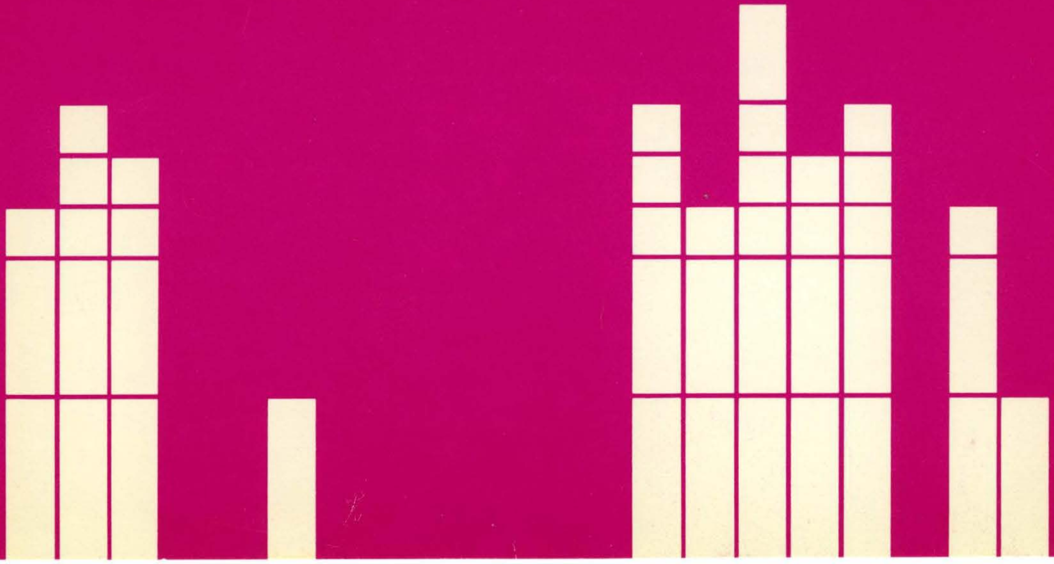
Publication Number
GC31-2068-1

4700 Finance Communication System

Controller Programming Library

Volume 3
Communication Programming

IBM



4700 Finance Communication System

Controller Programming Library

Volume 3 Communication Programming

Publication Number
GC31-2068-1

File Number
S370/4300/8100/S34-30

Second Edition (January 1984)

This edition applies to Release 3 of the 4700 Finance Communication System and to all subsequent releases and modifications until otherwise indicated in new editions or Technical Newsletters (TNLs).

The programming RPQs described in this manual, and all licensed materials available for it, are provided by IBM on a special quotation basis only under the terms of the Agreement for IBM Licensed Programs (Z120-2800). Your local IBM branch office can advise you regarding the special quotation and ordering procedures.

Changes occur often to the information herein; before using this publication in connection with the installation or operation of IBM equipment, consult the latest *IBM System/370 Bibliography of Industry Systems and Application Programs*, GC20-0370, for the editions that are applicable and current.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the address given below; requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comments is provided at the back of this publication. Address comments concerning the contents of this manual to IBM Corporation, Information Development, Department 78C, 1001 W.T. Harris Blvd., Charlotte, NC 28257, U.S.A. IBM may use or distribute any information you supply in any way it believes appropriate without incurring any obligation to you.

Preface

This is Volume Three of the *4700 Controller Programming Library*—one of a set of six volumes for the 4700 programmer. The table on page v summarizes the topics covered in the other volumes. All six volumes are available from your local IBM office or IBM representative using a single order number, GB0F-1387.

Who Should Read This Book

You need this volume if you are writing programs to communicate with either the host system or any secondary control point device over a Synchronous Data Link Control/Systems Network Architecture (SDLC/SNA) telecommunication line, or with the host over a Binary Synchronous Communication—Type 3 (BSC3) host link.

How This Book Is Organized

The manual is divided into two parts: the first part explains SDLC/SNA link programming; the second part explains BSC3 link programming. Each part ends with a thumb–tabbed reference section describing, in alphabetic order, the instructions you use to program the link.

At the end of the manual are appendixes describing the machine instruction formats, the COPY instruction parameter lists, the program checks, and the status codes and statistical counters. A glossary and index follow the appendixes.

