destination network address (03AB) is in the local node, so the message dispatcher puts the message on the queue of the station with that network address, $.PD.

6. The program dispatcher retrieves the transaction from its queue, determines that it is to be processed in this cell, and routes it according to the primary transaction identifier.

Program Dispatcher Data Sets

Figure 116 on page 356 shows the program dispatcher data set ($.SYSPD) for each of the three cells. The commands in the first part of each data set assign $SYSPRTR as the Communications Facility log, set the node assignment, and start the line and node stations. You don't have to start the path stations; the program dispatcher does that when it's started. If you put the SET NODE command in $.SYSPD as shown here, be sure that you start the program dispatcher before starting any remote stations in your configuration.

The second part of each data set defines the local cell and the paths to each of the remote cells. The remainder of each data set is the same as in example 8. See that example for an explanation of the TID statements and transactions.

IPL Transaction Data Set

An IPL transaction data set ($.SYSLCC) is required in cell C1, because that contains the definition of transaction IPL. The following IPL transaction data set enables cell C1 to respond to hardware IPL requests from the other two Series/1s:

```
NODE2   IPL C1      C2B9EDX002$EDXNUC2,EDX002
NODE3   IPL C1      C3B9EDX002$EDXNUC3,EDX002
```

The program dispatcher in cell C1 responds to a hardware IPL request by sending a bootstrap and then a nucleus to the remote Series/1. If you don't intend to use hardware IPL requests, $.SYSLCC can be a small empty data set.

Station Definitions

Figure 117 on page 358 through Figure 122 on page 363 show the station definitions for this example.

User/Message/Alias/Volume Configuration Worksheet

System: Example 9: Local Communications Controller with Transactions, Node 1
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
Local Communications Controller IOCP	$.IO0AB0	01AA	USER	
Program dispatcher	$.PD	01AB	USER	
Program dispatcher in node 2	PD2	02AB	USER	
Program dispatcher in node 3	PD3	03AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations × 88 = 176 bytes
_____Alias Stations × 40 = ______ bytes
2 Remote Stations × 32 = 64 bytes
_____Volume Stations × 104 = ______ bytes

Figure 117. Example 9: Node 1 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 9: Local Communications Controller with Transactions, Node 1
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller	LCCDEV	01C0	LINE	LCC	2500	50		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 2	NODE2	0200	NODE	CF	4F	LCCDEV			
Node 3	NODE3	0300	NODE	CF	1F	LCCDEV			

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

____HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
2 Terminals/Nodes	× 88 = 176 bytes
____Remote Stations	× 32 = ______ bytes

Figure 118. Example 9: Node 1 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 9: Local Communications Controller with Transactions, Node 2
Node Assignment: 02

Description	Name	NA	Type	Alias[1]
Local Communications Controller IOCP	$.IO0AB0	02AA	USER	
Program dispatcher	$.PD	02AB	USER	
Program dispatcher in node 1	PD1	01AB	USER	
Program dispatcher in node 3	PD3	03AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
_____Alias Stations	× 40 = _____ bytes
2 Remote Stations	× 32 = 64 bytes
_____Volume Stations	× 104 = _____ bytes

Figure 119. Example 9: Node 2 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 9: Local Communications Controller with Transactions, Node 2
Node Assignment: 02

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller	LCCDEV	02C0	LINE	LCC	2500	50		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 1	NODE1	0100	NODE	CF	2F	LCCDEV			
Node 3	NODE3	0300	NODE	CF	1F	LCCDEV			

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

______HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
2 Terminals/Nodes	× 88 = 176 bytes
______Remote Stations	× 32 = ______ bytes

Figure 120. Example 9: Node 2 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 9: Local Communications Controller with Transactions, Node 3
Node Assignment: 03

Description	Name	NA	Type	Alias[1]
Local Communications Controller IOCP	$.IO0AB0	03AA	USER	
Program dispatcher	$.PD	03AB	USER	
Program dispatcher in node 1	PD1	01AB	USER	
Program dispatcher in node 2	PD2	02AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
_____Alias Stations	× 40 = ______ bytes
2 Remote Stations	× 32 = 64 bytes
_____Volume Stations	× 104 = ______ bytes

Figure 121. Example 9: Node 3 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 9: Local Communications Controller with Transactions, Node 3
Node Assignment: 03

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller	LCCDEV	03C0	LINE	LCC	2500	50		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 1	NODE1	0100	NODE	CF	2F	LCCDEV			
Node 2	NODE2	0200	NODE	CF	4F	LCCDEV			

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

______HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
2 Terminals/Nodes	× 88 = 176 bytes
______Remote Stations	× 32 = ______ bytes

Figure 122. Example 9: Node 3 Line/Terminal/Node Configuration Worksheet

EDX Definitions

Figure 123 on page 364 shows the EDX definitions for node 1.

```
*** EXAMPLE 9, NODE 1
***
$EDXDEF   CSECT
          SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                  X
               INITMOD=(CSXINIT)
            •
            •
          DISK     DEVICE=4962-3,ADDRESS=03,                          X
               VOLNAME=(EDX002,EDX003,ASMLIB),TASK=YES,END=YES
            •
            •
          LCC      ADDRESS=50                    SUBCHANNEL 0
          LCC      ADDRESS=51                    SUBCHANNEL 1
          LCC      ADDRESS=52,END=YES            SUBCHANNEL 2
          END
```

Figure 123. Example 9: Node 1 EDX Definitions

Figure 124 shows the EDX definitions for either node 2 or node 3. It includes the definitions for remote disks and for the virtual terminals required by the remote management utility program.

```
*** EXAMPLE 9, NODE 2 OR 3
***
$EDXDEF   CSECT
          SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                  X
               INITMOD=(CSXINIT)
            •
            •
          DISK     DEVICE=4962-3,ADDRESS=E1,                          X
               VOLNAME=(EDX002,EDX003),TASK=NO,END=YES
            •
            •
CDRVTA    TERMINAL DEVICE=VIRT,ADDRESS=CDRVTB,LINSIZE=254,SYNC=YES
CDRVTB    TERMINAL DEVICE=VIRT,ADDRESS=CDRVTA,LINSIZE=254,END=YES
            •
            •
          LCC      ADDRESS=50                    SUBCHANNEL 0
          LCC      ADDRESS=51                    SUBCHANNEL 1
          LCC      ADDRESS=52,END=YES            SUBCHANNEL 2
          DEFINE   DEVICE,RINGADR=2F,LINE=50,CELL=C2[16],ADDRESS=E1
          DEFINE   VOLUME,CELL=C1,VOLNAME=(EDX002,EDX002)
          DEFINE   VOLUME,CELL=C1,VOLNAME=(EDX003,EDX003),END=YES
          END
```

Figure 124. Example 9: Node 2 or 3 EDX Definitions

[16] CELL=C3 for node 3.

Example 10: BSC Point-to-Point Connection with Transaction Processing

Figure 125 shows a configuration in which two Series/1s are connected by a BSC point-to-point line. This example includes the program dispatcher with support for remote disks, remote IPL, and remote management functions. Node 1 can do remote management of node 2 and IPL it. Node 2 can access volumes EDX002 and EDX003 on node 1's disk.

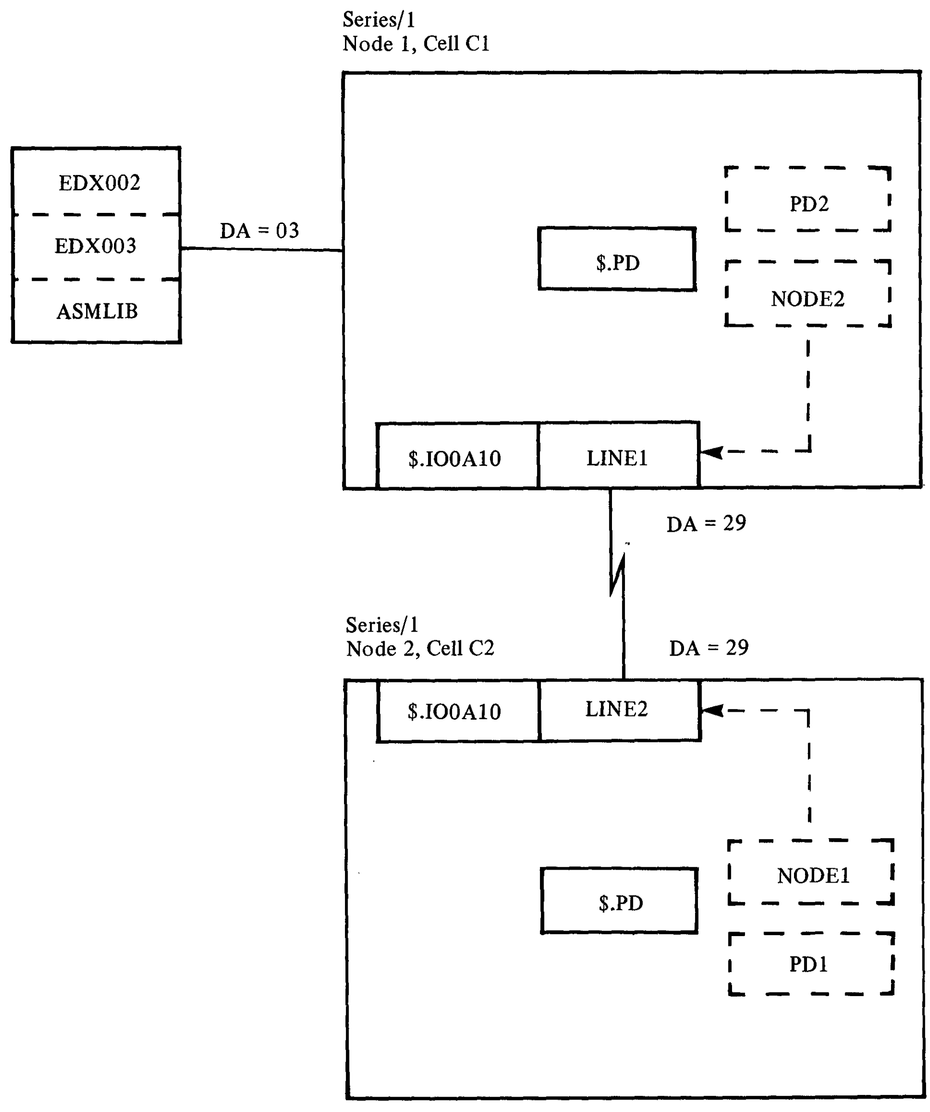


Figure 125. Example 10: BSC Point-to-Point Connection with Transaction Processing

The example doesn't include Communications Facility terminals or communication with a host system; you could, of course, add them to such a configuration.

Message Flow

The routing of messages between nodes connected by a BSC point-to-point line is explained in example 6. Be sure you understand that example before you study this one.

Look at Figure 126, the program dispatcher data sets. The path stations are those that represent the remote program dispatchers.

```
*** EXAMPLE 10, CELL C1
SET LOG $SYSPRTR
SET NODE 0100
S LINE1 LINE2
/*
CELL C1
PATH PD2,C2
/*
TID <IO> $.PD<IO>,,,N 32
TID IPL $.PDIPL 22
TID HMU $.HMU 22
/*
SYST**      MS********CELL C1 IS OPERATIONAL
/*

*** EXAMPLE 10, CELL C2
SET LOG $SYSPRTR
SET NODE 0200
S LINE1 LINE2
/*
CELL C2
PATH PD1,C1 P
/*
TID RMU $.RMU 22
/*
SYST        SPE1C1
SYST**      MS********CELL C2 IS OPERATIONAL
SYSTC1      CP PD T C2
/*
```

Figure 126. Example 10: Program Dispatcher Data Sets

A transaction with primary cell identifier C2 that originates in cell C1 is routed as follows:

1. The program dispatcher in cell C1 receives the transaction, determines that the path to cell C2 is station PD2, and sends the transaction to that station.

2. PD2 is a remote station (its network address, 02AB, is not in this node), so node 1's message dispatcher finds the station whose network address is 0200, NODE2. Station NODE2 is a vector station, so the message dispatcher puts the message on the queue of its link, LINE1.

3. Subsequently, $.IO0A10 retrieves the message from LINE1's queue and sends it over the BSC line.

4. $.IO0A10 in node 2 reads the data and sends it to the message dispatcher.

5. The message header shows that the destination network address (02AB) is in the local node, so the message dispatcher puts the message on the queue of the station with that network address, $.PD.

6. The program dispatcher retrieves the transaction from its queue, determines that it is to be processed in this cell, and routes it according to the primary transaction identifier.

Program Dispatcher Data Sets

Figure 126 on page 367 shows the program dispatcher data set ($.SYSPD) for each of the two cells. The commands in the first part of each data set assign $SYSPRTR as the Communications Facility log, set the node assignment, and start the local and remote line stations. You don't have to start the path stations; the program dispatcher does that when it's started. If you put the SET NODE command in $.SYSPD as shown here, be sure that you start the program dispatcher before starting any remote stations in your configuration.

The second part of each data set defines the local cell and the path to the remote cells. The remainder of each data set is the same as in example 8. See that example for an explanation of the TID statements and transactions.

IPL Transaction Data Set

An IPL transaction data set ($.SYSLCC) is required in cell C1, because that contains the definition of transaction IPL. The data set won't be used (it's used only for Local Communications Controller hardware IPL requests), so it can be a small empty data set.

Station Definitions

Figure 127 on page 369 through Figure 130 on page 372 show the station definitions for this example.

User/Message/Alias/Volume Configuration Worksheet

System: Example 10: BSC Point-to-Point with Transactions, Node 1
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
Series/1-Series/1 IOCP	$.IO0A10	01AA	USER	
Program dispatcher	$.PD	01AB	USER	
Program dispatcher in node 2	PD2	02AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
_____Alias Stations	× 40 = ______ bytes
1 Remote Station	× 32 = 32 bytes
_____Volume Stations	× 104 = ______ bytes

Figure 127. Example 10: Node 1 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 10: BSC Point-to-Point with Transactions, Node 1
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC line	LINE1	0129	LINE	CPU	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 2	NODE2	0200	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
_____Terminals/Nodes	× 88 = ______ bytes
1 Remote Station	× 32 = 32 bytes

Figure 128. Example 10: Node 1 Line/Terminal/Node Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 10: BSC Point-to-Point with Transactions, Node 2
Node Assignment: 02

Description	Name	NA	Type	Alias[1]
Series/1-Series/1 IOCP	$.IO0A10	02AA	USER	
Program dispatcher	$.PD	02AB	USER	
Program dispatcher in node 1	PD1	01AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
_____Alias Stations	× 40 = ______ bytes
1 Remote Station	× 32 = 32 bytes
_____Volume Stations	× 104 = ______ bytes

Figure 129. Example 10: Node 2 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 10: BSC Point-to-Point with Transactions, Node 2
Node Assignment: 02

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC line	LINE2	0229	LINE	CPU	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 1	NODE1	0100	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
_____Terminals/Nodes	× 88 = ______ bytes
1 Remote Station	× 32 = 32 bytes

Figure 130. Example 10: Node 2 Line/Terminal/Node Configuration Worksheet

EDX Definitions

Figure 131 on page 373 shows the EDX definitions for node 1.

```
*** EXAMPLE 10 NODE 1
***
$EDXDEF   CSECT
          SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                   X
               INITMOD=(CSXINIT)
            •
            •
          DISK     DEVICE=4962-3,ADDRESS=03,                           X
               VOLNAME=(EDX002,EDX003,ASMLIB),TASK=YES,END=YES
            •
            •
          BSCLINE  ADDRESS=29,TYPE=PT,END=YES   SERIES/1-SERIES/1
          END
```

Figure 131. Example 10: Node 1 EDX Definitions

Figure 132 shows the EDX definitions for node 2. It includes the definitions for remote disks and for the virtual terminals required by the remote management utility program.

```
*** EXAMPLE 10, NODE 2
***
$EDXDEF   CSECT
          SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                   X
               INITMOD=(CSXINIT)
            •
            •
          DISK     DEVICE=4962-3,ADDRESS=E1,                           X
               VOLNAME=(EDX002,EDX003),TASK=NO,END=YES
            •
            •
CDRVTA    TERMINAL DEVICE=VIRT,ADDRESS=CDRVTB,LINSIZE=254,SYNC=YES
CDRVTB    TERMINAL DEVICE=VIRT,ADDRESS=CDRVTA,LINSIZE=254,END=YES
            •
            •
          BSCLINE  ADDRESS=29,TYPE=PT,END=YES   SERIES/1-SERIES/1
          DEFINE   DEVICE,CUDA=0,LINE=29,CELL=C2,ADDRESS=E1
          DEFINE   VOLUME,CELL=C1,VOLNAME=(EDX002,EDX002)
          DEFINE   VOLUME,CELL=C1,VOLNAME=(EDX003,EDX003),END=YES
          END
```

Figure 132. Example 10: Node 2 EDX Definitions

Example 11: Multiple Local Communications Controller and BSC Connections

Figure 133 on page 375 shows a configuration in which six Series/1s are connected by two Local Communications Controller rings and a BSC point-to-point line. One of the Series/1s is a Communications Manager node; the other five are Communications Facility nodes. The stations shown in the diagram enable communication between any two nodes.

Message Flow

Before you study this example, be sure you understand how messages are routed between nodes connected by a single Local Communications Controller ring (example 5) and between nodes connected by a BSC point-to-point line (example 6). This example presents only one new concept—how messages are routed to a node that is accessible from, but not connected to, the origin node. Although the example only shows BSC and Local Communications Controller connections, such a configuration could also include HDLC connections. As explained in "Example 12: X.25 Connection between Series/1s" and "Example 13: X.25 Connection Between Series/1s through an X.25 Network," the routing of messages between nodes connected by or accessed from an HDLC line is essentially the same as when nodes are connected by a BSC point-to-point line. Only the definitions of the stations that represent the physical connection and the actions of the IOCP that manages the connection are different.

Consider the configuration from the viewpoint of node A. Nodes B and C are connected to the same ring as node A. Nodes D and E are on a different ring that can be accessed through node C. Node F is not connected to any ring, but it can be accessed through nodes C and D.

Look at Figure 134 on page 376, the line and node station definitions for node A. Notice that nodes D, E, and F are defined as node stations with no ring address or line name. Vector (not node) station blocks are created when these stations are started. A message sent to a station in node D is put on the queue of the station that DNODE is linked to, as explained in example 6. When you define nodes accessible from, but not on, the same ring, you must link them to the node station through which they can be accessed.

This example includes a Communications Manager node, so the node assignments must be the hexadecimal representation of an alphabetic character; node A is C1, node B is C2, and so on. To the Communications Manager, the nodes would be known by the names AA, BB, and so on. Any stations that will send messages to or receive messages from the Communications Manager must have names no longer than 4 characters. That rule doesn't apply to the stations shown in this example, because they aren't the origin or destination of messages; they are just used to route messages from one node to another.

Let's see how messages are routed over the longest path, from node A to node F. Assume that a message originates from some station in node A and has a destination address of some station in node F, say C699. The message is routed as follows:

1. In node A, the message dispatcher puts the message on the queue of the station to which the vector station with network address C600 is linked, CNODE.

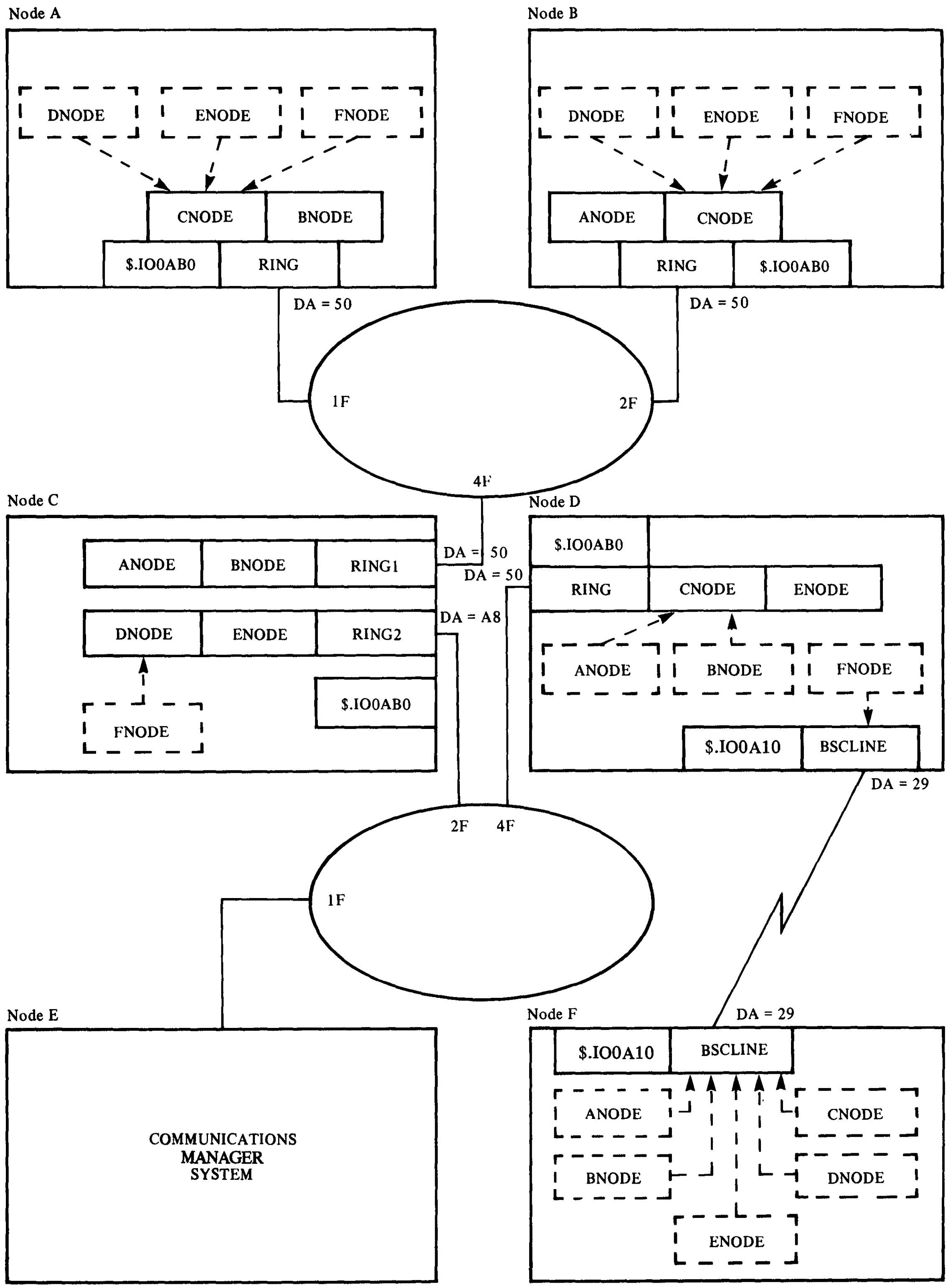


Figure 133. Example 11: Multiple Local Communications Controller and BSC Connections

$.IO0AB0 retrieves the message and sends it over the Local Communications Controller ring.

Line / Terminal / Node Configuration Worksheet

System: Example 11: Multiple Rings and BSC Connections, Node A , Node 3
Node Assignment: C1

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller Ring	RING	C1C0	LINE	LCC	2500	50		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node B	BNODE	C200	NODE	CF	2F	RING			
Node C	CNODE	C300	NODE	CF	4F	RING			
Node D	DNODE	C400	NODE	CF					
Node E	ENODE	C500	NODE	CM					
Node F	FNODE	C600	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

____HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
2 Terminals/Nodes	× 88 = 176 bytes
3 Remote Stations	× 32 = 96 bytes

Figure 134. Example 11: Node A Line/Terminal/Node Configuration Worksheet

2. In node C, $.IO0AB0 reads the data and passes it to the message dispatcher. The message dispatcher puts the message on the queue of the station to which the vector station with network address C600 is linked, DNODE. $.IO0AB0 retrieves the message and sends it over the second ring (because station DNODE is associated with line station RING2, whose device address is A8).

3. In node D, $.IO0AB0 reads the data and passes it to the message dispatcher. The message dispatcher puts the message on the queue of the station to which the vector station with network address C600 is linked, BSCLINE. $.IO0A10 retrieves the message and sends it over the BSC line.

4. In node F, $.IO0A10 reads the data and passes it to the message dispatcher. The message dispatcher puts the message on the queue of the destination station, the one with network address C699.

Station Definitions

Figure 134 on page 376 through Figure 138 on page 381 show the line and node station definitions for the five Communications Facility nodes.

As shown in earlier examples, you must also define the stations that send and receive messages and the remote stations to which local stations send messages. Assume, for example, that a terminal in node A communicates with an application program in node F. You must define the application program as a remote station in node A and the terminal as a remote station in node F. You don't need to define either station in the intermediate nodes through which messages are routed.

Line/Terminal/Node Configuration Worksheet

System: Example 11: Multiple Rings and BSC Connections, Node B , Node 3
Node Assignment: C2

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller Ring	RING	C2C0	LINE	LCC	2500	50		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node A	ANODE	C100	NODE	CF	1F	RING			
Node C	CNODE	C300	NODE	CF	4F	RING			
Node D	DNODE	C400	NODE	CF					
Node E	ENODE	C500	NODE	CM					
Node F	FNODE	C600	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

______HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
2 Terminals/Nodes	× 88 = 176 bytes
3 Remote Stations	× 32 = 96 bytes

Figure 135. Example 11: Node B Line/Terminal/Node Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 11: Multiple Rings and BSC Connections, Node C , Node 3
Node Assignment: C3

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller Ring 1	RING1	C3C0	LINE	LCC	2500	50		
Local Communications Controller Ring 2	RING2	C3C1	LINE	LCC	2500	A8		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node A	ANODE	C100	NODE	CF	1F	RING1			
Node B	BNODE	C200	NODE	CF	2F	RING1			
Node D	DNODE	C400	NODE	CF	4F	RING2			
Node E	ENODE	C500	NODE	CM	1F	RING2			
Node F	FNODE	C600	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = ______ bytes
2 Other Lines	× 264 = 528 bytes
4 Terminals/Nodes	× 88 = 352 bytes
1 Remote Station	× 32 = 32 bytes

Figure 136. Example 11: Node C Line/Terminal/Node Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 11: Multiple Rings and BSC Connections, Node D , Node 3
Node Assignment: C4

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Local Communications Controller Ring	RING	C4C0	LINE	LCC	2500	50		
BSC line	BSCLINE	C429	LINE	CPU	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node A	ANODE	C100	NODE	CF					
Node B	BNODE	C200	NODE	CF					
Node C	CNODE	C300	NODE	CF	2F	RING			
Node E	ENODE	C500	NODE	CM	1F	RING			
Node F	FNODE	C600	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = ______ bytes
2 Other Lines	× 264 = 528 bytes
2 Terminals/Nodes	× 88 = 176 bytes
3 Remote Stations	× 32 = 96 bytes

Figure 137. Example 11: Node D Line/Terminal/Node Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 11: Multiple Rings and BSC Connections, Node F , Node 3
Node Assignment: C6

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
BSC line	BSCLINE	C629	LINE	CPU	2500			

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node A	ANODE	C100	NODE	CF					
Node B	BNODE	C200	NODE	CF					
Node C	CNODE	C300	NODE	CF					
Node D	DNODE	C400	NODE	CF					
Node E	ENODE	C500	NODE	CM					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = _____ bytes
1 Other Line	× 264 = 264 bytes
_____Terminals/Nodes	× 88 = _____ bytes
5 Remote Stations	× 32 = 160 bytes

Figure 138. Example 11: Node F Line/Terminal/Node Configuration Worksheet

Example 12: X.25 Connection between Series/1s

Figure 139 on page 383 shows a configuration in which two Series/1s are connected by an HDLC line. The application program in node 1 communicates with users at 3101 terminals in either node. The 3101 terminals are attached to the Series/1s by feature programmable communications adapters.

You can use such a configuration for transaction processing by defining local and remote program dispatcher stations in each Series/1s, as shown in "Example 10: BSC Point-to-Point Connection with Transaction Processing" on page 366. However, you cannot use remote disk or remote IPL over an X.25 connection.

Configuration Description

Look at the station definition worksheets for example 12. Each node has a unique node assignment, and there are definitions of all remote stations to which local stations will send messages.

In each node, the remote node is defined as a node station with no ring address or line name. The circuit usage is CF, and the circuit stations in the two nodes have the same logical channel ID (LCI), as is required for proper message routing. The line station in one node is subtype DCE, and the line station in the other node is subtype DTE, as required when there is no X.25 packet-switching network between the two nodes. The circuit station subtype is PVC; this is the only circuit supported by the Communications Facility with DCE line stations.

When these stations have been defined, linked as in Figure 139 on page 383, and started, a display of the started stations in node 2 shows this information:

STATION	TYPE	NA	LINK
$.DISP	0000	0200	0000
LINE2	0AB8	0270	0000
CIRC2	16BD	02C2	0000
NODE1	0800	0100	02C2
APROG	0800	01AC	0000
TERM2A	06F0	0260	01AC
TERM2B	06F0	0261	01AC

Notice that:

- The network address of $.DISP (the system station) shows that the node assignment is 02. This is the result of issuing a CP SET NODE command.
- The remote station (APROG) has a vector station block (type 08). This is the result of starting a station whose type is other than NODE and whose network address is not in this node.
- The node station (NODE1) has a vector station block (type 08). This is the result of starting station whose type is NODE, but which doesn't have an associated line station.

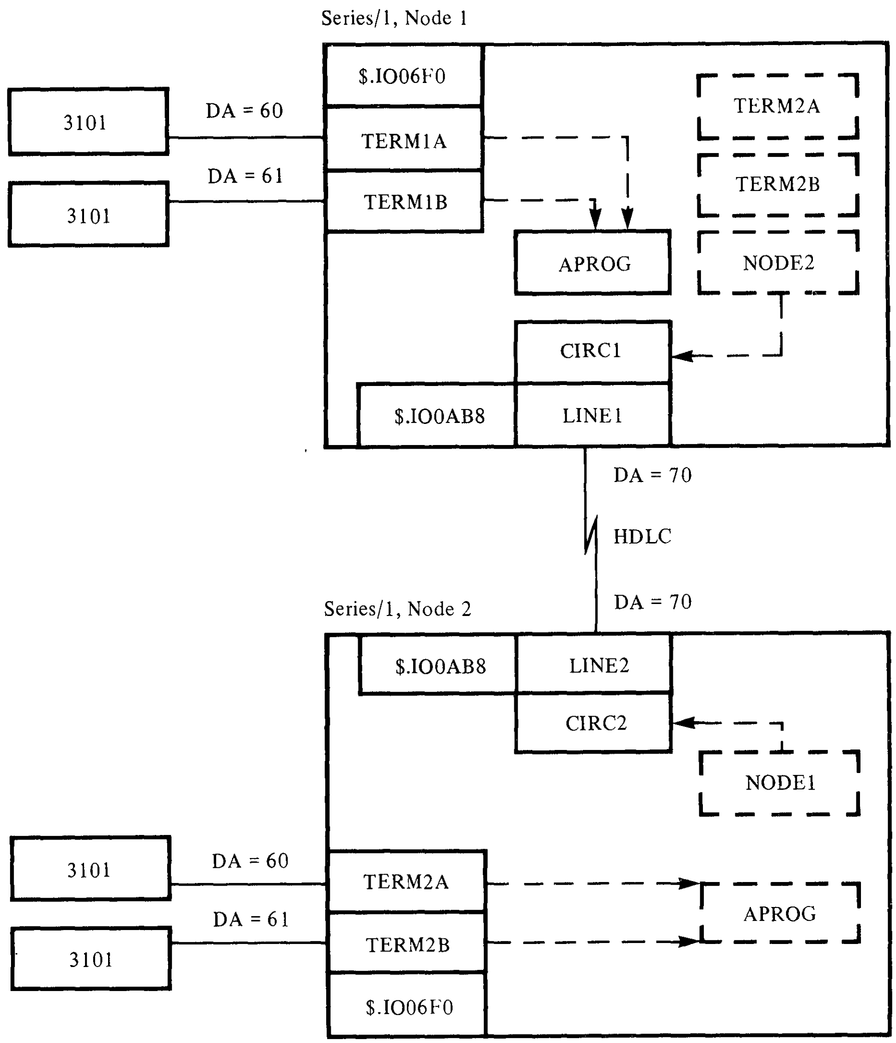


Figure 139. Example 12: X.25 Connection between Series/1s

Message Flow

The flow of messages between the terminals in node 1 and the application program is as described for example 5. The flow of messages between nodes 1 and 2 is similar to that described for example 6.

A message from TERM2A is routed to the application program in node 1 as follows. The 3101 IOCP, $.IO06F0, sends the message to the origin station's link, APROG, whose network address is 01AC. The message dispatcher determines that the address is not in this node, sets the second byte to 00, and looks for a station with that network address (0100). Station NODE1 satisfies the search.

NODE1 is a vector station, so the message dispatcher puts the message on the queue of the station to which NODE1 is linked. That station is the one with network address 02C2 (CIRC2).

The method of routing messages between nodes accounts for the restriction that you can't have more than one active connection (BSC point-to-point, Local Communications Controller, or X.25) through circuit stations with usage CF between two Series/1s. For all these connections, a station with network address *nn*00 is required to route messages to node *nn*, and you can't start more than one station with a given network address.

Subsequently, the X.25 IOCP, $.IO0AB8, retrieves the message from CIRC2's queue and transmits it over the HDLC line represented by LINE2. It writes the message over logical channel 001, as specified in the definition of station CIRC2. It writes the message header, as well as the message data, because CIRC2's usage is CF. $.IO0AB8 in node 1 reads the message from logical channel 001 and finds the circuit station with that logical channel ID (CIRC1). CIRC1's usage is CF, so $.IO0AB8 sends the message with its header to the message dispatcher. The routing information in the message header shows that the origin is 0260 and the destination is 01AC. The message dispatcher determines that the destination network address is in this node, so it puts the message on APROG's queue. The response from APROG to TERM2A is routed in a similar manner.

In summary, to enable communication between nodes connected by an X.25 connection, you must:

- Give each node a unique node assignment.

- In each node, define the remote stations to which local stations will send messages. Be sure that the network address is the same as that of the corresponding real station in the remote node. The station names need not be the same.

- Define the line in one node as subtype DCE and the line in the other node as subtype DTE.

- In each node, define a PVC circuit station with usage CF. Be sure that the two circuit stations have the same logical channel ID.

- In each node, define the remote node station and link it to the circuit station.

- When you start the Communications Facility, issue a CP SET NODE command to set the node assignment before starting any remote stations.

X.25 Call ID Data Set

An X.25 call ID data set ($.SYSX25) is required in each node, even though it is not used for permanent virtual circuits. It is used for switched virtual circuit call establishment. The data set contains one record of /*.

Station Definitions

Figure 140 through Figure 147 on page 389 show the station definitions for the two nodes in this configuration.

User/Message/Alias/Volume Configuration Worksheet

System: Example 12: X.25 Connection between Series/1s, Node 1
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
3101-FPCA IOCP	$.IO06F0	01AA	USER	
X.25 IOCP	$.IO0AB8	01AB	USER	
Application program	APROG	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

3 User/Msg Stations × 88 = 264 bytes
_____Alias Stations × 40 = _______ bytes
_____Remote Stations × 32 = _______ bytes
_____Volume Stations × 104 = _______ bytes

Figure 140. Example 12: Node 1 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 12: X.25 Connection between Series/1s, Node 1
Node Assignment: 01

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM1A	0160	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
3101	TERM1B	0161	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
Remote 3101	TERM2A	0260	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
Remote 3101	TERM2B	0261	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

2 3101F Devices	× 160 = 320 bytes
______4978/Printer Devices	× 264 = ______ bytes
______3101 Devices	× 304 = ______ bytes
2 Remote Stations	× 32 = 64 bytes

Figure 141. Example 12: Node 1 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 12: X.25 Connections between Series/1s, Node 1
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
HDLC Line	LINE1	0170	LINE	DCE	2526		128	2

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Remote Node	NODE2	0200	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

1 HDLC Line	× 112 = 112 bytes
_____Other Lines	× 264 = _______ bytes
_____Terminals/Nodes	× 88 = _______ bytes
1 Remote Station	× 32 = 32 bytes

Figure 142. Example 12: Node 1 Line/Terminal/Node Configuration Worksheet

Circuit Configuration Worksheet

System: Example 12: X.25 Connection between Series/1s, Node 1
Node Assignment: 01

PVC Circuit Definition

Description	Name	NA	Type	Circuit Type	Line	Usage	LCI
PVC Circuit	CIRC1	01C1	CIRC	PVC	LINE1	CF	001

S$POOL Storage

1 Circuit × 128 = 128 bytes

Figure 143. Example 12: Node 1 Circuit Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 12: X.25 Connection between Series/1s, Node 2
Node Assignment: 02

Description	Name	NA	Type	Alias[1]
3101-FPCA IOCP	$.IO06F0	02AA	USER	
X.25 IOCP	$.IO0AB8	02AB	USER	
Remote application program	APROG	01AC	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
_____Alias Stations	× 40 = _____ bytes
1 Remote Station	× 32 = 32 bytes
_____Volume Stations	× 104 = _____ bytes

Figure 144. Example 12: Node 2 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 12: X.25 Connection between Series/1s, Node 2
Node Assignment: 02

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM2A	0260	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
3101	TERM2B	0261	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

2 3101F Devices	× 160 = 320 bytes
_____4978/Printer Devices	× 264 = _____ bytes
_____3101 Devices	× 304 = _____ bytes
_____Remote Stations	× 32 = _____ bytes

Figure 145. Example 12: Node 2 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 12: X.25 Connection between Series/1s, Node 2
Node Assignment: 02

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
HDLC Line	LINE2	0270	LINE	DTE	2526		128	2

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Remote Node	NODE1	0100	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

1 HDLC Line	× 112 = 112 bytes
____Other Lines	× 164 = ______ bytes
____Terminals/Nodes	× 88 = ______ bytes
1 Remote Station	× 32 = 32 bytes

Figure 146. Example 12: Node 2 Line/Terminal/Node Configuration Worksheet

Circuit Configuration Worksheet

System: Example 12: X.25 Connection between Series/1s, Node 2
Node Assignment: 02

PVC Circuit Definition

Description	Name	NA	Type	Circuit Type	Line	Usage	LCI
PVC Circuit	CIRC2	02C2	CIRC	PVC	LINE2	CF	001

S$POOL Storage

1 Circuit × 128 = 128 bytes

Figure 147. Example 12: Node 2 Circuit Configuration Worksheet

EDX Definitions

Figure 148 shows the EDX definitions for either node in this example. Notice that the 3101s are not defined to EDX, so they can be used only as Communications Facility terminals. If you also want to use them as EDX terminals, use TERMINAL statements to define them.

Note that the label of the XHCDEV statement must be the same as the line station name and must be six characters or less. To use the EDX definitions for NODE2, just change LINE1 to LINE2.

```
*** EXAMPLE 12,  NODE 1
***
$EDXDEF    CSECT
           SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
                COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                    X
                INITMOD=(CTZOII,CSXINIT)
              •
              •
$SYSPRTR   TERMINAL DEVICE=4973,ADDRESS=01,END=YES
              •
              •
LINE1      XHCDEV   (71,70),ADAPTER=2080,END=YES
           END
```

Figure 148. Example 12: EDX Definitions

Example 13: X.25 Connection Between Series/1s through an X.25 Network

Figure 149 on page 393 shows a configuration in which three Series/1s are connected by HDLC lines through an X.25 packet-switching network. The application program in node 1 communicates with with users at 3101 terminals in any of the nodes. The 3101 terminals are attached to the Series/1s by feature programmable communications adapters.

You can use such a configuration for transaction-processing by defining local and remote program dispatcher stations in each Series/1, as shown in " Example 9: Local Communications Controller Connection with Transaction Processing" on page 354. However, you cannot use remote disk or remote IPL over an X.25 connection.

Configuration Description

Figure 153 on page 396 through Figure 164 on page 404 show the station definitions for this example. As for example 12, all the circuit stations are defined with usage=CF, each node has a unique node assignment, and there are definitions of all remote stations to which local stations sends messages. The differences appear in the $.SYSX25 definition, the line/terminal/node configuration worksheets and the circuit configuration worksheets. The line stations all have subtype DTE; this is required when the connection is through an X.25 packet-switching network.

The most important difference between example 12 and example 13 is that the circuit stations in example 12 are of subtype PVC while the circuit stations in example 13 are of subtype SVC. In example 12, the connection between the nodes is through the specification of the same logical channel ID (LCI). With SVC circuit stations, the logical channel used is assigned by XHCS and the connection between logical channels is done by the X.25 network. This logical connection, or virtual call, must be established before messages can flow correctly.

Notice that:

- CIRC1A and CIRC1B are defined with contact=INIT; the X.25 IOCP, $.IO0AB8 sends a call request packet to the network when these stations are started.

- CIRC1A has call ID of 2, which is associated with node 2's X.25 network address. CIRC1B has call ID of 3, which is associated with node 3's X.25 network address. CIRC1A represents the logical connection between nodes 1 and 2; CIRC1B represents the logical connection between nodes 1 and 3.

- CIRC2A and CIRC3B are defined with contact=WAIT; when they are started, $.IO0AB8 waits for an incoming call packet from the X.25 network. Because both CIRC2A and CIRC3B have call ID of 0. $.IO0AB8 accepts a call from any X.25 network address.

- CIRC2B like CIRC2A, is defined with contact=WAIT and call ID of 0. However, they are defined with different protocol ID's. CIRC2A has the same protocol ID as CIRC1A; the result is that the virtual call is made between node 1 and 2 over CIRC1A and CIRC2A instead of CIRC1A and CIRC2B.

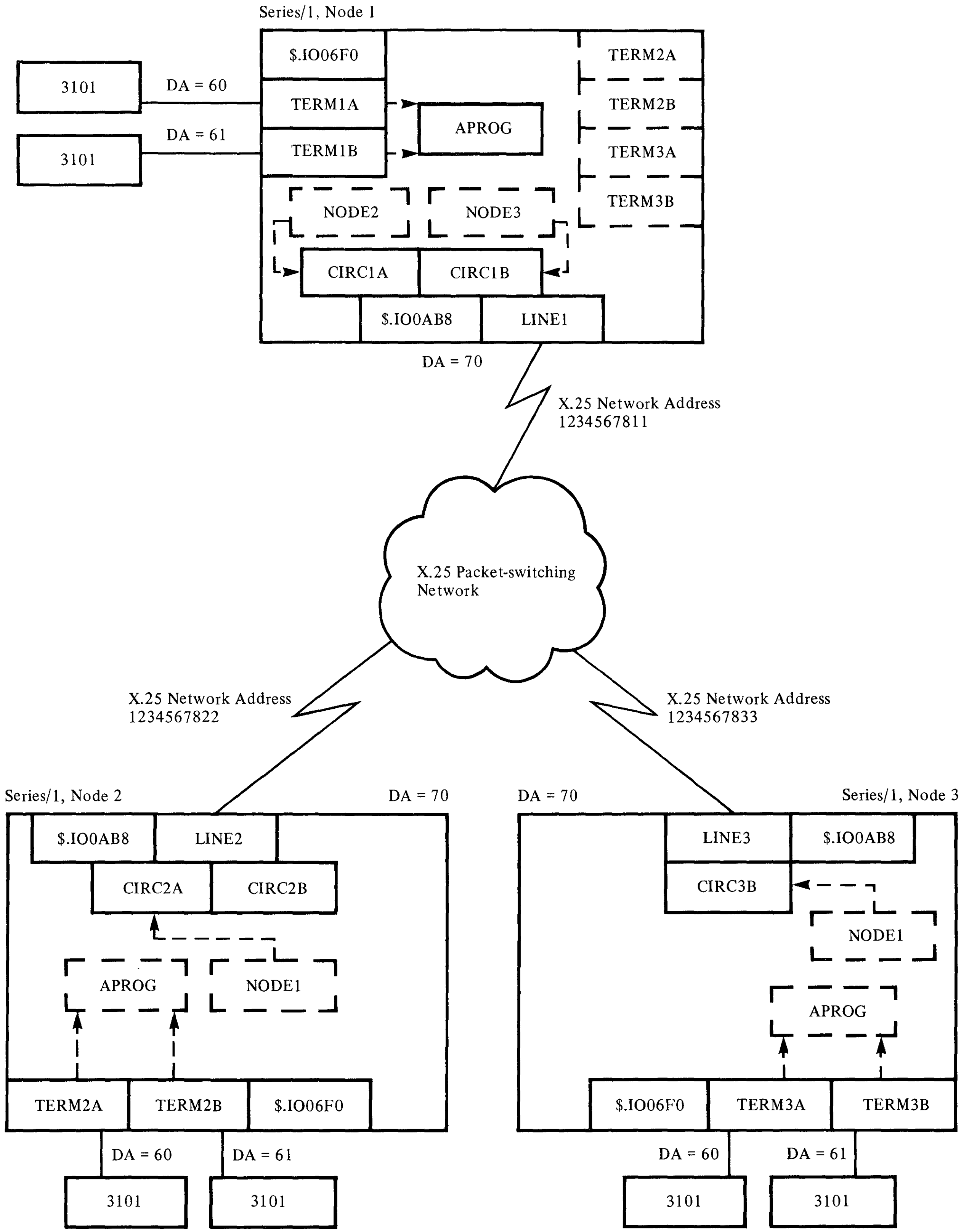


Figure 149. Example 13: X.25 Connection between Series/1s through an X.25 Network

Establishing Virtual Calls

When these stations have been defined, linked as in Figure 148 on page 390, and started, the logical connection between node 1 and node 2 is established as follows:

- When CIRC2A is started, $.IO0AB8 waits for an incoming call packet from the X.25 network. $.IO0AB8 accepts a call from any X.25 network address with the protocol ID as specified in CIRC2A's station block.

- When CIRC1A is started, $.IO0AB8 sends a call request packet to the X.25 network. The X.25 network address associated with call ID 2 and the protocol ID defined in CIRC1A's station block is sent on the call request packet.

- When the X.25 network receives the call request packet, it determines the path to node 2 (using the X.25 network address on the call request packet) and sends an incoming call packet to node 2.

- $.IO0AB8 in node 2 receives the incoming call packet for CIRC2A because CIRC2A's protocol ID matches the protocol ID on the incoming call. $.IO0AB8 accepts the call by sending a call accept packet to the network.

- When the X.25 network receives the call accept packet, it sends a call connected packet to node 1. At this point the virtual call is established.

The connection between node 1 and 3 is established in a similar way.

If the connection between node 2 and the X.25 network is not ready when the call request packet is received by the X.25 network, the network clears the call by sending a clear indication packet to the calling node. If CIRC2A is not active when the incoming call packet is received, XHCS clears the call. If CIRC1A is in RETRY mode, $.IO0AB8 keeps sending the call request packet until the virtual call is established or the circuit station is stopped. If CIRC1A is in STOP mode, the circuit station is stopped.