What Else To Read

Before you use this manual, you should understand the topics and the information in the following manuals:

1. *OS/VS-DOS/VSE-VM/370 Assembler Language*
2. *IBM 4700 Finance Communication System, Controller Programming Library, Volume 1: General Controller Programming*
3. *IBM 4700 Finance Communication System, Controller Programming Library, Volume 6: Control Program Generation*, GC31-2071.
4. *IBM 4700 Finance Communication System, System Summary*, GC31-2016.
5. *IBM 3600 Finance Communication System: System Summary*, GC27-0001.
6. *IBM Synchronous Data Link Control, General Information*, GA27-3093.
7. The following Systems Network Architecture publications:
   - *Systems Network Architecture, Concepts and Products*, GC30-3072.
   - *Systems Network Architecture, Technical Overview*, GC30-3073.
   - *Systems Network Architecture, Sessions between Logical Units*, GC20-1868.

   In addition, you should be familiar with and have available the *Systems Network Architecture Format and Protocol Reference Manual*, SC30-3112.

VOLUME 1: GENERAL CONTROLLER PROGRAMMING (GC31-2066)

- Programming Concepts
- Using the General Programming Instructions
- Coding Rules
- General Programming Instructions (Reference)
- General Machine Instruction Formats
- Parameter List Reference
- Status Codes, Program Check Codes, and Error Messages
- Programming Techniques for 3600 Compatibility

VOLUME 2: DISK AND DISKETTE PROGRAMMING (GC31-2067)

- Basic Disk and Diskette Programming
- Extended Disk and Diskette Programming
- Disk and Diskette Programming Instructions (Reference)
- Disk and Diskette Status Codes
- Disk and Diskette Parameter List Reference
- Disk and Diskette Machine Instruction Formats

VOLUME 3: COMMUNICATION PROGRAMMING (GC31-2068)

- SNA/SDLC Communication Programming
- SNA/SDLC Communication Macros
- SNA/SDLC Communication Instructions (Reference)
- BSC3 Host Communication Programming
- BSC3 Communication Instructions (Reference)
- Communication Status Codes
- Communication Parameter List Reference
- Communication Machine Instruction Formats

VOLUME 4: LOOP AND DCA DEVICE PROGRAMMING (GC31-2069)

- General Protocols for Displays
- 4704 and 3604 Displays
- 3270-Compatible Displays and Devices
- 3606 and 3608 Financial Services Terminals
- General Protocols for Printers
- 4710 and 4720 Printers
- 3610, 3611, and 3612 Printers
- 3615 and 3616 Printers
- 3270-Compatible Printers
- 3624 Consumer Transaction Facilities
- Data Stream Mapping (DATSM) Protocols
- Device Status Codes
- Device Parameter List Reference

VOLUME 5: CRYPTOGRAPHIC PROGRAMMING (GC31-2070)

- Cryptographic Concepts and Facilities
- Enciphering and Deciphering Operations
- Generating and Exchanging Cryptographic Keys
- Authenticating Messages
- Validating and Translating PINs
- Using the Encrypting PIN Keypad
- Host Support Encryption Routines (BDKDPRS and BDKDES)
- Cryptographic Programming Instructions (Reference)
- System Cryptography
- Cryptographic Machine Instruction Formats
- Cryptographic Parameter List Reference
- Cryptographic Program Checks and Status Codes

VOLUME 6: CONTROL PROGRAM GENERATION (GC31-2071)

- Overall View of Control Program Generation (CPGEN)
- Sample CPGEN
- CPGEN Macro Statements (Reference)
- Using the Local Configuration Facility (LCF)
- CPGEN Messages
- LCF Messages
- ALA Configuration Macros

Figure 0-1. 4700 Controller Programming Library (GBOF-1387)

Summary of Amendments

GC31-2068-1 (January, 1984)

This edition replaces GC31-2068-0. You should review this manual for changes and additions, which are marked with the same change bar you see at the left of this summary.

This edition now includes the description of the SDLC/SNA alternate telecommunications line attachment (ALA) custom feature (RPQ 8V0132), obsoleting the *Systems Network Architecture-Primary Custom Feature Description,* (GC31-2509). You can use this information to control communication between the 4700 controller and a 4730 Personal Banking Machine.

Users of the Systems Network Architecture-Primary (SNA-Primary) custom feature should refer to the descriptions of the alternate SDLC link in this and other *4700 Controller Programming Library* publications (the *Custom Feature Description* manual for the SNA-Primary RPQ, GC31-2509, is now obsolete).