Message Flow

After the virtual calls are established, a message from TERM2A or TERM3A is routed to the application program in the same way as in example 12.

In summary, to enable communication between nodes over an X.25 packet-switching network, you must:

- Give each node a unique node assignment.

- In each node, define the remote stations to which local stations sends messages. Be sure that the network address is the same as that of the corresponding real station in the remote node. The station names need not be the same.

- In each node, define the node station to which the local node is connected. Do not specify ring address or line name. Link the remote node to the circuit station that represents the logical connection to the remote node.

- Edit $.SYSX25 to include the call IDs and their associated X.25 network addresses.

- In each node, define the line station with subtype DTE.
- Decide which node will initiate virtual call establishment. Define the circuit station in this node with contact=INIT. Specify the call ID of the remote node and the protocol ID required by the circuit station in the remote node. Define the circuit station in the remote node with contact=WAIT and either the call ID of the local node or 0 if incoming call packets can be received from any X.25 network address. If there are two or more circuit stations with contact=WAIT, define a unique protocol ID to separate incoming call packets.
- When you start the Communications Facility, issue a CP SET NODE command to set the node assignment before starting any remote stations.
- Start the circuit stations with contact=WAIT before starting those with contact=INIT. Otherwise use the CP F MODE command to put the circuit stations with contact=INIT into RETRY mode.

X.25 Call ID Data Set

An X.25 call ID data set ($.SYSX25) is required in all three nodes for use by the X.25 IOCP, as in Figure 150. Node 1's circuits initiate calls to two different call IDs, so both must be included in its $.SYSX25 data set. Node 2's circuit is defined with call ID 0, so it doesn't require a call ID in its data set. (But it still must have a $.SYSX25 data set.) Node 3's circuit waits for a call from call ID 1, so its $.SYSX25 data set must include call ID 1.

```
2  Node2  1234567822 San Francisco shipping
3  Node3  1234567833 Tokyo warehouse
/*
```

Figure 150. Example 13: Node 1 $.SYSX25 Data Set

```
/*
```

Figure 151. Example 13: Node 2 $.SYSX25 Data Set

```
1  Node1 1234567811 Sydney shipping
/*
```

Figure 152. Example 13: Node 3 $.SYSX25 Data Set

Station Definitions

Figure 153 through Figure 164 on page 404 show the station definitions for this example.

User/Message/Alias/Volume Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 1
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
X.25 IOCP	$.IO0AB8	01AA	USER	
3101-FPCA IOCP	$.IO06F0	01AB	USER	
Application program	APROG	01AC	USER	
Remote application in node 2	APROG	02AB	USER	
Remote application in node 3	APROG	03AB	USER	

[1] Specify only for alias station

S$POOL Storage

3 User/Msg Stations	× 88 = 264 bytes
_____Alias Stations	× 40 = ______ bytes
2 Remote Stations	× 32 = 64 bytes
_____Volume Stations	× 104 = ______ bytes

Figure 153. Example 13: Node 1 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 1
Node Assignment: 01

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM1A	0160	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
3101	TERM1B	0161	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
Remote 3101	TERM2A	0260	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
Remote 3101	TERM2B	0261	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
Remote 3101	TERM3A	0360	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
Remote 3101	TERM3B	0361	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485's, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

2 3101F Devices	× 160 = 320 bytes
______4978/Printer Devices	× 264 = ______ bytes
______3101 Devices	× 304 = ______ bytes
4 Remote Stations	× 32 = 128 bytes

Figure 154. Example 13: Node 1 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 1
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
HDLC line	LINE1	0170	LINE	DTE	2526		128	2

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Remote node 2	NODE2	0200	NODE	CF					
Remote node 3	NODE2	0300	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

1 HDLC Line	× 112 = 112 bytes
______Other Lines	× 264 = ______ bytes
______Terminals/Nodes	× 88 = ______ bytes
2 Remote Stations	× 32 = 64 bytes

Figure 155. Example 13: Node 1 Line/Terminal/Node Configuration Worksheet

Circuit Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 1
Node Assignment: 01

SVC Circuit Definition

Description	Name	NA	Type	Circuit Type	Line	Usage	Contact	Call ID	Packet Size	Window	Protocol ID[1]	Facilities	User Data[2]
SVC circuit	CIRC1A	01C2	CIRC	SVC	LINE1	CF	INIT	2			C0000001		
SVC circuit	CIRC1B	01C3	CIRC	SVC	LINE1	CF	INIT	3			C0000001		

[1] Specify only for contact type INIT or WAIT
[2] Specify only for contact type INIT

S$POOL Storage

2 Circuits × 128 = 256 bytes

Figure 156. Example 13: Node 1 Circuit Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 2
Node Assignment: 02

Description	Name	NA	Type	Alias[1]
X.25 IOCP	$.IO0AB8	02AA	USER	
3101-FPCA IOCP	$.IO06F0	02AB	USER	
Remote application program in node 1	APROG	01AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations × 88 = 176 bytes
_____Alias Stations × 40 = _____ bytes
1 Remote Station × 32 = 32 bytes
_____Volume Stations × 104 = _____ bytes

Figure 157. Example 13: Node 2 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 2
Node Assignment: 02

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM2A	0260	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
3101	TERM2B	0261	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485's, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

2 3101F Devices	× 160 = 320 bytes
_____4978/Printer Devices	× 264 = _____ bytes
_____3101 Devices	× 304 = _____ bytes
_____Remote Stations	× 32 = _____ bytes

Figure 158. Example 13: Node 2 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 2
Node Assignment: 02

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
HDLC line	LINE2	0270	LINE	DTE	2526		128	2

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Remote node 1	NODE1	0100	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

1 HDLC Line	× 112 = 112 bytes
_____Other Lines	× 264 = _____ bytes
_____Terminals/Nodes	× 88 = _____ bytes
1 Remote Station	× 32 = 32 bytes

Figure 159. Example 13: Node 2 Line/Terminal/Node Configuration Worksheet

Circuit Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 2
Node Assignment: 02

SVC Circuit Definition

Description	Name	NA	Type	Circuit Type	Line	Usage	Contact	Call ID	Packet Size	Window	Protocol ID[1]	Facilities	User Data[2]
SVC circuit	CIRC2A	02C1	CIRC	SVC	LINE2	CF	WAIT	0			C0000001		
SVC circuit	CIRC2B	02C3	CIRC	SVC	LINE2	CF	WAIT	0			C0000003		

[1] Specify only for contact type INIT or WAIT
[2] Specify only for contact type INIT

S$POOL Storage

2 Circuits × 128 = 256 bytes

Figure 160. Example 13: Node 2 Circuit Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 3
Node Assignment: 03

Description	Name	NA	Type	Alias[1]
X.25 IOCP	$.IO0AB8	03AA	USER	
3101-FPCA IOCP	$.IO06F0	03AB	USER	
Remote application program in node 1	APROG	01AB	USER	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations × 88 = 176 bytes
_____Alias Stations × 40 = _______ bytes
1 Remote Station × 32 = 32 bytes
_____Volume Stations × 104 = _______ bytes

Figure 161. Example 13: Node 3 User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 3
Node Assignment: 03

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
3101	TERM3A	0360	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y
3101	TERM3A	0361	DEVICE	3101F	2500		4040	$SYSPRTR	4800	Y

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485's, bit rate must be 9600 and AUTO ANS=NO

S$POOL Storage

2 3101F Devices	× 160 = 320 bytes
_____4978/Printer Devices	× 264 = _____ bytes
_____3101 Devices	× 304 = _____ bytes
_____Remote Stations	× 32 = _____ bytes

Figure 162. Example 13: Node 3 Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 3
Node Assignment: 03

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
HDLC line	LINE3	0370	LINE	DTE	2526		128	2

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
Node 1	NODE1	0100	NODE	CF					

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

1 HDLC Line	× 112 = 112 bytes
_____Other Lines	× 264 = _____ bytes
_____Terminals/Nodes	× 88 = _____ bytes
1 Remote Station	× 32 = 32 bytes

Figure 163. Example 13: Node 3 Line/Terminal/Node Configuration Worksheet

Circuit Configuration Worksheet

System: Example 13: X.25 Connection between Series/1s through an X.25 Network, Node 3
Node Assignment: 03

SVC Circuit Definition

Description	Name	NA	Type	Circuit Type	Line	Usage	Contact	Call ID	Packet Size	Window	Protocol ID[1]	Facilities	User Data[2]
SVC circuit	CIRC3B	03C1	CIRC	SVC	LINE3	CF	WAIT	1			C0000001		

[1] Specify only for contact type INIT or WAIT
[2] Specify only for contact type INIT

S$POOL Storage

1 Circuit × 128 = 128 bytes

Figure 164. Example 13: Node 3 Circuit Configuration Worksheet

EDX Definitions

Figure 165 shows the EDX definitions for node 1, 2, or 3. Just change LINE1 to LINE2 for node 2 or to LINE3 for node 3.

```
*** EXAMPLE 13 NODE 1, 2, or 3
***
$EDXDEF   CSECT
          SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),          X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                   X
               INITMOD=(CTZOII,CSXINIT)
            •
            •
          DISK     DEVICE=4962-3,ADDRESS=03,                           X
               VOLNAME=(EDX002,EDX003,ASMLIB),TASK=YES,END=YES
            •
            •
LINE1     XHCDEV   (71,70),ADAPTER=2080,END=YES
          END
```

Figure 165. Example 13: EDX Definitions

Example 14: X.25 Connection between a Series/1 and DTEs

Figure 166 on page 407 shows a configuration in which a Series/1 (Node 1) is connected to three X.25 DTEs (Node 2, Remote 1 and Remote 2). In this example, Node 2 is a Series/1s running the Communications Facility; Remote 1 and Remote 2 are any equipment that can function as X.25 DTEs. Between node 1 and node 2, application programs APROG2A and APROG2B communicate with PROG3A and PROG3B, respectively. The application program in Remote 1 communicates with application program PROG1A. The application program in Remote 2 communicates with application programs PROG1B1 and PROG1B2.

Configuration Description

Look at the station definition worksheets for example 14. As in example 12, LINE2 is defined with subtype of DCE and LINE3 is defined with subtype of DTE; this is required when the connection is not through an X.25 network. The circuit stations CIRC2A and CIRC3A are defined with subtype PVC and have the same logical channel ID of 001. The circuit stations CIRC2B and CIRC3B are also defined with subtype PVC and they have the same logical channel ID of 002. As in example 13, LINE1 is defined with subtype of DTE; this is required for connections through an X.25 network.

Notice the following:

- The circuit stations CIRC2A, CIRC3A, CIRC2B and CIRC3B are defined with usage of STD; this means that only data messages may flow between the circuit stations and the application programs. X.25 control messages sent to the circuit stations are rejected by $.IO0AB8, and $.IO0AB8 does not send any X.25 control messages to the application programs.

- The circuit stations CIRC1A and CIRC1B are defined with usage of STD+; this means that X.25 control messages sent to the circuit stations are accepted and processed by $.IO0AB8. Also, if appropriate, $.IO0AB8 sends X.25 control messages to the application programs.

- The circuit stations CIRC1A, CIRC1B, CIRC2A, CIRC3A, CIRC2B and CIRC3B are linked to their respective application programs using the direct link vector (DLV). Data messages read by $.IO0AB8 from the HDLC line are sent to these application programs.

- The circuit station CIRC1B is linked to PROG1B1 using the alternate link vector. This means that all X.25 control messages associated with CIRC1B destined for the system is sent to PROG1B1. If the alternate link vector were not specified, all X.25 control messages would be sent to the station specified by the direct link vector, as in the case of CIRC1A and PROG1A.

- The circuit station CIRC1A is defined with contact of INIT; this means that, as in example 13, $.IO0AB8 automatically sends a call request packet to the X.25 network to initiate a virtual call. Because CIRC1A is defined with call ID 3, this call request packet specifies the X.25 network address of Remote 1. When the virtual call is established, the application program PROG1A communicates with the application in Remote 1.

- The circuit station CIRC1B is defined with contact of USERINIT. This means that $.IO0AB8 waits for a call request control message to be sent to CIRC1B

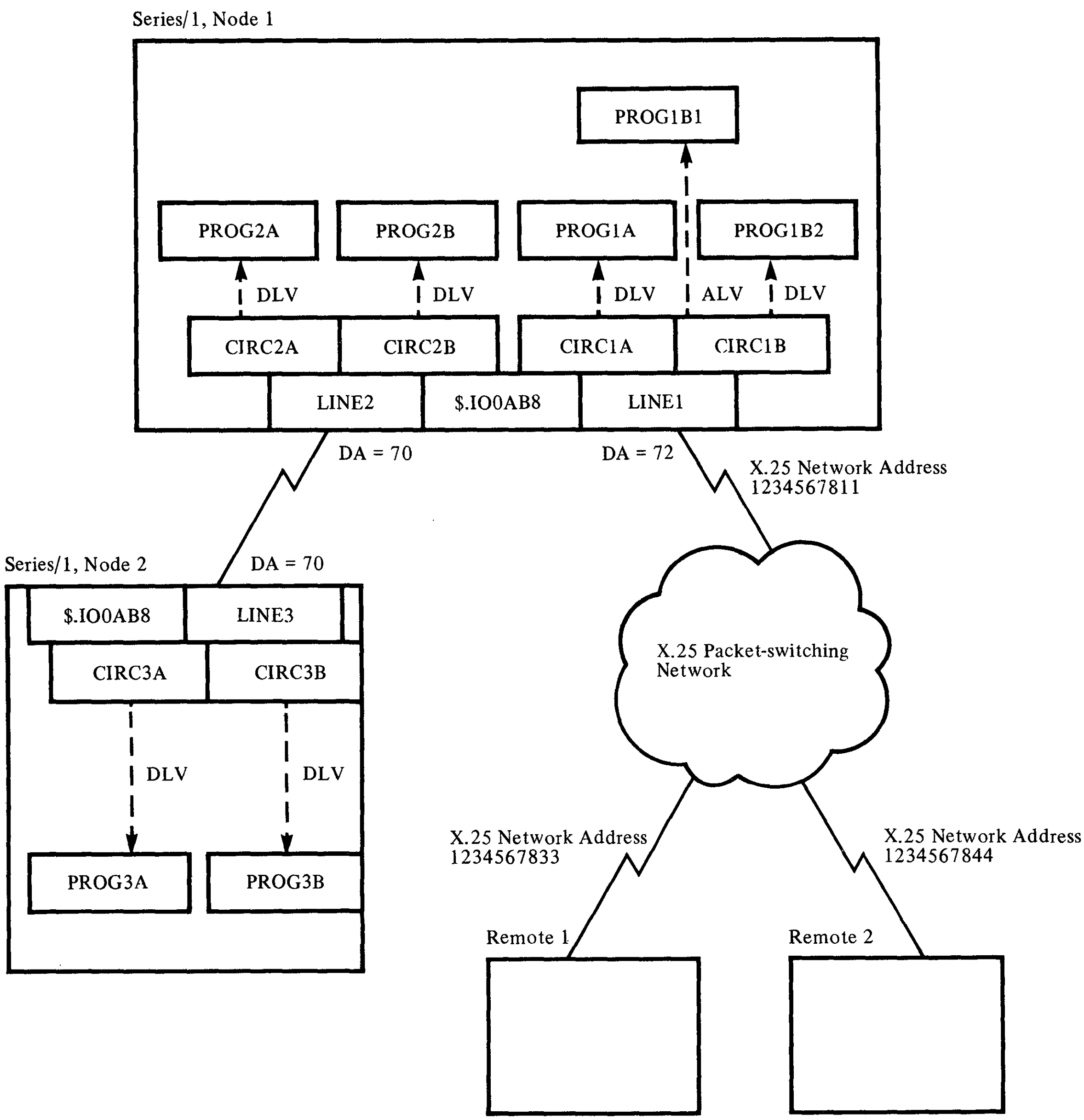


Figure 166. Example 14: X.25 Connection between a Series/1 and DTEs

before sending the call request packet to the X.25 network. The called X.25 network address on the call request packet is obtained from the call request control message.

The routing of messages between the DTEs is similar to that described in two previous examples: messages between Node 1 and Node 2 are routed as described in "Example 12: X.25 Connection between Series/1s" on page 382, and messages between Node 1 and Remote 1 or Remote 2 are described in "Example 13: X.25 Connection Between Series/1s through an X.25 Network" on page 392.

The difference between this example and the other X.25 examples is in the virtual call establishment and in the routing of messages within the Series/1s.

Establishing Virtual Calls

As in example 13, before messages can flow between Node 1 and Remote 1 or Remote 2, virtual calls between the DTEs must be established. Virtual call establishment between Node 1 and Remote 1 is similar to that described in example 13. As in example 13, when CIRC1A is started, $.IO0AB8 sends a call request packet to the X.25 network using information from CIRC1A's station definition.

The only difference between example 13 and this example is that, because CIRC1A is defined with usage of STD+, a call connected control message is sent by $.IO0AB8 to PROG1A when the call connected packet is received. If a clear indication packet is received, $.IO0AB8 sends the clear indication control message to PROG1A.

The virtual call establishment between Node 1 and Remote 2 differs from that described for the virtual call between Node 1 and Remote 1 because CIRC1B is defined with contact of USERINIT. When CIRC1B is started, $.IO0AB8 does not send a call request packet to the X.25 network, instead it waits for a call request control message to be sent to the circuit station.

When a call request control message is sent to the circuit station, the IOCP retrieves it and sends a call request packet to the X.25 network. If the call request is accepted by the called DTE, $.IO0AB8 receives a call connected packet from the X.25 network. A call connected control message is sent to PROG1B1, because it is the circuit station's alternate link. If the call is cleared, and a clear indication packet is received, $.IO0AB8 sends the clear indication control message to PROG1B1.

When the virtual calls are established, the flow of data messages for PROG1A and PROG1B2 from Remote 1 and Remote 2 is as described for PROG2A.

Message Flow

When the stations have been defined, linked as shown in Figure 166 on page 407, and started, a message from PROG3A is routed to PROG2A as follows. After PROG3A sends the message to CIRC3A, $.IO0AB8 retrieves it from CIRC3A's message queue, and transmits it over logical channel 001. $.IO0AB8 in Node 1 is notified that CIRC2A has received a message on its logical channel (channel 001), and reads the message.

Up to this point, the message flow is similar to that in example 12. However, because CIRC2A's usage type is STD, the message is sent to the station specified by the direct link vector, PROG2A. When PROG2A receives it, the message no longer has its original routing information. The origin station is CIRC2A, not PROG3A. When PROG2A sends a response to the message, the response is routed over the same path—from CIRC2A to CIRC3A and then to its direct link, PROG3A. Similarly, a message sent by PROG3B is routed to PROG2B.

Note that a separate path is required between each pair of application programs. This is required because the original routing information is lost. If the messages flowed in only one direction, from Node 2 to Node 1, and, if only one application program in Node 1 were to receive all the messages, then all messages from the Node 2 application programs could be routed through one pair of circuit stations. But, if responses were routed back over the single path, only the direct link of the circuit station in Node 2 would receive the responses.

In this example, node assignment is not required of the Series/1s because there are no remote stations defined, that is, there is no station whose network address identifies it as being in the remote node. You may issue CP SET NODE commands to set node assignments, but you don't have to. If you do not, the node assignment is 00, and the message dispatcher treats all stations as local stations, whatever their network address.

In summary, to enable communication between a Series/1 and a DTE:

- If the connection is through an X.25 packet switching network, define the line with subtype DTE. If the connection is not through an X.25 network, one line station must be defined with subtype DCE and the other line station must be defined with subtype DTE.
- If the application program sends and receives X.25 control messages, define the circuit station with usage of STD+, otherwise, define the circuit station with usage of STD.
- If the application program initiates the virtual call establishment, define the circuit station with contact of USERINIT. Define the circuit station with contact of INIT if you want $.IO0AB8 to initiate the virtual call. If the remote DTE initiates the virtual call establishment, define the circuit station with contact of WAIT.
- Link the circuit stations to the appropriate application programs using the direct link vector. If a separate application program is to receive X.25 control messages, use the alternate link vector.
- Define a separate path for each separate message destination.

X.25 Call ID Data Set

In this example only node 1 has SVC circuits with call IDs defined, so node 1 must have a $.SYSX25 data set containing the two call IDs and their associated X.25 network addresses. Node 2 must have a $.SYSX25 data set with at least one record of /*.

```
3 Remote1 1234567833 New York publishing
4 Remote2 1234567844 London distributing
/*
```

Figure 167. Example 14: Node 1 $.SYSX25 Data Set

/*

Figure 168. Example 14: Node 2 $.SYSX25 Data Set

Station Definitions

Figure 169 through Figure 174 on page 415 show the station definitions for this example.

User/Message/Alias/Volume Configuration Worksheet

System: Example 14: X.25 Connection between Series/1s and DTEs, Node 1
Node Assignment: 01

Description	Name	NA	Type	Alias[1]
X.25 IOCP	$.IO0AB8	01AA	USER	
Data and control message program	PROG1A	01AB	USER	
Data only program	PROG1B2	01AB	USER	
Control message only program	PROG1B1	01AC	USER	
Data only program	PROG2A	01AE	USER	
Data only program	PROG2B	01AF	USER	

[1] Specify only for alias station

S$POOL Storage

6 User/Msg Stations	× 88 = 528 bytes
______Alias Stations	× 40 = ______ bytes
______Remote Stations	× 32 = ______ bytes
______Volume Stations	× 104 = ______ bytes

Figure 169. Example 14: Node 1 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 14: X.25 Connection between Series/1s and DTEs, Node 1
Node Assignment: 01

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
HDLC line to network	LINE1	0172	LINE	DTE	128		128	2
HDLC line to node 2	LINE2	0170	LINE	DCE	128		128	2

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

2 HDLC Lines	× 112 = 224 bytes
_____Other Lines	× 264 = ______ bytes
_____Terminals/Nodes	× 88 = ______ bytes
_____Remote Stations	× 32 = ______ bytes

Figure 170. Example 14: Node 1 Line/Terminal/Node Configuration Worksheet

Circuit Configuration Worksheet

System: Example 14: X.25 Connection between Series/1s and DTEs, Node 1
Node Assignment: 01

PVC Circuit Definition

Description	Name	NA	Type	Circuit Type	Line	Usage	LCI
PVC circuit	CIRC2A	01C1	CIRC	PVC	LINE2	STD	001
PVC circuit	CIRC2B	01C2	CIRC	PVC	LINE2	STD	002

SVC Circuit Definition

Description	Name	NA	Type	Circuit Type	Line	Usage	Contact	Call ID	Packet Size	Window	Protocol ID[1]	Facilities	User Data[2]
SVC circuit	CIRC1A	01C3	CIRC	SVC	LINE1	STD+	INIT	3			C0001234		PASSWORD=CFP
SVC circuit	CIRC1B	01C4	CIRC	SVC	LINE1	STD+	USERINIT	4					

[1] Specify only for contact type INIT or WAIT
[2] Specify only for contact type INIT

S$POOL Storage

4 Circuits × 128 = 512 bytes

Figure 171. Example 14: Node 1 Circuit Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 14: X.25 Connection between Series/1s and DTEs, Node 2
Node Assignment: 02

Description	Name	NA	Type	Alias[1]
X.25 IOCP	$.IO0AB8	02AA	USER	
Data only application program	PROG3A	02AB	USER	
Data only application program	PROG3B	02BC	USER	

[1] Specify only for alias station

S$POOL Storage

3 User/Msg Stations	× 88 = 264 bytes
_____Alias Stations	× 40 = _______ bytes
_____Remote Stations	× 32 = _______ bytes
_____Volume Stations	× 104 = _______ bytes

Figure 172. Example 14: Node 2 User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 14: X.25 Connection between Series/1s and DTEs, Node 2
Node Assignment: 02

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
HDLC line	LINE3	0270	LINE	DTE	128		128	2

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

1 HDLC Line	× 112 = 112 bytes
_____Other Lines	× 264 = _____ bytes
_____Terminals/Nodes	× 88 = _____ bytes
_____Remote Stations	× 32 = _____ bytes

Figure 173. Example 14: Node 2 Line/Terminal/Node Configuration Worksheet

Circuit Configuration Worksheet

System: Example 14: X.25 Connection between Series/1s and DTEs, Node 2
Node Assignment: 02

PVC Circuit Definition

Description	**Name**	**NA**	**Type**	**Circuit Type**	**Line**	**Usage**	**LCI**
PVC 1	CIRC3A	02C1	CIRC	PVC	LINE3	STD	001
PVC 2	CIRC3B	02C2	CIRC	PVC	LINE3	STD	002

S$POOL Storage

2 Circuits × 128 = 256 bytes

Figure 174. Example 14: Node 2 Circuit Configuration Worksheet

EDX Definitions

Figure 175 shows the EDX definitions for node 1 and Figure 176 on page 416 shows the EDX definitions for node 2.

```
*** EXAMPLE 14, NODE 1
         SYSTEM    MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),        X
              COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                  X
              INITMOD=(CTZOII,CSXINIT)
              •
              •
         DISK      DEVICE=4962-3,ADDRESS=03,                         X
              VOLNAME=(EDX002,EDX003,ASMLIB),TASK=YES,END=YES
              •
              •
 LINE1   XHCDEV    (73,72),ADAPTER=2080,END=NO
 LINE2   XHCDEV    (71,70),ADAPTER=2080,END=YES
```

Figure 175. Example 14: Node 1 EDX Definitions

```
*** EXAMPLE 14, NODE 2
         SYSTEM   MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),            X
              COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                     X
              INITMOD=(CTZOII,CSXINIT)
           •
           •
         DISK     DEVICE=4962-3,ADDRESS=03,                             X
              VOLNAME=(EDX002,EDX003,ASMLIB),TASK=YES,END=YES
           •
           •
 LINE3   XHCDEV   (71,70),ADAPTER=2080,END=YES
***
         END
```

Figure 176. Example 14: Node 2 EDX Definitions

Example 15: LAN with Series/1-PC Connect Attachment and EDX Secondary SNA1 Connection

Figure 177 on page 419 shows a configuration in which a Series/1 is connected to a host computer by an EDX Secondary SNA1 connection. The Series/1 appears to the host as an SNA physical unit, two SNA type 2 logical units (terminals), and one SNA type 1 logical unit (printer). The real devices are PCs and PC printers connected by a local area network (LAN). One of the PCs in the LAN is attached to the Series/1 by a Series/1-PC Connect Attachment, and is called the gateway PC.

Using this example a PC user can:

- Pass through to a host as a 3278
- Initiate a host-to-PC file transfer
- Define a PC printer as a 3286 or 3287 printer.

Message Flow

To achieve the flow of messages between the host and the PCs, each remote station and its associated SNA logical unit stations are linked to each other. The remote stations (for example, PC12) are created because they are defined as remote message stations. The PC user must log on to one of these remote stations using the IBM PC Network SNA 3270 Emulation Program.

Once the PC user has logged on to one of these remote stations and the associated SNA logical units have started, communication between the host and the PC can start. When the PC user presses an interrupt key, the IBM PC Network SNA 3270 Emulation Program routes that message over the LAN to the gateway PC. The IBM PC Network SNA 3270 Emulation Program delivers the message to the Series/1-PC Connect Program (5719-CN1) running in the gateway PC. The Series/1-PC Connect Program then places a Personal Computer header on the message. This header contains information about the origin and destination stations. The header also serves as a function identifier which indicates the type of function to be performed. The Series/1-PC Connect Program transmits the message including the message header to the Series/1 through the Series/1-PC Connect Attachment.

On the Series/1, the PC Connect IOCP, $.IO0AA8, receives the message and examines the header to determine which function has been requested. In this case, host communication has been requested.

The PC Connect IOCP converts the Personal Computer header to a Communications Facility header and uses the origin station's direct link as the destination station.

The IOCP handles segmented messages. When it receives the first message in a chain of segmented messages, it obtains a 2K mapped or unmapped storage block. The IOCP stores all the messages from the same origin station in that block until the last message in the chain is received. After receiving and storing the last message in the chain or the only message in a chain, the IOCP sends the message to the origin station's direct link vector.

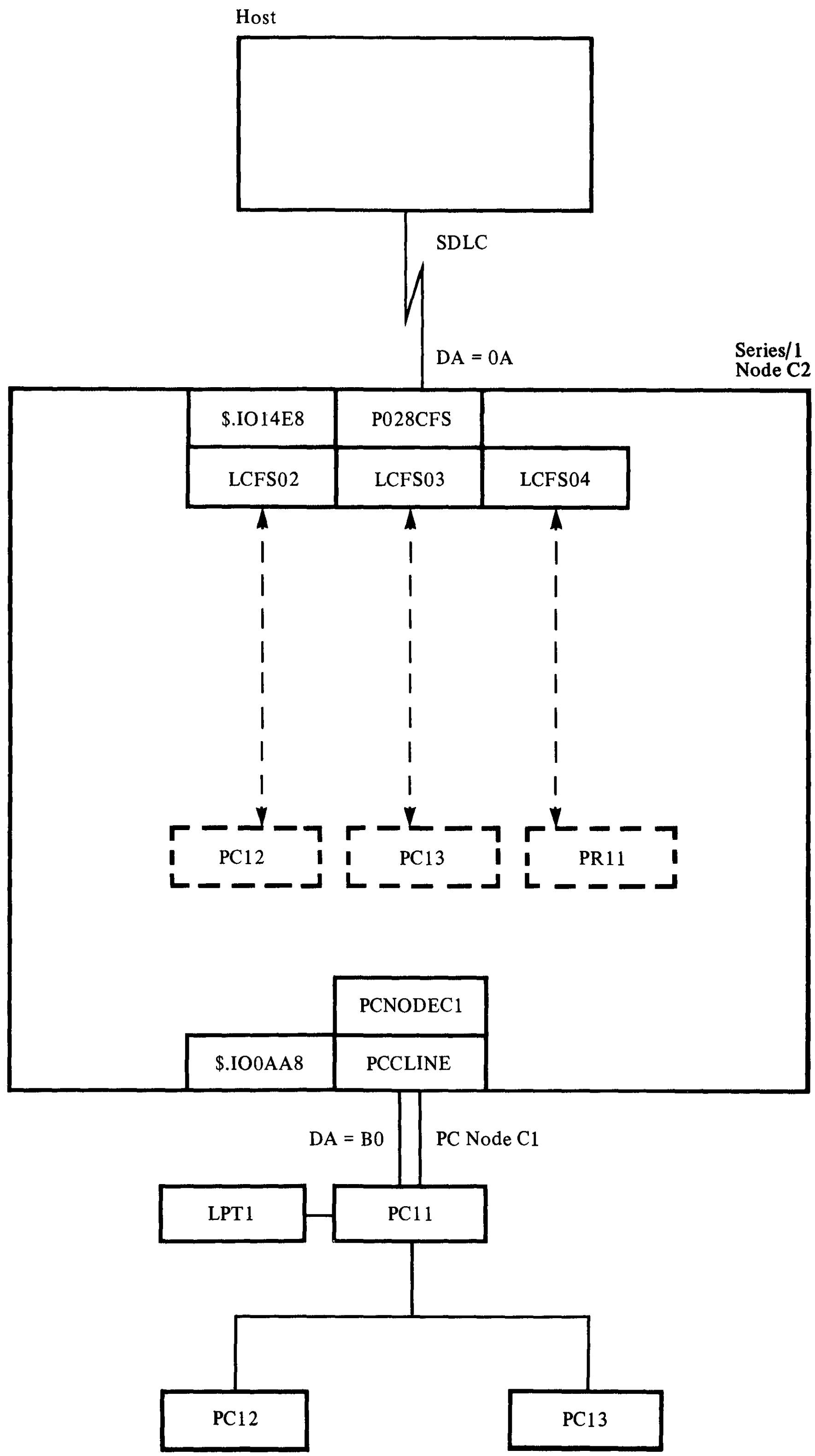


Figure 177. Example 15: LAN with Series/1-PC Connect Attachment and SNA Connection

The SNA IOCP, $.IO14E8, receives data from the host and forwards according to the local address specified for the SNA logical unit stations. For example, data for logical unit 2 is intended for station LCFS02, so it is sent to that station's link, PC12.

$.IO0AA8 receives the message from the PC node station's queue. It converts the Communications Facility message header to a Personal Computer message header. Then it transmits the message to the gateway PC through the Series/1-PC Connect Attachment.

Message routing for file transfer and printer support functions is the same as described for terminals.

Station Definitions

Figure 178 through Figure 180 on page 422 show the station definitions for this example.

User/Message/Alias/Volume Configuration Worksheet

System: Example 15: LAN with Series/1-PC Connect Attachment and SNA Connection
Node Assignment: C2

Description	Name	NA	Type	Alias[1]
SNA IOCP	$.IO14E8	C2AA	USER	
PC Connect IOCP	$.IO0AA8	C2AB	USER	
Remote station for PC12	PC12	C1A1	MESSAGE	
Remote station for PC13	PC13	C1A2	MESSAGE	
Remote station for PR11	PR11	C1A3	MESSAGE	

[1] Specify only for alias station

S$POOL Storage

2 User/Msg Stations	× 88 = 176 bytes
_____Alias Stations	× 40 = _____ bytes
3 Remote Stations	× 32 = 96 bytes
_____Volume Stations	× 104 = _____ bytes

Figure 178. Example 15: User/Message/Alias/Volume Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 15: LAN with Series/1-PC Connect Attachment and SNA Connection
Node Assignment: C2

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Series/1-PC Connect Attachment	PCCLINE	C2C0	LINE	PCC	4096	B0		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
PC Node C1	PCNODEC1	C100	NODE	PC		PCCLINE			

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

_____HDLC Lines	× 112 = ______ bytes
1 Other Line	× 264 = 264 bytes
1 Terminal/Node	× 88 = 88 bytes
_____Remote Stations	× 32 = ______ bytes

Figure 179. Example 15: Line/Terminal/Node Configuration Worksheet

SNA Configuration Worksheet

System: Example 15: LAN with Series/1-PC Connect Attachment and SNA Connection
Node Assignment: C2

PU Definition

Description	Name	NA	Type	Buffer Size	PU #[1]	Host Application	Host Mode
SNA physical unit	P028CFS	C20A	PU	2500		#1: TSO	LU2256
						#2: IMS	STD32741
						#3: CICS	STD32743

LU Definitions

Description	Name	NA	Type	Term Type	PU Name	Local Address	Logon ID	User Data
Emulated terminal	LCFS02	C2E2	LU	3278	P028CFS	2	1	
Emulated terminal	LCFS03	C2E3	LU	3278	P028CFS	3		
Emulated printer	LCFS04	C2E4	LU	3286	P028CFS	4	3	

[1] Valid only for EDX Secondary SNA2

S$POOL Storage

1 Physical Unit	× 128 = 128 bytes
3 Logical Units	× 280 = 840 bytes
_____Remote Stations	× 32 = ______ bytes

Figure 180. Example 15: SNA Configuration Worksheet

Notice the terminal type of the SNA logical units in Figure 180. It must correspond to the session that has been established by the IBM PC Network SNA 3270 Emulation Program and the associated remote station. For example, if PC12 is used to log on to a terminal session, then LCFS02 should be a logical unit type 2.

Notice the logon information in Figure 180. There are three host applications specified: #1 is for TSO/terminal communication; #2 is for IMS/printer communication; and #3 is for CICS/printer communication. An operator at the PC which is logged on to station PC12 is automatically connected to TSO when the LU station is started and the log on process is complete.

An operator at the PC logged on to PC13 station, is presented with a logon prompt screen when the LU station is started and the IBM PC Network SNA 3270 Emulation Program log on procedure is completed. See the "Series/1-to-Host

SNA IOCP, $.IO14E8" on page 169 for an example of the log on screen. The screen shows the applications specified in the SNA physical unit. The terminal user can select option 1 to log on to TSO or option 4 to log on to any other SNA application defined in the host.

The printer is automatically connected to CICS when the logical unit is started.

Station Links

The SNA LU stations must be directly linked to the remote stations that represent the PCs and the printer. The remote stations must also be directly linked to the SNA LU stations.

Station Mode of Operation

The mode of operation for LU stations must be PC. Before starting the LU stations, use the CP F MODE command to change the mode of command to PC for each of the LU stations that are directly linked to the remote stations that represent the PCs. It is recommended that you also set UBRETRY mode for these LU stations. If you specify UBSTOP and an auto-logon session is unbound, the LU station must be started again before the user can log on to the PC again.

EDX Definitions

Figure 181 shows the EDX definitions for example 15. You may not want to map the common area in to all partitions as shown in the example. For example, the partition in which EDX Secondary SNA1 runs does not have to be mapped unless Communications Facility programs also run in that partition.

```
*** EXAMPLE 15
***
$EDXDEF   CSECT
          SYSTEM    STORAGE=256,MAXPROG=(10,10,10,10),PARTS=(32,32,32,32),     X
               COMMON=(CSXEND,CSXEND,CSXEND,CSXEND),                           X
               INITMOD=(CSPCCIPL,CSXINIT)
            •
            •
          PCC       ADDRESS=B0
          PCC       ADDRESS=B1
          PCC       ADDRESS=B2,END=YES
SNALINE   EXIODEV   ADDRESS=0A,RSB=4,MAXDCB=6,END=YES
          END
```

Figure 181. Example 15: EDX Definitions

PC Installation and Setup

This section describes the installation and setup procedures for the three PCs and the printer in this example. The gateway PC should be installed and set up before the other PCs in the network.

Gateway PC

See the *Series/1-PC Connect for the Event Driven Executive Communications Facility* for instructions on installing the Series/1-PC Connect Attachment on a PC for a 3270 communications gateway. After completing this procedure, you must use the Series/1-PC Connect customization program, S1PCCUST.BAS, to specify the following:

Option 7: Enter AA as the gateway PC's LAN name.
Option 5: Enter PC12, PC13, and PR11 in the gateway PC's 3270 names list.
Option 10: Use this option to customize certain technical characteristics of the PC Connect.

In the new menu presented for option 10, do the following:

Option 4: Enter N to disable inter-LAN message support.
Option 5: Enter N to disable interprogram support.
Option 6: Enter N to disable disk emulation support.
Option 7: Enter Y to enable 3270 support.

You have customized the gateway PC as a 3270 communications gateway PC.

After completing the instructions presented above, you should start the Communications Facility stations and IPL the gateway PC. Now, you can load the IBM PC Network SNA 3270 Emulation Program, PSCPG.

Use the communications profile option of the PSCPG to specify the following parameters for the gateway PC:

- Gateway with network station
- Enable 3270-PC file transfer
- Printer device name—LPT1
- Gateway session name (required for printer support)—GATEWAY01
- Maximum number of sessions—8
- Names of each network station:
  - GATEWAY01
  - USER1
  - USER2

After storing these changes, choose the communicate option of the PSCPG program to establish communication.

User PC

On each user PC, PC12 and PC13, load the PSCPG program. Use the communications profile option of PSCPG to specify the following parameters for the user PCs:

- User PC network station
- Enable 3270-PC file transfer
- Printer device name—LPT1
- Names of each network station:
  - USER1 for PC12
  - USER2 for PC13.

After storing these changes, choose the communicate option of the PSCPG program to establish communication.

Enter PRINTON PR11 3 LU1 on PC12. 3 corresponds to the session number identified for the printer. A session start message should be received. Now, enter LOGON PC12 on PC12. A session start message should be received again.

Enter LOGON PC13 on PC13 to start a session. A session start message should be received.

There should be a logon prompt screen on the user PC, PC13.

EDX Secondary SNA1 and Host System Definitions

This configuration must also be defined to the EDX Secondary SNA1, to the host system access method (VTAM in this example), and to the host subsystem (CICS in this example). The sample definitions are not compatible with every SNA environment; you can use them as models to develop your customized definitions. Parameters have not been optimized, and there may be instances of values that restrict usage or degrade performance.

The examples show definitions of SNA 3274-SDLC and associated terminals to the network. No special parameters are needed in the network for the Communications Facility; you can use standard 3274 definitions. When you define your configuration, refer to the 3274 and SNA publications listed in "About This Book" on page iii and to the publications for your host system and subsystem.

Several parameters in the various definitions must agree with each other. The corresponding values defined in the figures in this section, the values defined to EDX in the EXIODEV statement (shown in the EDX definitions), and the values defined for Communications Facility SNA stations (shown on the SNA configuration worksheet) are as follows:

- **1** EDX-SNA SNAPU ADDRESS and EXIODEV ADDRESS. (This address has also been used as the PU station network address, which is recommended but not required.)
- **2** EDX-SNA SNAPU ENCODE and NCP/VTAM NRZI.
- **3** EDX-SNA SNAPU BUFSIZ must be greater than or equal to NCP/VTAM MAXDATA.
- **4** EDX-SNA SNAPU DCBNO, NCP/VTAM MAXOUT, and EXIODEV MAXDCB.
- **5** EDX-SNA SNALU number, NCP/VTAM LOCADDR, and LU station local address.
- **6** NCP VTAM MODETAB and VTAM mode table name.
- **7** NCP VTAM PU ADDR and secondary station address strapped on SDLC attachment card.
- **8** NCP/VTAM LU name and CICS NETNAME. (This name has also been used as the LU station name, and the NCP/VTAM PU name has been used as the PU station name, which is recommended but not required.)
- **9** VTAM LOGMODE name and Communications Facility SNA application mode (HOST MODE).
- **10** VTAM APPL name and Communications Facility SNA application name (HOST APPL).

EDX Secondary SNA1 Definitions

Figure 182 on page 427 shows the EDX Secondary SNA1 definitions for example 15. The definitions support four logical units using 256-byte buffers. Only two buffers per logical unit are allocated; you may want more for best performance. See the *EDX Secondary SNA1 Guide* for information about how buffers and related parameters affect performance.

```
NET1 CSECT ,               DEFINE THE NETWORK
     SNAPU RETRY=3,        NUMBER OF I/O RETRIES                 X
           ADDRESS=0A,     DEVICE ADDRESS                        X 1

           CNCTYPE=PP,     CONNECTION TYPE                       X
           NORING=NO,      INCLUDE RING SUPPORT                  X
           RATE=FULL,      MODEM SPEED                           X
           ENCODE=NRZ,     MODEM ENCODING                        X 2

           PAD=NO,         SEND PAD CHAR ON INITIAL XMISSION     X
           TODTR=0,        DATA-TERMINAL-READY TIMEOUT           X
           TODSR=0,        DATA-SET-READY TIMEOUT                X
           TOCTS=0,        CLEAR-TO-SEND TIMEOUT                 X
           TOHLA=0,        HOLD-LINE-ACTIVE TIMEOUT              X
           THRESH=4,       MINIMUM BUFFS TO SEND RECEIVE-READY   X
           BUFSIZ=256,     SIZE OF LARGEST USER MESSAGE          X 3

           BUFNO=4,        NUMBER OF BUFFERS AT ACTIVATION       X
           DCBNO=7,        MAXIMUM FRAMES TO SEND OR RECEIVE     X 4

           STAXID=S00000 ID TO SEND IN RESPONSE TO XID CMD
*
     SNALU LU=1,           LOGICAL UNIT NUMBER                   X 5

           SENDBUF=1,      NUMBER OF SEND BUFFERS FOR LU         X
           RECVBUF=1,      NUMBER OF RECEIVE BUFFERS FOR LU      X
           CTE=1,          NUMBER OF CORRELATION TABLE ENTRIES   X
           END=NO
*
     SNALU LU=2,           LOGICAL UNIT NUMBER                   X
           SENDBUF=1,      NUMBER OF SEND BUFFERS FOR LU         X
           RECVBUF=1,      NUMBER OF RECEIVE BUFFERS FOR LU      X
           CTE=1,          NUMBER OF CORRELATION TABLE ENTRIES   X
           END=NO
*
     SNALU LU=3,           LOGICAL UNIT NUMBER                   X
           SENDBUF=1,      NUMBER OF SEND BUFFERS FOR LU         X
           RECVBUF=1,      NUMBER OF RECEIVE BUFFERS FOR LU      X
           CTE=1,          NUMBER OF CORRELATION TABLE ENTRIES   X
           END=NO
*
     SNALU LU=4,           LOGICAL UNIT NUMBER                   X
           SENDBUF=1,      NUMBER OF SEND BUFFERS FOR LU         X
           RECVBUF=1,      NUMBER OF RECEIVE BUFFERS FOR LU      X
           CTE=1,          NUMBER OF CORRELATION TABLE ENTRIES   X
           END=YES         LAST LOGICAL UNIT TO GENERATE
     END
```

Figure 182. Example 15: EDX Secondary SNA1 Definitions

Host System Definitions

Figure 183 on page 428 shows the definitions of physical and logical units to the network control program (NCP) and VTAM. They are defined as a PU type 2 (cluster controller) and its associated LUs. No information about LU type is specified to the NCP. VTAM unformatted session services table (USSTAB) specifications are not included, because the Communications Facility can't use this VTAM service. The SNA IOCP has its own log on technique that replaces the USS screen and facilities. If USSTAB is specified for your system, the Communications Facility won't use it, but it shouldn't cause any problem.

```
GCFSDLC GROUP LNCTL=SDLC,          SDLC LINE                    X
              POLLED=YES,          3274 ALWAYS POLLED           X
              NRZI=NO              NRZI                           2
*
LINE028 LINE  ADDRESS=028,         LINE ADDRESS IN 3705         X
              DISCNT=NO,           VTAM ONLY OPERAND            X
              MODETAB=CFTBL,       VTAM MODE TABLE              X 6

              DLOGMOD=LU2256,      DEF LOGON MODE FROM MODETAB  X
              SPEED=4800,          LINE SPEED (SEE NCP DOCS)    X
              RETRIES=(5,1,20),    RECOVERY ATTEMPTS & PAUSE    X
              PACING=0,            PACING NOT NORMALLY USED     X
              VPACING=0,           FOR DISPLAYS OR PRINTERS     X
              ISTATUS=ACTIVE       LINE COMES UP WITH NCP
*
        SERVICE ORDER=(P028CFS)    ONLY 1 STATION HERE
*
P028CFS PU    ADDR=01,             MATCH SDLC CARD STRAPPING    X 7

              PUTYPE=2,            CLUSTER CONTROLLER           X
              MAXDATA=256,         SEGMENT SIZE                 X 3

              PASSLIM=7,           MAX CONSEC RU'S TO THIS PU   X
              MAXOUT=6             MAX FRAMES BEFORE RESPONSE     4
*
LCFS01  LU    LOCADDR=1  5  8

LCFS02  LU    LOCADDR=2
LCFS03  LU    LOCADDR=3
LCFS04  LU    LOCADDR=4
```

Figure 183. Example 15: Definitions to NCP and VTAM

Figure 184 on page 429 shows the definition of a VTAM mode table. The SNA application modes used with the Communications Facility correspond to SNA session parameters that are defined in SNA bind commands. A given mode name refers to an image of a specific bind command stored on the host. You must define all modes used with the Communications Facility. The sample mode table includes entries for LU types 1, 2, and 3, as suggested in the *3274 Guide*. The restriction the Communications Facility places on mode values is that only 24 × 80 screen size is supported for both default and alternate screens. For subtype 3286 and 3287 LU3 stations, and for subtype 3278 and 3279 LU2 devices, the Communications Facility will accept the bind, and assume the real device can handle the alternate size. See the *3274 Guide* for details of valid bind images.