Contents

Figures

Chapter 1. Programming the 4700 Communication Links

The IBM 4700 Finance Communication System depends on the host computer for compiling and assembling application programs and for managing the data network of which the 4700 system is a part. These operations depend on the telecommunication line between the 4701 controller and the host system called the host link.

The 4701 controller also acts as an SNA/SDLC primary logical unit (PLU) when attached to a secondary logical unit (SLU) over the alternate line attachment (ALA), a 4700 telecommunication custom feature.

Application programs operating in the controller can send and receive data over the links if the programs issue the proper commands for the type of link being used. This manual tells you about the types of links used by the 4700 system, their protocols, and how to write programs to send and receive data over those links.

The host link can be direct between the 4700 controller and the host's communication controller, or it can be through data circuit-terminating equipment (DCE) over a common carrier line for a remote 4700 controller.

Note: The alternate line operates the same as the IBM 4700 Systems Network Architecture–Primary RPQ 8V0132, attaching the 4701 to SNA physical unit (PU) types 1 and 2 and to logical unit types 0, 1, 2, 3, 4, and 6. RPQ-system users should refer to this programming library when programming ALA/SNA line devices rather than to the SNA-Primary RPQ *Custom Feature Description*, GC31-2509, which is now obsolete.

The manual has two parts. Part 1 contains the description of an SNA/SDLC (Systems Network Architecture/Synchronous Data Link Control) communication host and alternate lines, their protocols (including X.21 and those needed for a multiuse communication loop), and the 4700 communication and assembler instructions that operate them. Part 2 describes the BSC3 (Binary Synchronous Control-Type 3) data link, its protocols, and appropriate 4700 assembler instructions.

Part 1. Programming an SNA/SDLC Link

This part of the manual describes the information you must know to write application programs that communicate with another system over an SNA/SDLC communication link. This link uses the communication protocols of Systems Network Architecture (SNA) and the synchronous data link control (SDLC) line disciplines.

Chapter 2 of this first part describes general host link programming, followed by Chapter 3 describing programming for the alternate line attachment (ALA). Chapter 4 describes use of the SNA/SDLC host communication macros, and Chapter 5 describes the 4700 assembler communication instructions you must include in an application program to operate either the host or ALA link.

Chapter 2. SNA/SDLC Host Link Programming

The design of both the 4700 application program operating the host link and the host programs must be based on how the transmitted data is used. The main purpose of this chapter, however, is to discuss basic concepts—sending data to the host program, and receiving data sent by the host to the controller's application program.

This chapter uses VTAM as an example of the communication access method, and a VTAM application program as an example program with which the controller program communicates. As a result, you should consult the following references before designing the communications portion of an applications program:

- *Virtual Telecommunications Access Method (VTAM) Concepts and Planning*, GC27-6998
- *Advanced Communications Functions for VTAM (ACF/VTAM) Concepts and Planning*, GC38-0282
- *Telecommunications Access Method (TCAM) Concepts and Applications*, GC30-2049
- *Advanced Communications Functions for TCAM (ACF/TCAM) Concepts and Planning*, GC30-3049

The path between the work station, where your application program operates, and the host comprises the following elements:

- A controller application program, which operates on behalf of one or more work stations.
- The subsystem, which operates on behalf of one or more controller application programs.
- A switched or nonswitched communication link, including the X.21 communication link and multiuse communication loop.
- A 3704 or 3705 Communication Controller using Network Control Program/VS (NCP/VS) or Partitioned Emulator Program/VS (PEP/VS).
- A communications access method such as VTAM, ACF/VTAM, TCAM, or ACF/TCAM[1]. This manual uses VTAM for examples.

The program in the host with which your application program communicates may be a customer-written program or it may be a program product such as CICS/VS or IMS/VS[1].

---

[1] *IMS/VS, TCAM, and ACF/TCAM are used with OS/VS only.

SNA Considerations

Systems Network Architecture (SNA) is a comprehensive set of rules governing how communication products can communicate with each other. All 4700 subsystems that attach to a host communications controller over SDLC lines use this architecture. You should understand the following areas of SNA:

- Network definition: In many of the macro instructions and statements that define the 4700 subsystem in the communication controller's network control program (NCP) and in the host processor's access method, you must describe the physical and logical capabilities of the subsystem in SNA terms. You can also specify certain actions that are to occur when the network is activated and deactivated—also in SNA terms.