Figure 185 on page 429 shows the definition of application programs to VTAM. You must define all SNA application names used with the Communications Facility.

```
CFTBL    MODETAB                                                  6

*                  STANDARD 3274 LU1 FROM 3274 PUBLICATION
STD32741 MODEENT LOGMODE=STD32741,                                X 9

               FMPROF=X'03',                                      X
               TSPROF=X'03',                                      X
               PRIPROT=X'B1',                                     X
               SECPROT=X'90',                                     X
               COMPROT=X'3080',                                   X
               RUSIZES=X'8587',                                   X
               PSERVIC=X'010000000000000000000000'
*                  STANDARD 3274 LU3 FROM 3274 PUBLICATION
STD32743 MODEENT LOGMODE=STD32743,                                X
               FMPROF=X'03',                                      X
               TSPROF=X'03',                                      X
               PRIPROT=X'B1',                                     X
               SECPROT=X'90',                                     X
               COMPROT=X'3080',                                   X
               RUSIZES=X'8587',                                   X
               PSERVIC=X'030000000000000000000000'
*                   LU2 WITH 256-BYTE RU SIZE
LU2256   MODEENT LOGMODE=LU2256,                                  X
               FMPROF=X'03',                                      X
               TSPROF=X'03',                                      X
               PRIPROT=X'B1',                                     X
               SECPROT=X'90',                                     X
               COMPROT=X'3080',                                   X
               RUSIZES=X'8585',                                   X
               PSERVIC=X'020000000000000000000200'
         MODEEND
         END
```

Figure 184. Example 15: VTAM Mode Table

```
IMS      APPL  AUTH=(ACQ,PASS)
*
CICS     APPL  AUTH=(ACQ,VPACE)         10

*
TSO      APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=10
TSO0001  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0002  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0003  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0004  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0005  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0006  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0007  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0008  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0009  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
TSO0010  APPL  AUTH=(NOBLOCK,PASS,NOPO,TSO,NVPACE),EAS=1
```

Figure 185. Example 15: Definition of Applications to VTAM

Figure 186 on page 430 shows a CICS terminal control table with entries for the three type 2 logical units and the type 3 logical unit used in this sample configuration.

```
LC01      DFHTCT TYPE=TERMINAL,                                    X
                 TRMTYPE=LUTYPE2,          TERMINAL                X
                 TRMMODL=2,                                        X
                 TRMIDNT=LC01,                                     X
                 BUFFER=0,                                         X
                 RUSIZE=256,                                       X
                 TIOAL=(256,2048),                                 X
                 NETNAME=LCFS01,                                   X 8

                 CHNASSY=YES,                                      X
                 BRACKET=YES,                                      X
                 TRMSTAT=(TRANSCEIVE),                             X
                 ACCMETH=VTAM,                                     X
                 GMMSG=YES,                                        X
                 RELREQ=(YES,YES)
*
LC02      DFHTCT TYPE=TERMINAL,                                    X
                 TRMTYPE=LUTYPE2,          TERMINAL                X
                 TRMMODL=2,                                        X
                 TRMIDNT=LC02,                                     X
                 BUFFER=0,                                         X
                 RUSIZE=256,                                       X
                 TIOAL=(256,2048),                                 X
                 NETNAME=LCFS02,                                   X
                 CHNASSY=YES,                                      X
                 BRACKET=YES,                                      X
                 TRMSTAT=(TRANSCEIVE),                             X
                 ACCMETH=VTAM,                                     X
                 GMMSG=YES,                                        X
                 RELREQ=(YES,YES)
*
*
LC03      DFHTCT TYPE=TERMINAL,                                    X
                 TRMTYPE=LUTYPE3,          PRINTER                 X
                 TRMMODL=2,                                        X
                 TRMIDNT=LC03,                                     X
                 BUFFER=0,                                         X
                 NETNAME=LCFS03,                                   X
                 TRMSTAT=(TRANSCEIVE),                             X
                 ACCMETH=VTAM
```

Figure 186. Example 15: CICS Terminal Control Table

Example 16: PC Connect with Inter-LAN Services

Figure 187 shows a configuration in which two Series/1s are connected to a Local Communications Controller. A PC LAN is also connected to each Series/1 through a Series/1-PC Connect Attachment.

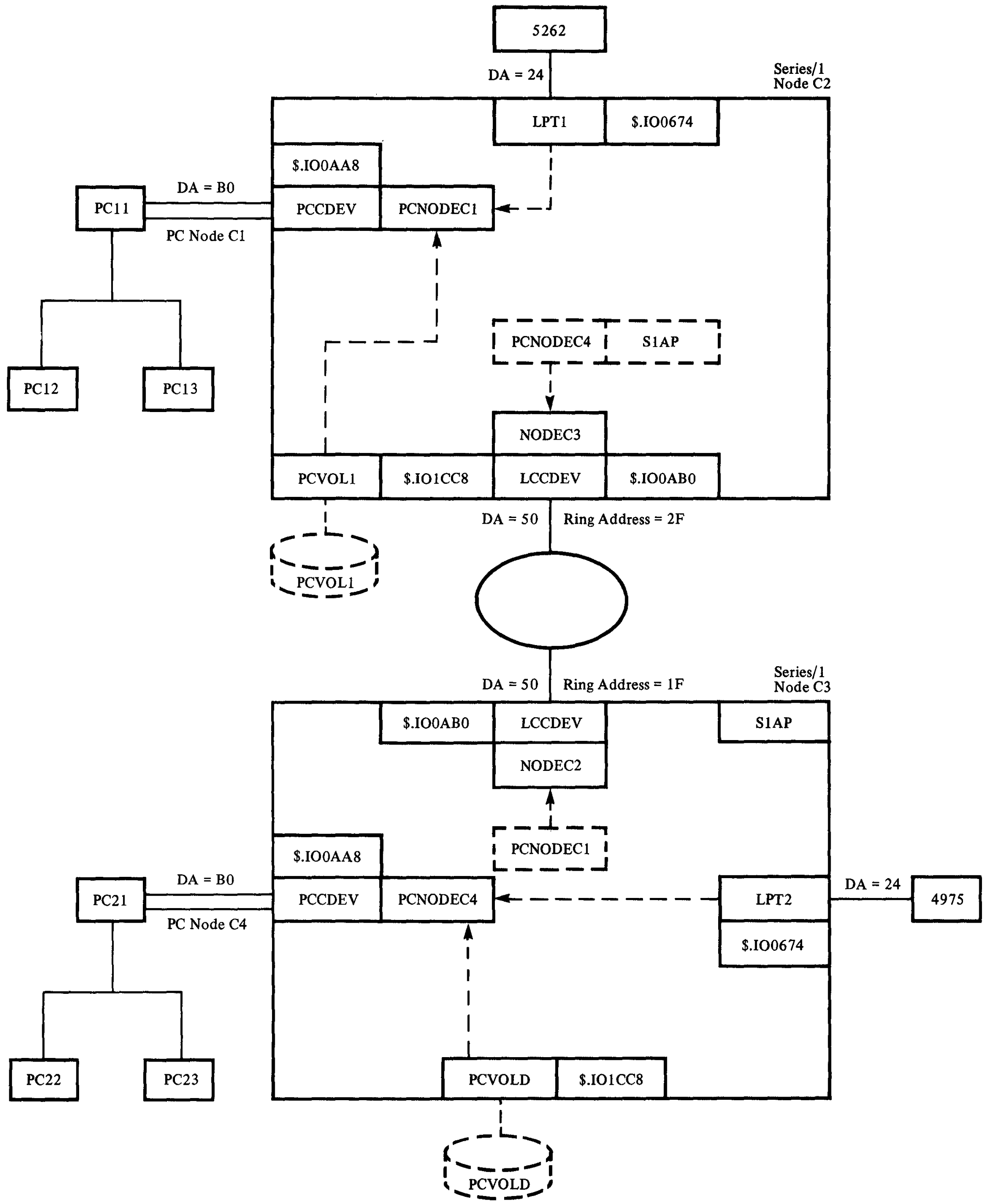


Figure 187. Example 16: PC Connect with Inter-LAN Services

Using this configuration, a PC user can:

- Use Series/1 printers as LAN-shared printers
- Use Series/1 data sets as LAN-shared disk devices
- Issue inter-LAN messages
- Write a PC program to communicate with a PC program on another LAN
- Write a PC program to communicate with a Series/1 program
- Write a PC program to issue user error logs.

This example does not include communication with a host system. You could attach another PC, running the IBM PC Network SNA 3270 Emulation Program, to the Series/1 through a Series/1-PC Connect Attachment to provide communication with the host.

EDX Definitions

Figure 188 shows the EDX definitions for example 16. In this example partition numbers 1 and 3 are not mapped with COMMON.

```
*** EXAMPLE 16
***
$EDXDEF   CSECT
          SYSTEM   STORAGE=512,MAXPROG=(10,10,10,10,10,10,10,10),     X
               PARTS=(32,32,32,32,32,32,32,32),COMMON=(CSXEND,        X
               CSXEND,0,CSXEND,CSXEND,0,CSXEND,CSXEND),               X
               INITMOD=(CSPCCIPL,CSXINIT)
            •
            •
            •
$SYSPRTR  TERMINAL DEVICE=5262,ADDRESS=24,END=YES
            •
            •
            •
          LCC      ADDRESS=50
          LCC      ADDRESS=51
          LCC      ADDRESS=52
          PCC      ADDRESS=B0
          PCC      ADDRESS=B1
          PCC      ADDRESS=B2,END=YES
          END
```

Figure 188. Example 16: EDX Definitions

Station Definitions

Figure 189 through Figure 194 on page 439 show the station definitions for the two Series/1 nodes in this configuration.

User/Message/Alias/Volume Configuration Worksheet

System: Example 16: PC Connect with Inter-LAN Services
Node Assignment: C2

Description	Name	NA	Type	Alias[1]
Printer IOCP	$.IO0674	C2AA	USER	
LCC IOCP	$.IO0AB0	C2AB	USER	
PC Connect disk-server IOCP	$.IO1CC8	C2AC	USER	
PC Connect IOCP	$.IO0AA8	C2AD	USER	
Application Remote Station	S1AP	C3A0	USER	
PC LAN C1 Volume Station	PCVOL1	C2AE	VOLUME	

[1] Specify only for alias station

S$POOL Storage

4 User/Msg Stations	× 88 = 352 bytes
_____Alias Stations	× 40 = _____ bytes
1 Remote Station	× 32 = 32 bytes
1 Volume Station	× 104 = 104 bytes

Figure 189. Example 16: User/Message/Alias/Volume Configuration Worksheet

User/Message/Alias/Volume Configuration Worksheet

System: Example 16: PC Connect with Inter-LAN Services
Node Assignment: C3

Description	Name	NA	Type	Alias[1]
Printer IOCP	$.IO0674	C3AA	USER	
LCC IOCP	$.IO0AB0	C3AB	USER	
PC Connect disk-server IOCP	$.IO1CC8	C3AC	USER	
PC Connect IOCP	$.IO0AA8	C3AD	USER	
Application	S1AP	C3A0	USER	
PC LAN C4 Volume Station	PCVOLD	C3AE	VOLUME	

[1] Specify only for alias station

S$POOL Storage

5 User/Msg Stations	× 88 = 440 bytes
_____Alias Stations	× 40 = _____ bytes
_____Remote Stations	× 32 = _____ bytes
1 Volume Station	× 104 = 104 bytes

Figure 190. Example 16: User/Message/Alias/Volume Configuration Worksheet

Device Configuration Worksheet

System: Example 16: PC Connect with Inter-LAN Services
Node Assignment: C3

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
Printer	LPT2	C324	DEVICE	Printer	2500	N	4040	0000		

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO.

S$POOL Storage

_____3101F Devices	× 160 = _____ bytes
1 4978/Printer Device	× 264 = 264 bytes
_____3101 Devices	× 304 = _____ bytes
_____Remote Stations	× 32 = _____ bytes

Figure 191. Example 16: Device Configuration Worksheet

Device Configuration Worksheet

System: Example 16: PC Connect with Inter-LAN Services
Node Assignment: C2

Description	Name	NA	Type	Device Type	Buffer Size	PDS[1]	Polling CUDA	Selection CUDA[2]/ HDCPY[3]	Bit Rate[4]	Auto Ans[4]
Printer	LPT1	C224	DEVICE	Printer	2500	N	4040	0000		

[1] Specify only for printer device station
[2] Specify only for printer or 4978 device station
[3] Specify only for 3101 or 3101F device station
[4] Specify only for 3101F device station
For 7485s, bit rate must be 9600 and AUTO ANS=NO.

S$POOL Storage

______3101F Devices	× 160 = ______ bytes
1 4978/Printer Device	× 264 = 264 bytes
______3101 Devices	× 304 = ______ bytes
______Remote Stations	× 32 = ______ bytes

Figure 192. Example 16: Device Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 16: PC Connect with Inter-LAN Services
Node Assignment: C2

Line Definitions

Description	Name	NA	Type	Line Type	Buffer Size	Device Address[1]	Packet Size[5]	Window[5]
Series/1-PC Connect Attachment	PCCDEV	C2C0	LINE	PCC	4096	B0		
LCC	LCCDEV	C2C1	LINE	LCC	4096	50		

Terminal and Node Definitions

Description	Name	NA	Type	Term/ Node Type	Ring Address[2]	Line Name	Volume Name[6]	Polling CUDA[3]/ Port#[4]	Select CUDA[3]
PC Node C1	PCNODEC1	C100	NODE	PC		PCCDEV	PCVOL1		
PC Node C4	PCNODEC4	C400	NODE	PC					
S/1 Node C3	NODEC3	C300	NODE	CF	1F	LCCDEV			

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

____HDLC Lines	× 112 = _____ bytes
2 Other Lines	× 264 = 528 bytes
2 Terminals/Nodes	× 88 = 176 bytes
1 Remote Station	× 32 = 32 bytes

Figure 193. Example 16: Line/Terminal/Node Configuration Worksheet

Line/Terminal/Node Configuration Worksheet

System: Example 16: PC Connect with Inter-LAN Services
Node Assignment: C3

Line Definitions

Description	**Name**	**NA**	**Type**	**Line Type**	**Buffer Size**	**Device Address**[1]	**Packet Size**[5]	**Window**[5]
Series/1-PC Connect Attachment	PCCDEV	C3C0	LINE	PCC	4096	B0		
LCC	LCCDEV	C3C1	LINE	LCC	4096	50		

Terminal and Node Definitions

Description	**Name**	**NA**	**Type**	**Term/ Node Type**	**Ring Address**[2]	**Line Name**	**Volume Name**[6]	**Polling CUDA**[3]**/ Port#**[4]	**Select CUDA**[3]
PC Node C4	PCNODEC4	C400	NODE	PC		PCCDEV	PCVOLD		
PC Node C1	PCNODEC1	C100	NODE	PC					
S/1 Node C2	NODEC2	C200	NODE	CF	2F	LCCDEV			

[1] Specify only for Local Communications Controller or Series/1-PC Connect Attachment line stations
[2] Specify only for Local Communications Controller node station
[3] Specify only for 3271, 3277, or 3286 terminal station
[4] Specify only for port terminal station
[5] Specify only for DCE or DTE line station
[6] Specify only for PC node station

S$POOL Storage

____HDLC Lines	× 112 = _____ bytes
2 Other Lines	× 264 = 528 bytes
2 Terminals/Nodes	× 88 = 176 bytes
1 Remote Station	× 32 = 32 bytes

Figure 194. Example 16: Line/Terminal/Node Configuration Worksheet

Notice the terminal type of the SNA logical units in Figure 191 on page 436.

Station Links

In node C2, you must directly link:

- LPT1 to PCNODEC1
- PCVOL1 to PCNODEC1
- PCNODEC4 to NODEC3.

In node C3, you must directly link:

- LPT2 to PCNODEC4
- PCVOLD to PCNODEC4
- PCNODEC1 to NODEC2.

Disk-Server Initialization

You must allocate a volume for each volume station defined to the system. This volume must have the same name as the volume station you have defined. Use the EDX utility, $INITDSK, to allocate a volume called PCVOL1 in node C2 and PCVOLD in node C3.

You can now use the Communications Facility program, $.DSFORM, to create data sets to be used as emulated PC disks. You must create these data sets starting with the letter following the last physical disk on the gateway PC. Assuming each gateway PC has a hard disk, C, use $.DSFORM to create drives D, E, and F on PCVOL1 and PCVOLD. These drives must be in an alphabetic order, else the PC Connect disk-server IOCP, $.IO1CC8, will not recognize them.

PC Installation and Setup

This section describes the installation and setup procedures for each PC in this example. The two gateway PCs should be installed and setup before the other PCs in the network.

Gateway PC

See the *Series/1-PC Connect for the Event Driven Executive Communications Facility* for information about installing Series/1-PC Connect Attachment on the two gateway PCs, PC11 and PC21.

After completing this procedure, use the Series/1-PC Connect customization program, S1PCCUST.BAS, to specify the following options for PC11:

Option 1: Enter DD to add another LAN to LAN names list.
Option 3: Enter LPT1 to add the Series/1 printer to the emulated printer names list.
Option 7: Enter AA to specify the gateway PC's LAN name.

Using the same customization program, specify the following options for PC21:

Option 1: Enter AA to add another LAN to LAN names list.
Option 3: Enter LPT2 to add the Series/1 printer to the emulated printer names list.
Option 7: Enter DD to specify the gateway PC's LAN name.

After customizing the two gateway PCs, you can start the Series/1 stations. You must start the volume stations before you start the PC node stations. IPL the gateway PC.

Use the IBM PC Network Program to customize each PC in the LAN. Your AUTOEXEC.BAT files should look like the following examples:

1 PC11

```
&:.***  NETWORK BATCH FILE
&:.***
ECHO OFF
PATH C:\NETWORK
YNPROMPT  Y  N  39  Do you want to start the network (Y/N)?
IF ERRORLEVEL 1 GOTO A
ECHO ON
NET START SRV PC11 /SRV:5 /USN:4 /MBI:8192 /PRB:16384 /SHR:29 /SES:29 /CMD:25
NET SHARE DSKD=D:\
NET SHARE DSKE=E:\
NET SHARE DSKF=F:\
NET SHARE KLPT1=LPT1
&:.A
IBMS1PC
DATE
TIME
```

2 PC21

```
&:.***  NETWORK BATCH FILE
&:.***
ECHO OFF
PATH C:\NETWORK
YNPROMPT  Y  N  39  Do you want to start the network (Y/N)?
IF ERRORLEVEL 1 GOTO A
ECHO ON
NET START SRV PC21 /SRV:5 /USN:4 /MBI:8192 /PRB:16384 /SHR:29 /SES:29 /CMD:25
NET SHARE DSKD=D:\
NET SHARE DSKE=E:\
NET SHARE DSKF=F:\
NET SHARE KLPT2=LPT2
&:.A
IBMS1PC
DATE
TIME
```

3 PC12

```
&:.***  NETWORK BATCH FILE
&:.***
ECHO OFF
PATH A:\;B:\
YNPROMPT  Y  N  39  Do you want to start the network (Y/N)?
IF ERRORLEVEL 1 GOTO A
ECHO ON
NET START MSG PC12 /ASG:29 /MBI:4096 /SES:29 /CMD:25
NET USE D:  //PC11/DSKD
NET USE E:  //PC11/DSKE
NET USE F:  //PC11/DSKF
NET USE LPT1:  //PC11/KLPT1
&:.A
DATE
TIME
```

4 PC13

```
&:.***  NETWORK BATCH FILE
&:.***
ECHO OFF
PATH A:\;B:\
YNPROMPT  Y  N  39  Do you want to start the network (Y/N)?
IF ERRORLEVEL 1 GOTO A
ECHO ON
NET START MSG PC13 /ASG:29 /MBI:4096 /SES:29 /CMD:25
NET USE D:  //PC11/DSKD
NET USE E:  //PC11/DSKE
NET USE F:  //PC11/DSKF
NET USE LPT1:  //PC11/KLPT1
&:.A
DATE
TIME
```

5 PC22

```
&:.***  NETWORK BATCH FILE
&:.***
ECHO OFF
PATH A:\;B:\
YNPROMPT  Y  N  39  Do you want to start the network (Y/N)?
IF ERRORLEVEL 1 GOTO A
ECHO ON
NET START MSG PC22 /ASG:29 /MBI:4096 /SES:29 /CMD:25
NET USE D:  //PC21/DSKD
NET USE E:  //PC21/DSKE
NET USE F:  //PC21/DSKF
NET USE LPT2:  //PC21/KLPT2
&:.A
DATE
TIME
```

6 PC23

```
&:.***  NETWORK BATCH FILE
&:.***
ECHO OFF
PATH A:\;B:\
YNPROMPT  Y  N  39  Do you want to start the network (Y/N)?
IF ERRORLEVEL 1 GOTO A
ECHO ON
NET START MSG PC23 /ASG:29 /MBI:4096 /SES:29 /CMD:25
NET USE D:  //PC21/DSKD
NET USE E:  //PC21/DSKE
NET USE F:  //PC21/DSKF
NET USE LPT2:  //PC21/KLPT2
&:.A
DATE
TIME
```

After all the PCs have completed the AUTOEXEC.BAT procedures, issue the following commands on PC22:

COPY AUTOEXEC.BAT E&:.
NET PRINT E&:.AUTOEXEC.BAT LPT2
NET SEND MSGGATE AA.PC12 *message text*

Issue the following commands on PC13:

COPY AUTOEXEC.BAT F&:.
NET PRINT F&:.AUTOEXEC.BAT LPT1
NET SEND MSGGATE DD.PC23 *message text*

These commands copy a file to an emulated disk. The file is then printed on an emulated printer from the emulated disk, and a message is sent to a user on another LAN.

Maintaining the Communications Facility

This chapter explains how to assemble and link-edit Communications Facility programs.

The name of each source module begins with 'S$'. The prolog of each module indicates whether you should use the EDL compiler ($EDXASM) or the Series/1 macro assembler ($S1ASM) to assemble it.

Some of the Communications Facility utility programs include storage-resident message data sets. The name of each source message data set begins with 'S$M'. Use the EDX message utility, $MSGUT1, to convert a source message data set to a storage resident module. Where the source name is S$M*xxxxx*, the storage resident module name must be O$M*xxxxx*.

For information about the various assemblers and linkage editors, and the message utility see the *EDX Operator Commands and Utilities Reference*.

There are four categories of Communications Facility programs: supervisor modules, assembler overlay programs, multimodule programs, and single-module programs.

Supervisor Modules

The supervisor modules add support for the Communications Facility to the EDX supervisor. After you have assembled one of these modules, include the object module in a link-edit of the EDX supervisor. The source and object names of the supervisor modules are:

Source Module	**Object Module**
S$CSA	CSA
S$CSF	CSF
S$CSM	CSM
S$CSPDIO	CSPDIO
S$CSX	CSX
S$CSXSYS	CSXSYS
S$INIT	CSXINIT
S$PCCINT	CSPCCINT
S$PCCIPL	CSPCCIPL
S$PCCOPN	CSPCCOPN

Source Module	Object Module
S$PCCPTC	CSPCCPTC
S$RDINIT	CSXRDINT

Assembler Overlay Programs

The assembler overlay programs assemble Communications Facility language extensions for the $EDXASM assembler. After you have assembled one of these modules, use $UPDATE or $EDXLINK to prepare the load module. The source and load module names of assembler overlay programs are:

Source Module	Load Module
S$CFID	A.CFID
S$CFTERM	A.CFTERM
S$CSXDEF	A.CSXDEF
S$GETPUT	A.GETPUT
S$SENREC	A.SENREC

Multimodule Programs

Some Communications Facility programs consist of more than one source module. After you have assembled the modules for a multimodule program, use $EDXLINK to prepare the load module. This section shows the components of each of these programs. The names given are source module names, unless noted otherwise. Some components are distributed in both source and object form; where the source module name is S$*xxxxxx*, the object module name is O$*xxxxxx*.

Load Module	Source or Object Module
$.CFD	S$CF S$DKRD S$DKWR
$.CONFIG	S$CONFIG S$MOPBOG[23] S$MOPFIG[23] S$MOPOPM[23]
$.CPCMCF	S$CPCMCF S$CM1MSG

Load Module	Source or Object Module
$.CPDEF	S$CPDEF
	S$CPDOV1[17]
	S$CPDOV2[17]
	S$CPDOV3[17]
	S$CPDOV4[17]
	S$CPDOV5[17]
$.CPF	S$CPF
	S$CPFOV1[17]
	S$CPFOV2[17]
	S$CPFOV3[17]
	S$CPFOV4[17]
$.CPHELP	S$CPHELP
	S$CPHLP1[17]
	S$CPHLP2[17]
	S$CPHLP3[17]
$.DSFORM	S$DSFORM
	S$DSKUT
	$EDXATSR[18]
	$$RETURN[18]
	$$SVC[18]
$.IO0AA8	S$IO0AA8
	$EDXATSR[18]
	$$DEVLOG[18]
	$$RETURN[18]
	$$SVC[18]
$.IO0AB0	S$IO0AB0
	$EDXATSR[18]
	$$RETURN[18]
$.IO0AB8	S$IO0AB8[19]
	S$IO0AB9[19]
	S$IO0ABA[19]
	S$IO0ABB[19]
	S$IO0ABC[19]
	S$IO0ABD[19]
	S$IO0ABE[19]
	S$IO0ABF[19]
	$EDXATSR[18]
	$$SVC[18]
	$$RETURN[18]
	$XHCUSR[24]

Load Module	Source or Object Module
$.IO06F0	S$IO06F0[19]
	S$IO06F1[19]
	S$IO06F2[19]
	S$IO06F3[19]
	S$IO06F5[19]
	S$IO06FT[19]
	S$IO0634
	$EDXATSR[18]
	$$SVC[18]
	$DEVLOG[18]
	$$RETURN[18]
$.IO0670	S$IO0670
	S$IO0679
$.IO0674	S$IO0674
	S$IO0675
	S$IO0676[22]
	S$IO0677[22]
$.IO0680	S$IO0680[19]
	S$IO0682[19]
	S$IO0683[19]
	S$IO0685[19]
	S$IO06F1[19]
	S$IO068T[19]
	S$IO0634
	$EDXATSR[18]
	$$SVC[18]
	$DEVLOG[18]
	$$RETURN[18]
$.IO14E8	S$IO14E8
	$NETCMD[20]
	$$SVC[18]
	$$RETURN[18]
	$NETPACT[18 25]
$.MIGR20	S$MIGR20
	S$MMGR20[23]
	S$MOPOPM[23]
$.OPAID	S$OPAID
	S$MOPAID[23]
	S$MOPBOG[23]

Load Module	Source or Object Module
$.OPCH	S$OPCH S$OPCHX S$MOPCH[23] S$MOPBOG[23]
$.OPCOPY	S$OPCOPY S$OPNAQ S$MOPCPY[23] S$MOPNAQ[23] S$MOPBOG[23]
$.OPDEF	S$OPDEF S$OPNAQ S$OPCIRC S$OPNODE S$OPLINE S$MOPDEF[23] S$MOPBOG[23] S$MOPNAQ[23]
$.OPDISP	S$OPDISP S$MOPDI[23] S$MOPBOG[23]
$.OPEDIT	S$OPEDIT S$MOPED[23] S$MOPBOG[23]
$.OPHELP	S$OPHELP S$MOPHLP[23]
$.OPLINK	S$OPLINK S$MOPLIN[23] S$MOPBOG[23]
$.OPLIST	S$OPLIST S$MOPLIS[23] S$MOPBOG[23]
$.OPSETX	S$OPSETX S$MOPSET[23] S$MOPBOG[23]
$.PANEL	S$PANEL S$PNL001 S$PNL002

Load Module	Source or Object Module
$.PDIPL	S$PDIPL
	S$D2IPL
	S$D3IPL
	S$D4IPL
	S$D5IPL
	S$D6IPL
	S$D7IPL
	S$D8IPL
	S$L0IPL
	S$B1IPL
	S$B9IPL
$.PNLPRT	S$PNLPRT
	S$PNLPR1
$.PNLUT1	S$PNLUT1
	S$PNLPR1
	S$MPNUT1[23]
$.WSC	S$WSC
	S$WSCA
$.WSCUT1[21]	S$WSCUT1
	S$WSCUTA

---

17 Link as an overlay segment

18 Object module supplied with EDX

19 Both source and object modules supplied with the Communications Facility

20 Object module supplied with EDX-SNA

21 Specify AUTOCALL $AUTO,ASMLIB to include $IMAGE subroutines

22 Include for LU-1 support; both source and object modules supplied with the Communications Facility

23 Source message data set

24 Object module supplied with EDX XHCS

25 For EDX Secondary SNA2 only.

Single-Module Programs

The rest of the Communications Facility programs consist of a single source module. After you have assembled one of these modules, use $UPDATE or $EDXLINK to prepare the load modules. Where the source module name is S$*xxxxxx*, the load module name is $.*xxxxxx*.

Appendix A. Configuring and Connecting 3101 Display Terminals

The IBM 3101 display terminals are supported under the Communications Facility in character mode only. This appendix contains information to aid you in planning the link between the 3101 IOCP, $.IO06F0, and the 3101 display terminals.

Connecting the 3101 to $.IO06F0

The 3101 is connected to $.IO06F0 by the attachment features supported by the Communications Facility. For each attachment and type of interface, you must set the hardware switches on the 3101, physically jumper the attachment card, connect the cables, and define the 3101F device station.

Figure 195 shows the Series/1 attachment cards that can be used to connect the 3101 to $.IO06F0.

Attachment Card	Feature	Feature Number
MFA	Multifunction Attachment	1310
FPCA	Feature-Programmable Multiline Communications	2095/2096
FPCA-RPQ	Asynchronous Direct 8-Line RS-422A Adapter Custom	2095/RPQ D02350
Terminal/Host Adapter	Series/1 System Unit Terminal/Host Adapter	3629

Figure 195. 3101 Attachment Features

There are three types of electrical interfaces by which a 3101 can be connected to a Series/1:

TTY
: 20 milliampere current loop interface

RS-422A
: EIA RS-422A interface standard

EIA
: EIA RS-232C/CCITT V.24 interface standard

Local 3101 Connections

Current Loop Interface

Figure 196 shows the 3101 terminal with the FPCA adapter's current loop interface.

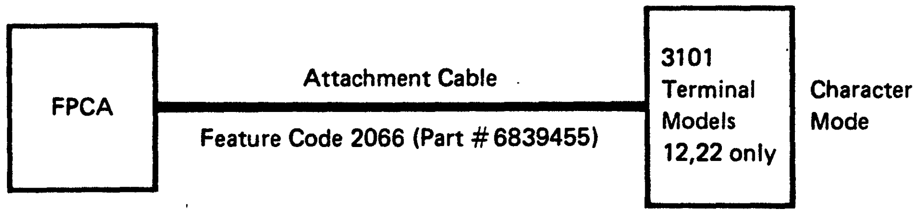


Figure 196. Current Loop Interface with FPCA

3101 Switch Settings: For a current loop interface through the FPCA adapter, you may set the 3101 setup switches as follows (ON=1, OFF=0):

```
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
0 0 0 1 0 0 1 0  0 0 0 1 0 0 1 1  1 0 1 0 0 0 0 0  1 0 0 1 1 0 0 1
```

These 3101 switch settings indicate a speed of 9600 bps.

Defining 3101F Device Station: Use the CP DEF DEVICE 3101F command, with the following options, to create a station definition in $.SYSNET:

```
CP DEF station-name na DEV 3101F buffsize poll hdcopy bit-rate OFF
```

Attachment Options: You may set the selection jumpers for the FPCA card as follows:

```
RTS    ON
DTR    ON
DCD    ON
TTY    INTERFACE
```

RS-422A Interface with FPCA-RPQ

Figure 197 shows the 3101 terminal with the FPCA-RPQ adapter.

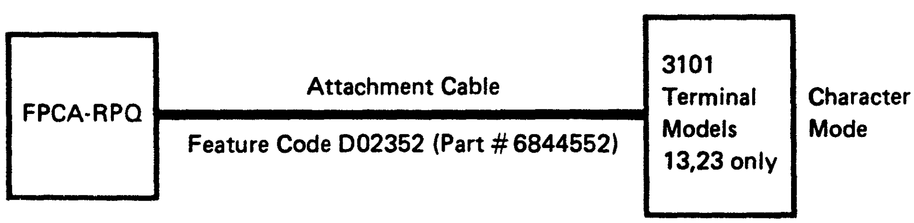


Figure 197. RS-422A Interface with FPCA-RPQ

3101 Switch Settings: For an RS-422A interface through the FPCA-RPQ adapter, you may set the 3101 setup switches as follows (ON=1, OFF=0):

```
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
0 0 0 1 0 0 1 0  0 0 0 1 0 0 1 1  1 0 1 0 0 0 0 0  1 0 0 1 1 0 0 1
```

These 3101 switch settings indicate a speed of 9600 bps.

Defining 3101F Device Station: Use the CP DEF DEVICE 3101F command, with the following options, to create a station definition in $.SYSNET:

```
CP DEF station-name na DEV 3101F buffsize poll hdcopy bit-rate OFF
```

RSA-422A Interface with MFA

Figure 198 shows the 3101 terminal with the MFA adapter.

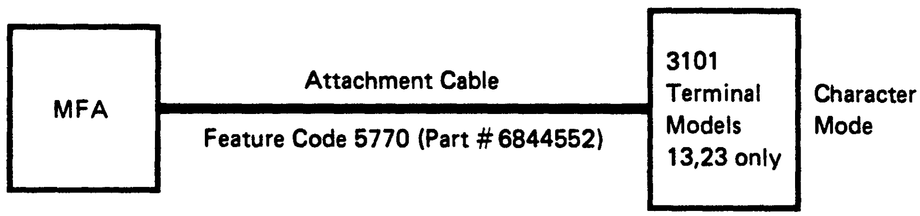


Figure 198. RS-422A Interface with the MFA

3101 Switch Settings: For an RS-422A interface through the MFA adapter, you may set the 3101 setup switches as follows (ON=1, OFF=0):

```
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
0 0 0 1 0 0 1 0  0 0 0 1 0 0 1 1  1 0 1 0 0 0 0 0  1 0 0 1 1 0 0 1
```

These 3101 switch settings indicate a speed of 9600 bps.

Defining 3101F Device Station: Use the CP DEF DEVICE 3101F command, with the following options, to create a station definition in $.SYSNET:

```
CP DEF station-name na DEV 3101F buffsize poll hdcopy bit-rate OFF
```

Attachment Options: If the 3101 terminal is not defined to EDX, you must use the CFTERM instruction to define the 3101 terminal in your EDX supervisor during system generation:

```
CFTERM   ADDRESS=addr,LMODE=RS422,PRTS=ON
```

Direct Connection

Figure 199 on page 452.4 shows the 3101 terminal with MFA, FPCA, and FPCA-RPQ adapters.

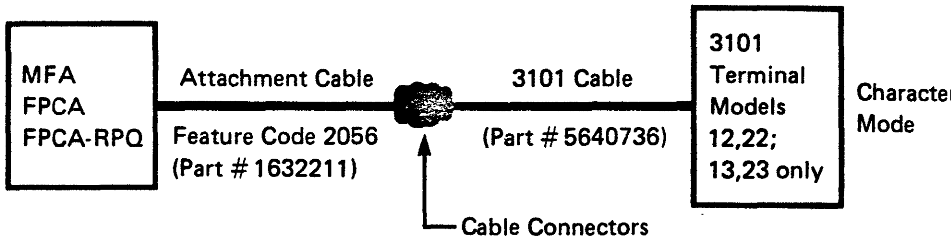


Figure 199. Direct Connection with MFA, FPCA, and FPCA-RPQ

3101 Switch Settings: For a direct connection through the MFA, FPCA, and FPCA-RPQ adapters, you may set the 3101 setup switches as follows (ON=1, OFF=0):

```
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
0 0 1 1 0 0 1 0  0 0 0 1 0 0 1 1  1 0 1 0 0 0 0 0  1 0 0 1 1 0 0 1
```

These 3101 switch settings indicate a speed of 9600 bps.

Defining 3101F Device Station: Use the CP DEF DEVICE 3101F command, with the following options, to create a station definition in $.SYSNET:

```
CP DEF station-name na DEV 3101F buffsize poll hdcopy bit-rate OFF
```

Attachment Options: You may set the selection jumpers for the FPCA card as follows:

```
RTS    ON
DTR    ON
DCD    ON
EIA    INTERFACE
```

If the 3101 terminal is not defined to EDX, and you are using the MFA card, you must use the CFTERM instruction to define the 3101 terminal in your EDX supervisor during system generation:

```
CFTERM   ADDRESS=addr,LMODE=LOCAL,PRTS=ON
```

Remote 3101 Connections

Point-to-Point Nonswitched

Figure 200 on page 452.5 shows the 3101 terminal with MFA and FPCA adapters.

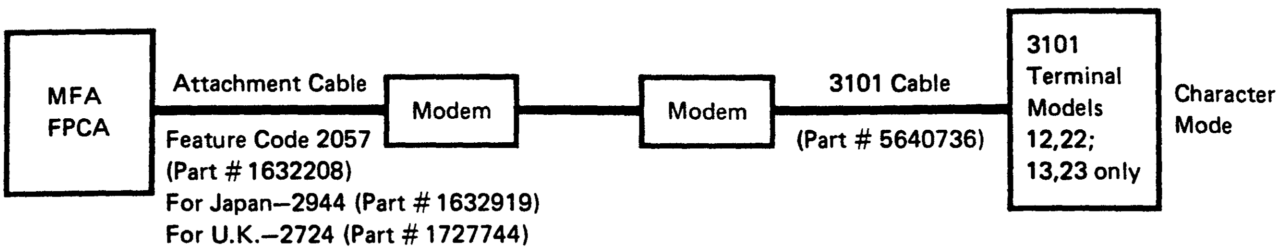


Figure 200. Point-to-Point Nonswitched MFA and FPCA

3101 Switch Settings: For a point-to-point nonswitched connection through the MFA and FPCA adapter, you may set the 3101 setup switches as follows (ON=1, OFF=0):

```
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
0 0 1 1 0 0 1 0  0 0 0 1 0 0 1 1  1 0 1 0 0 0 0 0  0 1 0 1 0 1 0 1
```

These 3101 switch settings indicate a speed of 1200 bps.

Defining 3101F Device Station: Use the CP DEF DEVICE 3101F command, with the following options, to create a station definition in $.SYSNET:

```
CP DEF station-name na DEV 3101F buffsize poll hdcopy bit-rate ON
```

Attachment Options: You may set the selection jumpers for the FPCA card as follows:

```
RTS    ON
DTR    ON
DCD    ON
EIA    INTERFACE
```

If the 3101 terminal is not defined to EDX, and you are using the MFA card, you must use the CFTERM instruction to define the 3101 terminal in your EDX supervisor during system generation:

```
CFTERM   ADDRESS=addr,LMODE=PTTOPT,PRTS=ON
```

Modem Options: The selection jumpers for the Series/1 modem are set to:

```
DTR    OFF
AA     ON
CD     OFF
```

The selection jumpers for 3101 modem are set to:

```
DTR    OFF
AA     OFF
CD     ON
```

Point-to-Point Switched

Figure 201 shows the 3101 terminal on a point-to-point switched connection with the MFA and FPCA adapters.

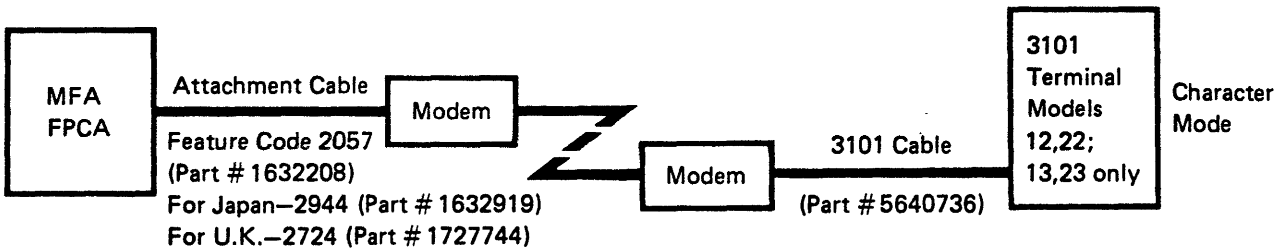


Figure 201. Point-to-Point Switched with MFA and FPCA

3101 Switch Settings: For a point-to-point switched connection through the MFA and FPCA adapters, you may set the 3101 setup switches as follows (ON=1, OFF=0):

```
1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8  1-2-3-4-5-6-7-8
0 0 1 1 0 0 1 0  0 0 0 1 0 0 1 1  1 0 1 0 0 0 0 0  0 1 0 1 0 1 0 1
```

These 3101 switch settings indicate a speed of 1200 bps.

Defining 3101F Device Station: Use the CP DEF DEVICE 3101F command, with the following options, to create a station definition in $.SYSNET:

```
CP DEF station-name na DEV 3101F buffsize poll hdcopy bit-rate ON
```

Attachment Options: You may set the selection jumpers for the FPCA card as follows:

```
RTS    ON
DTR    OFF
DCD    OFF
EIA    INTERFACE
```

If the 3101 terminal is not defined to EDX, and you are using the FPCA card, you must use the CFTERM instruction to define the 3101 terminal in your EDX supervisor during system generation:

```
CFTERM  ADDRESS=addr,LMODE=SWITCHED,PRTS=ON
```

Modem Options: The selection jumpers for the Series/1 modem are set to:

```
DTR   OFF
AA    ON
CD    OFF[26]
```

[26] You may need to set carrier detect (CD) permanently on your Series/1 modem to establish a connection between the 3101 and $.IO06F0.

The selection jumpers for 3101 modem are set to:

DTR	OFF
AA	OFF
CD	ON

Remote Dial-in Answering Sequence

Make sure that the Series/1 modem is powered on before starting the 3101 device station. Powering on the modem may cause the ring indicator (RI) to activate momentarily. $.IO06F0 interprets this ring indication as an incoming call and attempts to answer the line.

The following sequence of events shows the status of the RS-232C pins between a Series/1 and a Series/1 modem during a remote dial-in operation.

1. Before the remote 3101 dials in, ensure that request-to-send (RTS) and clear-to-send (CTS) are permanantly active on the Series/1 modem.

2. After the remote 3101 modem dials in, the Series/1 modem's ring indication (RI) is activated.

3. In response to the first or second ring, $.IO06F0 activates the data terminal ready (DTR) signal which activates the modem's auto-answer circuits.

4. When the Series/1 modem establishes a connection with the remote 3101 modem, the Series/1 modem activates data set ready (DSR). At the same time, it also activates carrier detect (CD) if supported and if the signal is of acceptable quality.

5. The 3101 modem transmits and receives data from the Series/1 or host application.

6. When the station is stopped or halted, $.IO06F0 deactivates DTR, causing the Series/1 modem to disconnect.

When using a terminal/host adapter on a 5170 Series/1 system unit, you must activate carrier detect (CD) on the Series/1 modem. Activating carrier detect causes both carrier detect and data set ready (DSR) to be permanently active. Permanent data-set-ready status prevents a time-out from occurring when $.IO06F0 turns on data terminal ready (DTR) in response to the ring indication (RI).

Glossary

This is a glossary of technical Communications Facility terms that appear in the book. Only terms unique to the Communications Facility are defined here. For definitions of Event Driven Executive terms, see the appropriate EDX book. For definitions of 3270 terms, see the *3270 Component Description* manual. For definitions of X.25/HDLC Communications Support terms, see the *X.25/HDLC Communications Support Programming and Operating Reference Manual.* For general data processing terms see *Vocabulary for Data Processing, Telecommunications, and Office Systems,* GC20-1699.

<IO>. The transaction identifier associated with the Communications Facility program $.PD<IO>.

$.CF. The name given to the control program after it is loaded under the name $.CFD or $.CFS.

$.CFD. The version of the control program for Communications Facility systems that use disk queuing of messages.

$.CFMENU. The sample application program, distributed as part of the Communications Facility, that demonstrates how to communicate with users at 3270-type terminals.

$.CFS. The version of the control program for Communications Facility systems using storage queuing of messages.

$.CONFIG. The utility program that allows the user to define and modify stations and maintain the system message data set.

$.DISP. The Communications Facility system station.

$.DSFORM. The utility program that creates, deletes, or renames a data set to be used as an emulated PC disk.

$.DSINIT. The utility program that initializes a data set for the disk queuing of messages.

$.DSPROF. The control data set that contains information about emulated PC disks on a Series/1 volume.

$.HMU. The transaction-processing program that allows an operator at a Series/1 to maintain a remote Series/1.

$.HSL. The small, efficient loader used by the program dispatcher to load transaction-processing programs.

$.IO0AA8. The input/output control program that manages communication between a Series/1 and a gateway PC connected by a Series/1-PC Connect Attachment.

$.IO0AB0. The input/output control program that manages communication between Series/1s attached to a Local Communications Controller.

$.IO0AB8. The input/output control program that manages X.25 packet level communication between a Series/1 and a DTE connected by an HDLC line, with or without an intervening X.25 packet-switching data network.

$.IO0AC0. The input/output control program that manages 3270 display stations and printers attached to the Series/1 by a multipoint BSC line.

$.IO0AD0. The input/output control program that provides 3270 emulation when a Series/1 is connected to a host processor through a channel attachment.

$.IO0AE0. The input/output control program that provides 3270 emulation when a Series/1 is connected to a host processor by a multipoint BSC line.

$.IO0A10. The input/output control program that manages communication between Series/1s connected by a point-to-point BSC line.

$.IO0A20. The input/output control program that manages communication between a Series/1 and various other systems connected by a point-to-point BSC line.

$.IO06F0. (1) The input/output control program that manages 3101 display stations attached to the Series/1 through a feature programmable communications adapter, or a multipoint attachment, or a 4950/5170-495 terminal/host adapter as if they were 3277 display stations. (2) The renamed $.IO0680 I/O control program.

$.IO0670. The input/output control program that manages 4978 display stations with RPQ D02428 and 4980 display stations as if they were 3277 display stations.

$.IO0674. The input/output control program that manages Series/1 printers as if they were 3286 printers or 3287 printers in SCS mode.

$.IO0680. The input/output control program that manages 7485 display stations attached to the Series/1 through a feature programmable communications adapter, or a multifunction attachment, or a 4950/5170-495 terminal/host adapter as if they were 3277 display stations.