- Protocol control by the application program: You must understand both the host and controller disciplines used to control the session and exchange data when you design application programs that use the host link.

The preface of this manual lists the prerequisite and companion SNA publications you should have available before programming the SDLC/SNA host link. These publications describe the total set of all protocols that are possible in a network containing SNA communication products.

SNA Terminology

You should understand the following terms before continuing, because this manual uses them frequently. The descriptions of these terms are stated from a 4700 subsystem viewpoint. For more general descriptions of these terms, such as from an architectural or industry view, refer to the *IBM Vocabulary for Data Processing, Telecommunications, and Office Systems*, GC20-1699.

Application program (AP): The user-written program responsible for the SLU functions. The AP can be used by either one or many 4700 stations, and operate within the SNA application layer.

Host processor, or *host*: Controlling processor connected to the 4700 controller by the communication link.

Physical unit (PU): That portion of the 4700 controller programming responsible for half of the SSCP-PU session. The PU also sends and receives maintenance statistics over the SSCP-PU session.

Primary logical unit (PLU): The portion of the host application programming that supports the primary half-session of the host-controller LU-LU session flow.

Secondary logical unit (SLU): The portion of the controller programming that supports the secondary half-session of the host-controller LU-LU session flow and the controller half-session of the SSCP-PU session flow. In this chapter, SLU means the user application program (AP) or logical work station communicating with the host.

System services control point (SSCP): An SNA function in the host that controls host-controller (SSCP-PU) and host-controller application program (SSCP-LU) sessions.

SNA Profiles Supported

A combination of controller data and user-written application programming makes the following SNA functions possible:

- Physical unit (PU)-Type 2.
- Transmission Subsystem (TS) Profiles 3, 4, and 7 for LU-LU operation.
- Function Management (FM) Profiles 3, 4, 7, and 18 (for LU-LU operation).

This support implies that logical unit (LU) types 0, 1, 2, 3, and 6 can be used as long as the host application program specifies the appropriate parameters when a session begins. In turn, the application program must support the sequence of commands, requests, replies, and options issued by the host. Data transfer between the Communications Network Management/Controller Support (CNM/CS) facility in the controller and the Network Problem Determination Application (NPDA) at the host is through the SSCP-PU session.

The 4700 controller is a *Type 2 physical unit.* NCP configuration defines the 4700 controller by specifying the PUTYPE=2 operand in the appropriate PU macro instruction.

If a 4701 controller attaches to a communications controller over *switched* SDLC lines, the maximum number of logical units in the controller must be specified by the MAXLU operand of the NCP PU macro instruction. The controller can contain from one to a maximum of 60 work stations (MAXLU=60).

Device Addressing

Network definition (CPGEN) defines the following addresses:

- The SDLC station address. The SDLC station address identifies the controller to the host, which uses the address to route messages to the controller. The CPGEN process sets this address to X'C1', but you can change it during transmission of the diskette image from the host to the controller. The address may also be altered using a command available during controller startup, by a command to the system monitor program, or by an application program language instruction.

  Note: If the controller address switches are set to other than zero, the switch address takes precedence.

- The logical unit (LU) address. Each work station using the communication link has a one-byte LU address used to establish sessions between a host processor application program and the work station. The LU value assigned to a work station corresponds to the station ID of that work station.

If the CPGEN COMLINK macro specified LUASSIGN, making the optional ASSIGN instruction functions available to you, the LU addresses have the following added states:

- Active LU addresses—those LU addresses currently assigned to work stations. Each work station specified by a STATION macro to use the communication link has an LU address corresponding to the specified station number.

- Inactive LU address—any address ranging 2 through 60 (0 and 1 are reserved) that is unassigned by a STATION macro.

Communication Assembler and Macro Instructions

The same instructions used to transfer data between the controller and the 4700 terminals are used to transfer messages between the controller and the host; however, these instructions use the CP operand rather than the logical device address. LREAD receives messages from the host system, LWRITE sends messages to the host system, and LCHECK determines when a write operation is actually completed.

Your program can issue these instructions at any point. Generally, however, the program routine selected by the ACP (asynchronous host) entry point contains the LREAD instruction for reading host data. Your program can issue LWRITE to send data to the host at any time except when certain other types of write operations have already begun. The most data that an LWRITE instruction can send is 4095 bytes. Your program should issue LCHECK following an LWRITE to ensure that data sent by the LWRITE was actually transferred from the station's storage to the network successfully, and that the storage is available for later writing or reading.