$.IO1CC8. The input/output control program that manages PC read, write, and directory requests to Series/1 disks.

$.IO14E8. The input/output control program that provides 3270 emulation when a Series/1 is connected to a host processor that uses SNA.

$.MENU. The data set containing the image used by $.WSMENU, in the format created by the EDX utility $IMAGE.

$.PD. The program dispatcher; the program that manages the processing of transactions.

$.PD<IO>. The transaction-processing program that performs disk I/O in response to remote disk requests.

$.PDBSTS. The utility program that allows a user to gain exclusive control of a disk volume directory in a remote Series/1 before loading a program that will update that directory.

$.PDIPL. The transaction-processing program that sends an IPL bootstrap to a remote Series/1, causing that Series/1 to IPL itself.

$.PDSCHD. The transaction-processing program that schedules another transaction for processing at a specific time.

$.PNLUT1. The utility program that prints the descriptions of panels created by the $.PANEL program.

$.RMU. The transaction-processing program that processes transactions received from a host management utility—$.HMU or Communications Facility/Host.

$.RPQCS0. The data set containing the control store for the data stream RPQ D02428.

$.RPQIS0. The data set containing the image store for the data stream RPQ D02428.

$.SETSTG. The utility program that changes the size of the message buffer pool and file control block pool for $.CFD.

$.SYSIPL. The data set containing CP commands that are to be executed when the Communications Facility is loaded.

$.SYSLCC. The data set containing IPL transactions, used to IPL remote nodes on a Local Communications Controller ring.

$.SYSMSG. The data set containing the text of error messages and informational messages issued by the Communications Facility and, optionally, by user programs.

$.SYSNET. The data set containing the definitions of all the stations in a node and of remote stations that will be communicated with from that node.

$.SYSPD. The data set containing CP commands, path definitions, transaction definitions, and transactions that are to be processed when the program dispatcher is started.

$.SYSPNL. The data set containing panels displayed by the $.PANEL, $.CFMENU, and $.IO14E8 programs.

$.SYSPRT. The data set containing image and control store loads for 4978 terminals and members used to support printers as buffered devices.

$.SYSX25. The data set containing two-digit call IDs and their associated X.25 network addresses used during call establishment for switched virtual circuits. As shipped, it contains 10 records of /*.

$.UT1. The utility program that allows access to various Communications Facility functions for diagnostic purposes.

$.UT2. The utility program that allows the user to examine and purge messages on the $.WASTE queue or any other disk queue.

$.WASTE. The station to which undeliverable messages are sent.

$.WSC. The work session controller; the part of the Communications Facility that allows an application program to communicate with multiple EDX devices attached to any Series/1 in the network.

$.WSCIMG. The data set containing images that can be displayed through the work session controller and members used to save data for transaction-processing programs.

$.WSCUT1. The utility program that allows a user to convert a screen image, prepared through the EDX $IMAGE utility, for use by the work session controller.

$.WSMENU. The program used to start a session between an EDX terminal and work session controller application programs.

$EDXDEF. The EDX data set that contains the definition statements for generation of the EDX supervisor. It must be edited and reassembled when the Communications Facility is installed.

$INITIAL. The program that is loaded when EDX is IPLed.

$PCCR0n. The data set that contains patches to the Series/1-PC Connect Attachment microcode. *n* is the EC level.

$LNKCNTL. The EDX data set that contains the linkage editor statements necessary to generate the EDX supervisor. It must be edited when the Communications Facility is installed or when certain devices are added to an existing Communications Facility system.

$STORAGE. The label in the EDX program header that contains the address of the program's dynamic storage area. The Communications Facility books call the dynamic storage area $STORAGE.

$4980CSA. The data set containing the control store for a 4980 terminal managed by the Communications Facility.

$4978CS0. The data set containing the control store for a 4978 terminal managed by EDX.

$4980CS0. The data set containing the control store for a 4980 terminal managed by EDX.

$4980ISA. The data set containing the image store for a 4980 terminal managed by the Communications Facility.

$4978IS0. The data set containing the image store for a 4978 terminal managed by EDX.

$4980IS0. The data set containing the image store for a 4980 terminal managed by EDX.

$4980R0A. The data set containing the microcode random access memory for a 4980 managed by the Communications Facility.

$4980R01. The data set containing the microcode random access memory for a 4980 terminal managed by EDX.

alias station. A station that defines an alternate name for another station in a Communications Facility configuration.

alphanumeric mode. An attribute of a 4978 device station that allows the user to enter any characters in a numeric field. Contrast with *numeric mode.*

alternate link vector. The network address of a station that is the alternate destination for messages sent to a particular station. Undeliverable transactions and X.25 control messages are sent to a station's alternate destination.

asynchronous adapter. A term used to mean feature programmable communications adapter, multifunction attachment, or 4950/5170-495 terminal/host adapter.

basic mode. An attribute of a station that causes messages sent to or received from it to be transferred without addition or removal of control characters. Contrast with *record mode.*

BCUG (bilateral closed user group). The X.25 facility that allows two DTEs to establish communication with each other by simply specifying a BCUG ID. Additional benefits vary with the network provider.

bilateral closed user group (BCUG). The X.25 facility that allows two DTEs to establish communication with each other by simply specifying a BCUG ID. Additional benefits vary with the network provider.

buffer, Communications Facility. A storage area, from 1 to 32K bytes long, preceded by a buffer header.

buffer header. Five words preceding a Communications Facility buffer that contain information about the size and content of the buffer.

buffer reference block (BRB). The first 4 words of a workspace pool, used to control the allocation of buffers and workspaces from the pool.

buffered device. A printer supported in a way that allows data to be retained and merged with variable data on write operations.

call accept control message. The Communications Facility message sent by an application program to indicate it is ready to receive data from a remote DTE through a switched virtual circuit. Call accept is sent in response to an incoming call control message.

call connected control message. The Communications Facility message sent by the X.25 I/O control program to a STD+ circuit station's direct or alternate link to indicate that the remote DTE has accepted the call initiated by the circuit station. The circuit station may now send data through the network.

call ID. A user-defined two-digit number that represents an X.25 network address. Call IDs are used when defining a switched virtual circuit station if calls are to be sent to or received from a specific X.25 network address. Call IDs are associated with X.25 addresses in the $.SYSX25 data set.

call request control message. The Communications Facility message sent by an application program to a circuit station with a contact type of USERINIT to initiate a virtual call to a remote DTE.

cause code. A 1-byte code in a restart, clear, or reset packet that indicates the reason for the restart, clear, or reset. The X.25 I/O control program includes this code in log and control messages that it sends as a result of receiving them from XHCS.

CCITT. International Telephone and Telegraph Consultative Committee. An organization of common carriers and PTTs whose main goal is to recommend standards that facilitate interconnection of communications equipment.

cell. A node in the Communications Facility configuration in which the program dispatcher runs or a non-Series/1 host system where transactions are processed.

CELL. The statement in $.SYSPD that identifies the local cell.

cell identifier. A 2-character name that uniquely identifies a cell.

CFBUF. The message buffer pool; a workspace pool in the Communications Facility control program that contains storage-queued messages.

CFTERM. The instruction that defines a 3101 or 7485 terminal, on a multifunction attachment, that is intended for Communications Facility use only.

circuit station. A station that represents an X.25 virtual circuit. See also *virtual circuit.*

clear confirmation control message. The Communications Facility message sent by the X.25 I/O control program to the STD+ circuit station's direct or alternate link informing it that the switched virtual circuit has been cleared as a result of a clear request control message sent by the circuit station.

clear indication control message. The Communications Facility message sent by the X.25 I/O control program to a STD+ circuit station informing it that the remote DTE has issued a clear request for the switched virtual circuit.

clear request control message. The Communications Facility message sent by an application program to a STD+ circuit station as a negative response to an incoming call or at any time it wants to clear the switched virtual circuit.

closed user group (CUG). The X.25 facility that defines a group of DTEs that can communicate with each other. The number of DTEs that can be in a group and additional benefits vary with the network provider.

command message. A message, the content of which is a CP command.

command processor. A part of the Communications Facility that processes CP and, with the program dispatcher, PD commands.

command processor (CP) commands. A set of Communications Facility commands used to define and control the Communications Facility configuration and display information about it.

command-processing program. A program that processes a particular CP command. The command processor controls loading and execution of command-processing programs as it receives commands.

Communications Facility buffer. A storage area, from 1 to 32K bytes long preceded by a buffer header.

Communications Facility configuration. A complete set of nodes, cells, and stations that communicate with one another through the Communications Facility.

Communications Facility/Host. The program that allows an operator at a System/370, a 30*xx*, or a 4300 to maintain a remote Series/1.

Communications Facility terminal. A device defined to the Communications Facility, controlled by an I/O control program, and accessed from a program through SEND and RECEIVE instructions.

configuration processor ($.CONFIG). The utility program used to define and modify stations and maintain the system message data set.

control message. A message defined by the Communications Facility that contains information related to controlling the X.25 network. Circuit stations with a usage type of STD+ and the X.25 I/O control program can send and receive these messages using the SEND S and SEND SM instructions. The length of the control message varies depending on which type it is. See also the name of the individual control message.

control program ($.CF). The part of the Communications Facility that includes message dispatching functions, the command processor, and the log processor.

control, 3270 ($.IO0AC0). The input/output control program that controls 3270 displays and printers attached to the Series/1.

CP commands. A set of Communications Facility commands used to define and control the Communications Facility configuration and display information about it.

CSPDIO. A module in the EDX supervisor that provides remote disk support.

CUG (closed user group). The X.25 facility that defines a group of DTEs that can communicate with each other. The number of DTEs that can be in a group and additional benefits vary with the network provider.

data circuit-terminating equipment (DCE). The equipment installed at the user's premises that provides all the functions required to establish, maintain, and terminate a connection, including the signal conversion and coding between the data terminal equipment (DTE) and the line. In the Communications Facility, software functions provide a connection point for devices capable of interfacing to a packet-switching data network as DTEs. The Communications Facility does not provide all the DCE support defined by Recommendation X.25; it cannot be an X.25 network.

data message. A message, the content of which is user data to be sent from one station to another.

data stream RPQ. An RPQ (D02428) that improves performance when a 4978 display station is managed as if it were a 3277 display station.

data terminal equipment (DTE). That part of a data station that serves as a data source, data sink, or both, and provides for the data communication control function according to protocols. In the Communications Facility, DTE is hardware or software that is capable of attaching to an X.25 packet-switching network.

D-bit. A bit in an X.25 packet header that instructs the network to wait until delivery to the remote data terminal equipment has been confirmed before confirming delivery to the sending data terminal equipment. As data circuit-terminating equipment (DCE), the Communications Facility confirms only that the SEND M instruction to the local application program has completed successfully; it does not wait for confirmation of receipt of the message.

DCE (data circuit-terminating equipment). The equipment installed at the user's premises that provides all the functions required to establish, maintain, and terminate a connection, including the signal conversion and coding between the data terminal equipment (DTE) and the line. In the Communications Facility, software functions provide a connection point for devices capable of interfacing to a packet-switching data network as DTEs. The Communications Facility does not provide all the DCE support defined by Recommendation X.25; it cannot be an X.25 network.

DDM (device descriptor module). An XHCS module that defines an HDLC line. The name of the Communications Facility line station representing the line must have the same name as its DDM.

DDSK-30. The 30 megabyte disk within the 4952, 4954, or 4956 processor or within the 4965 storage and I/O expansion unit. Because DDSK-30 is the keyword on the EDX DISK statement, EDX and Communications Facility books use it to identify the disk unit.

DDSK-60. The 60 megabyte disk within the 4954 or 4956 processor or within the 4965 storage and I/O expansion unit. Because DDSK-60 is the keyword on the EDX DISK statement, EDX and Communications Facility books use it to identify the disk unit.

DEF. The CP command that defines a new station.

DEFINE BRB. The instruction that defines a buffer reference block; storage following this instruction will be treated as a workspace pool.

DEFINE BUFFERPOOL. The instruction that defines a workspace pool within a program. This is the same instruction as DEFINE POOL and DEFINE WORK.

DEFINE DEVICE. The instruction that defines a device descriptor block for a remote disk.

DEFINE POOL. The instruction that defines a workspace pool within a program. This is the same instruction as DEFINE BUFFERPOOL and DEFINE WORK.

DEFINE VOLUME. The instruction that makes an entry for a remote disk volume in the volume descriptor cross-reference table.

DEFINE WORK. The instruction that defines a workspace pool within a program. This is the same instruction as DEFINE BUFFERPOOL and DEFINE POOL.

delivery confirmation bit (D-bit). A bit in an X.25 packet header that instructs the network to wait until delivery to the remote data terminal equipment has been confirmed before confirming delivery to the sending data terminal equipment. As data circuit-terminating equipment (DCE), the Communications Facility confirms only that the SEND M instruction to the local application program has completed successfully; it does not wait for confirmation of receipt of the message.

device descriptor module (DDM). An XHCS module that defines an HDLC line. The name of the Communications Facility line station representing the line must have the same name as its DDM.

device station. A station that represents a Series/1 terminal or printer to be managed as if it were a 3270 device in the Communications Facility configuration.

device type. The combination of station type and station subtype for a Series/1 terminal or printer being managed as if it were a 3270 device. Device type indicates which I/O control program is to control a particular device.

diagnostic aid utility ($.UT1). The utility program that allows access to various Communications Facility functions for diagnostic purposes.

diagnostic code. An optional 1-byte code in restart, reset, and clear packets that gives information about the reason for the restart, reset, or clear. The X.25 I/O control program includes this code, if XHCS sent it, in the log and control messages.

direct link vector. The network address of a station that is the default destination for messages sent by a particular station.

disk-server. A function supported by the Series/1-PC Connect Program which allows a PC user to use a Series/1 data set as a LAN-shared PC disk.

disk queue. A message queue on disk, used to hold low-priority messages destined for a particular station until the station is ready to receive them.

disk-queue data set. A data set used to hold low-priority messages destined for a particular station until the station is ready to receive them.

disk-queue data set initialization utility ($.DSINIT). The utility program that initializes a data set for the disk queuing of messages.

disk-queue file control block. A control block that contains information about a station that has a disk queue.

disk, remote. A feature of the Communications Facility that allows a program to access a disk volume that is attached to another Series/1.

dispatcher, message. The part of the Communications Facility that determines the final destination of a message and routes it through the system to that destination.

dispatcher, program ($.PD). The part of the Communications Facility that manages the processing of transactions.

DTE (data terminal equipment). That part of a data station that serves as a data source, data sink, or both, and provides for the data communication control function according to protocols. In the Communications Facility, DTE is hardware or software that is capable of attaching to an X.25 packet-switching network.

dynamic partition. An EDX partition in which the system must allocate I/O segmentation registers for each I/O request issued and deallocate them when the I/O operation completes.

EDL language extensions. A set of Event Driven Executive Language (EDL) instructions that perform various Communications Facility functions.

EDX Secondary SNA1. The IBM Series/1 Event Driven Executive Systems Network Architecture licensed program, program number 5719-SX1. EDX Secondary SNA1 allows an EDX user application, such as the Communications Facility SNA IOCP, to communicate with host applications through an SDLC connection using SNA protocols.

EDX Secondary SNA2. The IBM Series/1 Event Driven Executive Systems Network Architecture licensed program, program number 5719-XX9. This program is an extension of the EDX Secondary SNA1. It allows users to define up to four PUs in a Series/1.

EDX Secondary SNA2 support. The set of Communications Facility modules neccessary to use EDX Secondary SNA2 and up to four Communications Facility SNA PU stations in a Series/1.

EDX terminal. A terminal defined to the EDX operating system and used to perform EDX system functions.

emulated PC disk management utility ($.DSFORM). The utility program that creates, deletes, or renames a data set to be used as an emulated PC disk.

emulated PC disks. A Series/1 data set formatted as a Personal Computer (PC) disk.

emulation, 3270. The facility that allows a host processor to communicate with a Series/1 as if it were communicating with a 3270 system; also, the I/O control program that provides 3270 emulation over a BSC line ($.IO0AE0).

EQU. A statement in $.SYSPD that defines a transaction that has attributes similar to another transaction.

error indication control message. The Communications Facility message sent by the X.25 I/O control program to a STD+ circuit station's direct or alternate link to indicate that the I/O control program rejected a control message or that it could not initiate a call for an SVC circuit station.

error log. EDX uses the $LOG utility to log errors to a disk or diskette data set.

error message. A message sent to the Communications Facility system log to indicate that an error has occurred.

F. (1) The CP command that modifies the attributes of an existing station. (2) The PD command that modifies an entry in the path table or the transaction identifier table.

facilities. A set of optional characteristics and capabilities available from the network provider to switched virtual circuits during call establishment. In the Communications Facility, these facilities may be included in the SVC circuit station definition and/or in a call request control message from an application program. See also the individual facility.

fast select (FS). The X.25 facility that allows data to be appended to a call request, call clear, or call accept packet. Contrast with *fast select restricted.*

fast select restricted (FSR). The X.25 facility that allows data to be appended to a call request or call clear packet. Call accept packets are not allowed. Contrast with *fast select.*

FCB (file control block). A control block that contains information about a station that has a disk queue.

FILE. The CP command that displays disk-queuing parameters and changes or assigns a station's data set name for disk-queuing.

file control block (FCB). A control block that contains information about a station that has a disk queue.

FS (fast select). The X.25 facility that allows data to be appended to a call request, call clear, or call accept packet. Contrast with *fast select restricted.*

FSR (fast select restricted). The X.25 facility that allows data to be appended to a call request or call clear packet. Call accept packets are not allowed. Contrast with *fast select.*

gateway PC. A Personal Computer (PC) connected to a Series/1 by the Series/1-PC Connect Attachment.

GET B. The instruction that gets a buffer from a workspace pool.

GET S. The instruction that gets storage from the system storage pool, S$POOL.

GET W. The instruction that gets storage from a workspace pool.

halted station. A station whose station block has been deleted from S$POOL.

HDLC (high-level data link control). The group of standards defining the link level for communications with a public data network. As defined by Recommendation X.25, LAPB conforms to one subset of standards and SDLC normal response mode conforms to a different subset. The Communications Facility adheres only to the LAPB subset.

header, buffer. Five words preceding a Communications Facility buffer that contain information about the size and content of the buffer.

header, message. 24 bytes at the beginning of a message that contain such information as its origin, destination, and priority.

high-speed loader ($.HSL). A small, efficient loader used by the program dispatcher to load transaction-processing programs.

hold, input. A condition of a station in which all messages it sends are discarded.

hold, output. A condition of a station in which messages can be sent to it, but it can't receive messages from its queue.

host communication. A function supported by the Series/1-PC Connect Program which allows a PC user to pass through to one or multiple hosts as a 3278 or to transfer files to and from any host over SDLC.

host management utility ($.HMU). The transaction-processing program that allows an operator at a Series/1 to maintain a remote Series/1.

host processor. A computer in a Communications Facility configuration where control functions are performed; it may be a Series/1 or another type of computer.

image library. A library of screen images in data set $.WSCIMG that can be displayed through the work session controller BI command.

image library management utility ($.WSCUT1). The utility program that converts a screen image that was created by the EDX $IMAGE program and stores it in $.WSCIMG for use by the work session controller. It is also used to display the image.

incoming call control message. The Communications Facility message sent by the X.25 I/O control program to a STD+ circuit station to indicate that a remote DTE wishes to begin communications.

initialization data set ($.SYSIPL). A data set containing CP commands that is read when the Communications Facility is loaded.

input hold. A condition of a station in which all messages it sends are discarded.

input/output control program (IOCP). A program that handles transmission of messages to and from a particular type of device or line in a Communications Facility configuration.

inter-LAN. A function supported by the Series/1-PC Connect Program which allows a PC to communicate with other computers on another LAN.

interrupt confirmation control message. The Communications Facility message sent by the X.25 I/O control program to a STD+ circuit station informing it that the remote data terminal equipment received an interrupt control message sent by the circuit station.

interrupt control message. The Communications Facility message sent by an application program to a STD+ circuit station or by the X.25 I/O control program to a STD+ circuit station's direct or alternate link. The interrupt control message includes 1 byte of data that is sent across the X.25 circuit without flow control.

IOCP (input/output control program). A program that handles transmission of messages to and from a particular type of device or line in a Communications Facility configuration.

IPL transaction data set ($.SYSLCC). The data set containing IPL transactions, used to IPL remote nodes on a Local Communications Controller ring.

LAN (local area network). A PC network created when Personal Computers are connected by hardware and provided with software support to enable them to communicate with each other.

language extensions. A set of Event Driven Executive Language (EDL) instructions that perform various Communications Facility functions.

LAN-shared disk. A Series/1 data set formatted as an emulated PC disk that is shared by the PCs in the same LAN.

LAN-shared printer. A Series/1 printer used as an emulated PC printer that is shared by the PCs in the same LAN.

LAPB (link access procedure balanced). The link protocol used by the XHCS for controlling the X.25 network access link.

LCI (logical channel identifier). A number assigned to a logical channel to uniquely identify the channel and all packets flowing through it. In the Communications Facility, permanent virtual circuits have LCIs in their station definitions to assign them to that particular logical channel.

line station. A station that represents a telecommunication line in a Communications Facility configuration.

line type. The combination of station type and station subtype for a communication line. Line type indicates which I/O control program (such as point-to-point or 3270 control) is to control a particular line.

LINK. (1) The CP or $.CONFIG command that defines a connection between two stations—either a direct link vector or an alternate link vector. (2) The work session controller high-level language subroutine that enables an application program to complete its own execution by loading and executing some other application program.

linked station. A station that has a single specified station as the default destination of messages it sends.

local area network (LAN). A PC network created when Personal Computers are connected by hardware and provided with software support to enable them to communicate with each other.

local node. The node from which the Communications Facility configuration is being viewed.

local station. A station that exists at the local node.

log message. A message that is sent to the Communications Facility system log.

log processor. The part of the Communications Facility that formats error and informational messages and sends them to the system log.

logical channel. A logical conduit for packets on a DTE-to-DCE link. Logical channels are assigned to virtual circuits dynamically (SVCs) or statically (PVCs). All packets on the circuit flow through the logical channel. See also *logical channel identifier.*

logical channel identifier (LCI). A number assigned to a logical channel to uniquely identify the channel and all packets flowing through it. In the Communications Facility, permanent virtual circuits have LCIs in their station definitions to assign them to that particular logical channel.

logical unit (LU) station. A station that represents an SNA logical unit (a terminal or a printer) in a Communications Facility configuration.

LS. The work session controller command that sets a lock sequence for a static screen terminal.

LU (logical unit) station. A station that represents an SNA logical unit (a terminal or a printer) in a Communications Facility configuration.

mapped partition. A partition that contains the common area (system tables and station blocks).

message. A unit of data to be transmitted from one station to another.

message buffer pool (CFBUF). A workspace pool in the Communications Facility control program that contains storage-queued messages.

message data set ($.SYSMSG). The data set containing the text of error messages and informational messages issued by the Communications Facility and, optionally, by user programs.

message dispatcher. The part of the Communications Facility that determines the final destination of a message and routes it through the system to the destination.

message header. 24 bytes at the beginning of a message that contain such information as the origin, destination, and priority of the message.

message priority. An attribute of a message that determines where it is placed in the destination station's message queue.

message queue. A queue of messages destined for a single station, either in processor storage or on disk.

message queue management utility ($.UT2). The utility program that allows the user to examine and purge messages on the $.WASTE queue or any other disk queue.

message sequence number. A number associated with a message representing its sequence with respect to its origin.

message station. A queue of messages, not associated with a Communications Facility program or device.

message type. An attribute of a message that indicates whether it is a data, command, log, transaction, or status message.

MOV. The instruction that moves data, allowing indirect and indexed moves.

multinode mode. A mode of operation of the message dispatcher that allows routing of messages to stations in remote nodes.

name, station. A 1- to 8-character alphanumeric value that uniquely identifies each station in a node.

network address. A 4-character hexadecimal value that uniquely identifies a station in the network. The first two characters are the node assignment, and the last two are the station address.

network configuration data set ($.SYSNET). The data set containing the definitions of all the stations in a node and of remote stations that will be communicated with from that node.

node. A Series/1 in the Communications Facility configuration.

node assignment. The first 2 characters of a station's network address; they uniquely identify a node in the network.

node station. A station that represents a remote node in a Communications Facility configuration.

non-display mode. An attribute of a terminal or device station that causes lowercase data to be converted to uppercase. For 4978 device stations, it also causes non-display output fields to be converted to lowercase, which displays as blanks. Contrast with *text mode.*

nontransparent mode. A mode of BSC transmission that prohibits bit patterns with a value less than X'40' from being transmitted as data.

numeric mode. An attribute of a 4978 device station that allows the user to enter only digits, decimal points, and minus signs in a numeric field. Contrast with *alphanumeric mode.*

output hold. A condition of a station in which messages can be sent to it, but it can't receive messages from its queue.

packet. The basic transmission unit on a data link accessing an X.25 network. See also *packet size.*

packet size. The size of the largest data packet sent to an X.25 network. The packet size is defined in the circuit or DxE line station definitions.

packet switching. The process of routing and transferring data by means of addressed packets so that a channel is occupied only during the transmission of a packet.

packet-switching data network (PSDN). A communications network that uses the mechanism of packet switching to transmit data. See also *packet switching.*

page. In EDX, a 256-byte block of storage.

panel. A screen image for a 3270 display station or a Series/1 device being managed as a 3270 display station.

panel data set ($.SYSPNL). The data set containing panels displayed by the $.PANEL, $.CFMENU, and $.IO014E8 programs.

panel design aid ($.PANEL). An interactive program for creating panels to be displayed at a 3270-type terminal.

panel print utility ($.PNLUT1). The utility program that prints the descriptions of panels created by the $.PANEL program.

pass-through. In the Communications Facility, the facility that allows users of Series/1 and 3270 devices on the Series/1 to communicate with host applications without any additional application programs in the Series/1. The devices on the Series/1 appear as 3270 devices to the host application.

passthrough control message. The Communications Facility message that contains user-defined control information. The X.25 I/O control program sends this message to the circuit station's direct or alternate link when it receives a message with the Q-bit on in an incoming data packet. The application program sends this message to a STD+ circuit when it wants the data sent in a data packet with the Q-bit on.

path. The route used to send a transaction from the program dispatcher in one cell to the program dispatcher in another cell or to a host transaction-processing system.

PATH. A statement in $.SYSPD that defines a path.

path definition table. A table used by the program dispatcher that defines the paths to remote cells.

path table. Synonymous with *path definition table.*

PCC. The instruction that defines a Series/1-PC Connect Attachment to the EDX supervisor.

PC header. A 20-byte storage area prefixed to the data sent from the gateway PC. It contains information such as the origin and destination stations, data length, and function identifiers.

PD commands. A subset of Communications Facility CP commands used to control the operation of the program dispatcher.

Personal Computer. As used in Communications Facility, an IBM Personal Computer.

permanent virtual circuit (PVC). A permanent virtual connection that provides services similar to a leased line. Data sent to the network through a logical channel being used as a PVC is always delivered to a specific logical channel at a specific DTE destination in the network. In the Communications Facility, PVC circuit stations represent permanent virtual circuits.

physical unit (PU) station. A station that represents an SNA physical unit in a Communications Facility configuration.

preferred path. The path used to route transactions for unknown cells.

printer data set ($.SYSPRT). The data set containing image and control store loads for 4978 terminals and members used to support printers as formatted devices.

priority, message. An attribute of a message that determines where it is placed in the destination station's message queue.

program dispatcher ($.PD). The part of the Communications Facility that manages the processing of transactions.

program dispatcher (PD) commands. A subset of Communications Facility CP commands used to control the operation of the program dispatcher.

program dispatcher data set ($.SYSPD). The data set containing CP commands, path definitions, transaction definitions, and transactions, that are to be processed when the program dispatcher is started.

program-to-program communication. A function supported by the Series/1-PC Connect Program which allows a PC user to communicate between PC programs on separate IBM PC networks or to communicate between Series/1 programs and PC programs.

protocol identifier (ID). The optional 4-byte field included in the user data portion of a call request or incoming call packet that provides an additional means of screening incoming calls from an X.25 network. In the Communications Facility, switched virtual circuit station definitions may include a protocol ID.

PSDN (packet-switching data network). A communications network that uses the mechanism of packet switching to transmit data. See also *packet switching.*

PU (physical unit) station. A station that represents an SNA physical unit in a Communications Facility configuration.

PUT TCB. The instruction that creates a task control block.

PVC (permanent virtual circuit). A permanent virtual connection that provides services similar to a leased line. Data sent to the network through a logical channel being used as a PVC is always delivered to a specific logical channel at a specific DTE destination in the network. In the Communications Facility, PVC circuit stations represent permanent virtual circuits.

Q. (1) The CP command that displays information about Communications Facility stations, BSC lines, EDX terminals, EDX-SNA control blocks, and EDX-XHCS control blocks. (2) The PD command that displays the transaction identifier table, the path definition table, remote disk definitions, and scheduled transactions.

qualifier bit (Q-bit). A bit in a data packet header that indicates the type of information in the packet. Q-bit of 0 (off) means the data is user data; Q-bit of 1 (on) means the data is application-defined control data. Communications Facility application programs send and receive control data in the passthrough control message.

queued message utility ($.UT2). The utility program that allows the user to examine and purge messages on the $.WASTE queue or any other disk queue.

R. The PD command that removes an entry from the path table or the transaction identifier table.

RC. (1) The PD command that sets the number of times the program dispatcher will attempt to load a program when storage is not available. (2) The work session controller command that reads the cursor position from a static screen terminal.

reason code. A code indicating the reason an undeliverable message is in the $.WASTE queue.

recognized private operating agency (RPOA). The X.25 facility that defines a particular transit network through which a virtual call is to be routed internationally, when more than one RPOA transit network exists at an international gateway.

Recommendation X.25. The CCITT recommendation that defines standards for the connection of processing equipment to a packet-switching data network. It addresses the physical level, the link level, and the packet level. The Communications Facility adheres to the recommendation as amended in 1981.

record mode. An attribute of a station that causes control characters to be removed from messages received from it and control characters added, if necessary, to messages sent to it. Contrast with *basic mode.*

remote cell. Any cell in the Communications Facility configuration other than the local cell.

remote disk support. A feature of the Communications Facility that allows a program to access a disk volume that is attached to another Series/1.

remote IPL utility ($.PDIPL). The transaction-processing program that sends an IPL bootstrap or nucleus to a remote Series/1, causing that Series/1 to IPL itself.

remote management utility ($.RMU). The transaction-processing program that processes transactions received from a host management utility—$.HMU or Communications Facility/Host.

remote node. Any node in the Communications Facility configuration other than the local node.

remote station. A station in a remote node.

RESERVE. A statement in $.SYSPD that reserves space in the path table or the transaction identifier table.

reset confirmation control message. A Communications Facility message sent by the X.25 I/O control program to a STD+ circuit station's direct or alternate link informing it that the circuit has been reset as a result of a reset initiated by the application program.

reset indication control message. A Communications Facility message sent by the X.25 I/O control program to a STD+ circuit station's direct or alternate link informing it that the circuit has been reset by the remote data terminal equipment or as a result of an error detected by the network or XHCS.

reset request. A Communications Facility message sent by the STD+ circuit station to reinitialize the virtual call or the permanent virtual circuit. Reinitialization removes all data and interrupt packets in the network.

RETRY mode. An attribute of a circuit station that causes the I/O control program to reestablish the virtual call when a call is cleared for a switched virtual circuit whose contact type is INIT. Contrast with *STOP mode.*

REV (reverse charging). The X.25 facility used to request that the cost of a communications session be charged to the called data terminal equipment.

reverse charging (REV). The X.25 facility used to request that the cost of a communications session be charged to the called data terminal equipment.

RMU. The transaction identifier associated with the Communications Facility program $.RMU.

RPOA (recognized private operating agency). The X.25 facility that defines a particular transit network through which a virtual call is to be routed internationally, when more than one RPOA transit network exists at an international gateway.

RTE. A statement in $.SYSPD that overrides a transaction's primary cell identifier.

S$CFEQU. The Communications Facility system equate table, which defines names for the fields of the station blocks, the buffer header, and the message header.

S$POOL. The system storage pool; a workspace pool in the common area, used by the Communications Facility for station blocks and work areas.

SC. The work session controller command that positions the cursor on a static screen terminal.

SCHD. The transaction identifier associated with the Communications Facility program $.PDSCHD.

scheduler ($.PDSCHD). The transaction-processing program that schedules another transaction for processing at a specific time.

SCS (SNA character string). A character string composed of EBCDIC controls, optionally intermixed with end-user data, that is carried within a request/response unit.

SCS mode. An attribute of a printer device station that causes data sent to it to be interpreted as an SNA character string. Contrast with *3270 mode.*

secondary cell identifier. A field of a transaction, whose meaning is defined by the program that processes the transaction. It may, for example, be the identifier of the cell to which an acknowledgment is to be sent.

secondary mode. An attribute of a point-to-point line station that defines the local node as the secondary Series/1. Contrast with *primary mode.*

secondary transaction identifier. A field of a transaction, whose meaning is defined by the program that processes the transaction. It may, for example, be the identifier of a transaction to be sent as an acknowledgment.

SEND E. The instruction that sends an error message to the Communications Facility system log.

SEND L. The instruction that sends an informational message to the Communications Facility system log.

SEND M. The instruction that sends a data message from a Communications Facility buffer.

SEND S. The instruction that sends a status message from an EDX text area. See also *status message.*

SEND SM. The instruction that sends a status message from a Communications Facility buffer.

Series/1-PC Connect. A combination of Series/1-PC Connect Attachment and software support for Series/1 and PC. This makes some of the resources of Series/1 available to PCs in a LAN.

Series/1-PC Connect Attachment. This hardware feature includes a Series/1-PC Connect Attachment card, which installs in the Series/1; a PC extender card, which installs in the PC; and a 20-foot cable which connects the two cards.

Series/1-PC Connect Attachment microcode patch ($PCCR0n). The data set that contains patches to the Series/1-PC Connect Attachment microcode. *n* is the EC level.

set $.CFD storage utility ($.SETSTG). The utility program that changes the size of the message buffer pool and the file control block pool in $.CFD.

SET CFBUF. The CP command that sets the warning level and the class 2 usage level of the message buffer pool (CFBUF).

SET LOG. The CP command that assigns a log device or station for system log messages.

SNA character string (SCS). A character string composed of EBCDIC controls, optionally intermixed with end-user data, that is carried within a request/response unit.

SNA logical unit station. A station that represents an SNA logical unit (a terminal or a printer) in a Communications Facility configuration.

SNA physical unit station. A station that represents an SNA physical unit in a Communications Facility configuration.

SNA prompt screen (panel). An optional menu put out by SNA logical units that prompts the terminal operator to log on to a host SNA application.

ST. (1) The CP command that displays message statistics and Local Communications Controller hardware statistics. (2) The work session controller command that sets the transaction identifiers of the transactions that are to be sent when a PF key on a static screen terminal is pressed after a WK command.

started station. A station that is represented by a station block in S$POOL.

static partition. An EDX partition in which the system maps all user and common areas for I/O at initialization time.

station. A named unit of hardware or software managed by the Communications Facility.

station address. The last two characters of a station's network address. They uniquely identify the station within the node.

station block. A control block in S$POOL that contains information about a started station.

station name. A 1- to 8-character alphanumeric value that uniquely identifies each station in a node.

station subtype. An attribute of a station that further defines its type; for example, a device-type station may have a subtype such as 3101, 4978, or printer.

station type. An attribute of a station that specifies its type (for example, line, device, terminal, user, or message station).

status message. A message sent with a SEND S or SEND SM command that results in a unique return code (+6) when it is received. A status message is used to (1) tell a station to stop or halt. (2) send X.25 control messages between the X.25 I/O control program and applications linked (alternate or direct) to a STD+ circuit station.

STOP mode. The attribute of a circuit station that causes the I/O control program to stop the station when a call is cleared for a switched virtual circuit whose contact type is INIT. Contrast with *RETRY* mode.

stopped station. A station, represented by a station block in S$POOL, for which the flow of messages has been temporarily stopped. Messages sent to a stopped station are undeliverable.

subchannel. A device address supported under loaded microcode control of Local Communications Controller or Series/1-PC Connect Attachment. It is used to receive or transmit control or data messages.

subtype, station. An attribute of a station that further defines its type; for example, a device-type station may have a subtype such as 3101, 4978, or printer.

SVC (switched virtual circuit). A dynamically-established connection between two pieces of data terminal equipment (DTE). The switched virtual circuit is the packet network equivalent of a switched or dial-up line. In the Communications Facility, SVC circuit stations represent switched virtual circuits.

switched virtual circuit (SVC). A dynamically-established connection between two pieces of data terminal equipment (DTE). The switched virtual circuit is the packet network equivalent of a switched or dial-up line. In the Communications Facility, SVC circuit stations represent switched virtual circuits.

system initialization data set ($.SYSIPL). The data set containing CP commands that are to be executed when the Communications Facility is loaded.

system station. The station block, named $.DISP, that represents the Communications Facility control program.

system storage pool (S$POOL). A workspace pool in the EDX supervisor, used by the Communications Facility for station blocks and work areas.

terminal, Communications Facility. A device defined to the Communications Facility, controlled by an I/O control program, and accessed from a program through SEND and RECEIVE instructions.

terminal, EDX. A terminal defined to the EDX operating system and used to perform EDX system functions.

terminal, work session controller. A terminal managed by the work session controller and accessed from an application program by means of work session controller transactions.

terminal station. A station that represents a 3270 control unit, display, or printer attached to a Series/1, or a station block used to emulate a 3270 control unit, display, or printer in a Communications Facility configuration.

text mode. An attribute of a terminal or device station that causes lowercase data to be transferred without modification. Contrast with *non-display mode.*

TID (transaction identifier). The 4-character name of a transaction.

TID statement. A statement in $.SYSPD that identifies a transaction to be processed in the local cell.

transaction. A special-format, user-defined message, routed through the Communications Facility network by the program dispatcher and processed at its destination by a specific transaction-processing program.

transaction identifier (TID). The 4-character name of a transaction.

transaction identifier (TID) table. The table that defines the transactions to be processed in the local cell. It contains, for each transaction, its identifier, its attributes, and the name and attributes of its associated programs.

transaction message. A message, the content of which is a transaction.

transaction-processing program. A program designed to process transactions. The program dispatcher controls loading and execution of transaction-processing programs as it receives transactions.

transaction table. Synonymous with *transaction identifier (TID) table.*

transaction type. A 2-character indicator of the actions that occur when a transaction is entered: loading one of four types of program, creating a station, and/or sending the transaction message to the station.

transparent mode. A mode of BSC transmission that allows any bit pattern to be transmitted as data.

type, line. The combination of station type and station subtype for a communication line. Line type indicates which I/O control program (such as point-to-point or 3270 control) is to control a particular line.

type, message. An attribute of a message that indicates whether it is a data, command, log, transaction, or status message.

type, station. An attribute of a station that specifies its type (for example, line, device, terminal, user, or message station).

type, transaction. A 2-character indicator of the actions that occur when a transaction is entered: loading one of four types of program, creating a station, and/or sending the transaction message to the station.

UBRETRY mode. The attribute of an SNA LU station that causes the I/O control program to attempt to rebind a session that was terminated by the host. Contrast with *UBSTOP mode.*

UBSTOP mode. The attribute of an SNA LU station that causes the I/O control program to stop the station when the host terminates a session. Contrast with *UBRETRY mode.*

undeliverable message. A message that cannot be delivered because its destination station is stopped or is unknown to the message dispatcher.

UP. The PD command that specifies whether or not the program dispatcher is to load program $.UP*xxxx* when it receives the undefined transaction *xxxx.*

usage class. The class designation of a station that determines what level of CFBUF usage will suspend the execution of SEND instructions from the station.

usage level. The amount (in percentage) of the Communications Facility message buffer pool (CFBUF) in use.

user PC. Other PCs in the LAN that are linked to the gateway PC. They share the resources of Series/1 through the gateway PC.

user station. A station that represents a user or system program in a Communications Facility configuration.

vector station. A station block that represents a remote station in a multinode Communications Facility configuration.

virtual call. A temporary logical connection between two pieces of data terminal equipment. Virtual calls are placed through switched virtual circuits. See also *switched virtual circuit.*

virtual circuit. A logical connection established between two pieces of data terminal equipment. It can be permanent—defined when you subscribe to your network port—or it can be switched—dynamically established when a call is placed. The Communications Facility manages stations that represent these circuits; XHCS manages the circuits. See also *switched virtual circuit* and *permanent virtual circuit.*

volume protection utility ($.PDBSTS). The utility program that allows a user to gain exclusive control of a disk volume directory in a remote Series/1 before loading a program that will update that directory.

volume station. A station that represents emulated PC disks which reside on a Series/1 volume.

warning processor. A task in $.CF that sends an error message to the system log when use of the message buffer pool (CFBUF) exceeds its warning level.

window. (1) The number of data packets a DTE or DCE can send across a logical channel before waiting for authorization to send another data packet. It is the main mechanism for pacing the flow of X.25 packets across an X.25 network. In the Communications Facility, window is defined in the line or circuit station definition. (2) In $.PANEL, the area on the screen that can be seen one time when defining a 3270 panel.

work session controller ($.WSC). The part of the Communications Facility that allows an application program to communicate with multiple EDX devices attached to any Series/1 in the network.

work session controller data set ($.WSCIMG). The data set containing images that can be displayed through the work session controller and members used to save data for transaction-processing programs.

work session controller sample program ($.WSMENU). The program that can be used to start a session between an EDX terminal and work session controller application programs.

work session controller terminal. A terminal managed by the work session controller and accessed from an application program by means of work session controller transactions.

workspace pool. An area of processor storage from which the Communications Facility programs allocate buffers and work areas. The pool includes information used to control the allocation of elements in the pool.

WSC. (1) The transaction identifier associated with the Communications Facility program $.WSC. (2) The command that starts a work session controller terminal.

X.25 data set. The data set containing two-digit call IDs and their associated X.25 network addresses used during call establishment for switched virtual circuits. As shipped, it contains 10 records of /*.

X.25 header. A header that precedes an X.25 control or data message. It identifies what type of message, control or data, is contained in the message. This header is 2 bytes long in the Communications Facility Version 2.0.

X.25 network. A packet-switching data network that adheres to the standards defined by the CCITT Recommendation X.25.

X.25 network address. A field of up to 15 binary-coded decimal (BCD) digits that identifies the DTE to which a call is directed or from which a call originated. The Communications Facility provides a data set, $.SYSX25, in which the user may relate this address to a two-digit call ID.

XHCS. The IBM Series/1 Event Driven Executive X.25/HDLC Communications Support licensed program (Program Number 5719-HD2). XHCS allows an application program, such as the Communications Facility, to communicate with remote applications through an HDLC communications link using X.25 packet level procedures.

XOFF mode. An attribute of a point-to-point line station that causes data for that line to be written in nontransparent mode. Contrast with *XON mode.*

XON mode. An attribute of a point-to-point line station that causes data for that line to be written in transparent mode. Contrast with *XOFF mode.*

3101. A term used to represent 3101 terminals (models 12, 13, 22, and 23); 3161, 3163, or 3164 terminals in 3101 emulation mode; and IBM Personal Computers running in 3101 emulation mode as if they were 3277 model 2 terminals.

3270 control ($.IO0AC0). The input/output control program that controls 3270 displays and printers attached to the Series/1.

3270 emulation. The facility that allows a host processor to communicate with Series/1 as if it were communicating with a 3270 system; also, the input/output control program that provides 3270 emulation over a BSC line ($.IO0AE0).

3270 mode. An attribute of a printer device station that causes data sent to it to be interpreted as a 3270 data stream. Contrast with *SCS mode.*

3270 panel utility ($.PNLUT1). The utility program that prints the descriptions of panels created by the $.PANEL program.

3270 pass-through. In the Communications Facility, the facility that allows users of Series/1 and 3270 devices on the Series/1 to communicate with host applications without any additional application programs in the Series/1. The devices on the Series/1 appear as 3270 devices to the host application.

4978/printer data set ($.SYSPRT). The data set containing image and control store loads for 4978 terminals and members used to support printers as formatted devices.

Index

Special Characters

A

D

F

G

K

M

T

U

V

W

X

Numerics

READER'S COMMENT FORM

SL23-0104-2

IBM Series/1 EDX
Communications Facility
Design and Installation Guide

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.
Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

COMMENTS

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

Fold and tape

Please do not staple

Fold and tape

No postage necessary if mailed in the U.S.A.

Business Reply Mail

First Class Permit 40 Armonk New York

Postage will be paid by:

International Business Machines Corporation
Information Development, Dept. 4J1/037
P.O. Box 10500
Palo Alto, CA 94303-9945

Fold and tape

Please do not staple

Fold and tape

READER'S COMMENT FORM

SL23-0104-2

IBM Series/1 EDX
Communications Facility
Design and Installation Guide

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

COMMENTS

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

Fold and tape | **Please do not staple** | Fold and tape

No postage necessary if mailed in the U.S.A.

Business Reply Mail

First Class Permit 40 Armonk New York

Postage will be paid by:

International Business Machines Corporation
Information Development, Dept. 4J1/037
P.O. Box 10500
Palo Alto, CA 94303-9945

Fold and tape | **Please do not staple** | Fold and tape

SN23-8600-00

IBM / **Technical Newsletter**

This Newsletter No. SN23-8600
Date March 27, 1987

Base Publication No. SL23-0104-2
S1-34

Previous Newsletters None

IBM Series/1
EDX Communications Facility
Design and Installation Guide

Program Number 5719-CF2

This technical newsletter provides replacement pages to add new information for Version 2.1 PTF 2 of the Communications Facility. The replacement pages remain in effect for subsequent versions and modifications unless specifically altered. Pages to be replaced are:

iii to x	131, 132
x.1, x.2 (added)	213 to 222
3 to 6	222.1, 222.2 (added)
11 to 14	227 to 230
45 to 48	230.1, 230.2 (added)
48.1, 48.2 (added)	239 to 242
91, 92	253 to 264
99, 100	264.1, 264.2 (added)
121 to 124	452.1 to 452.8 (added)
124.1, 124.2 (added)	465, 466

Technical changes are indicated by a vertical line to the left of each change.

Summary of Changes

Technical and typographical corrections have also been made.

The following functions have been added: support for the COMBASE parameter of the EDX system statement; retry support for $.IO0AC0; remote TCBs for $.IO0670; storage requirements for $.IO0674; an appendix that describes the hardware configuration required to connect a 3101 device to a Series/1.

Note: *Please file this cover letter at the back of the manual to provide a record of changes.*

IBM Corporation, Department 4J1/036-MS33, 1520 California Avenue, P.O. Box 10500, Palo Alto, California 94304

Printed in USA

SL23-0104-02

SL23-0104-2
Program Number:
5719-CF2
File Number: S1-34

Printed in U.S.A.