Optional ASSIGN instruction functions, included by specifying LUASSIGN on the COMLINK macro, provide the following capabilities:

- Assigning an LU address to a station.
- Assigning an LU address to the LU free pool.
- Determining the station ID for a given LU address.
- Determining the LU address for a given station.

For compatibility with some 3600 system releases, the 4700 system also provides high-level SNA/SDLC communication macro instructions. These instructions define work areas and registers, create low-level code for processing link messages, link the application program to user-written error (debugging) routines, begin and end host-controller sessions, as well as send and receive messages and responses. Refer to Chapter 5, "4700 SNA/SDLC Communication Macros" for a description of these macros and how they are used.

Starting Host-Controller Communication

To allow a work station to communicate with the host, you must first do the following:

- Allow the work station to communicate over the host link by specifying Y (Yes) on the CPU operand of the STATION configuration macro. Specify CPU=Y... or CPU=Y,PU... .

- Define the host communication entry point in the station's application program by entering the entry point name on the ACP= operand of the BEGIN assembler instruction. This allows the inactive station to begin operating when the controller receives a host message.

After the controller and host establish the basic communication link, the host initiates contact by sending a Set Normal Response Mode (SNRM) command. When the controller responds with a Non-sequenced Acknowledgment (NSA), the host begins the polling operations. The host transmits an SDLC normal poll sequence containing the controller address and a Receive Ready (RR) command with the poll flag set. The host can also indicate a poll while transmitting a message by setting a poll flag on in the I-frame.

If, after initial contact, the host encounters a condition (such as an 'out of buffers' condition) that does not permit receipt of a message, polling continues with the Receive Not Ready (RNR) command. The controller remains active, but transmits no data.

Messages

This section gives detailed operating information for the types of messages and commands in a controller–host (SLU–PLU) session. Note that the controller responds to many of the commands received by the SLU; this is also discussed under "Controller Fields and Indicators."

Message Structure

Each unit of communication between the host and the controller is called a *Basic Link Unit* (BLU). The BLU contains the following fields:

- Beginning SDLC Flag

- SDLC Address

- SDLC Control

- Basic Transmission Unit (BTU) or Path Information Unit (PIU)

- SDLC Frame Check Sequence

- Ending SDLC Flag

The Basic Transmission Unit (BTU) or Path Information Unit (PIU) contains the following fields:

- Transmission Header (TH)
- Basic Information Unit (BIU)

The Basic Information Unit (BIU) contains the following fields:

- Request or Response Header (RH)
- Request or Response Unit (RU)

The term *message* in this manual refers to any request or response unit (RU), including data or SNA commands (a command is simply data that performs a specific task). In this manual, the terms *message* and *RU* are used as equals when no distinction is made between a request or response. Where a distinction is made, a request RU is called a *request*, and a response RU is called a *response*. RUs transmitted from the host to the controller are *outbound* RUs. RUs transmitted from the controller to the host are referred to as *inbound* RUs.

All messages contain a *Transmission Header* (TH). The controller supports the Format Identifier 2 (FID2) Transmission Header for host communication. The TH contains such information as segmenting flags, flow indicators, origin address (OAF), destination address (DAF) and a sequence number field. The TH is developed in an internal buffer for all SLU write operations. For read operations, the TH is used by the controller to verify and route the message or response.

All messages and responses (other than middle- or end-of-segment) have *request/response headers* (RH). This information is supplied by both the SLU and the controller. The SLU supplies this information in the write control field for write operations and the controller supplies this information to the SLU in the read control field on read operations.

Following the RH is the *request/response unit* (RU). The RU is the data portion of the message. For commands, it identifies the type of command and may include additional information; for negative responses, it carries the sense information; and for certain commands it carries additional information. For some data RUs, the first part of the data may be a header called a Function Management (FM) header.

Message Types and Flow

SNA defines two flows for messages: normal flow, where messages are processed sequentially (first-in/first-out), and expedited flow, where messages of this flow type are processed sequentially, but before any normal flow messages in the network. The following are definitions of each message type by function and flow:

Session Control: Used by the SSCP to activate or deactivate the SSCP-PU and SSCP-LU sessions and by the PLU and SLU to activate or deactivate the LU-LU session. Additionally, they are used to start, clear, and resynchronize the data flow on the LU-LU session. Session control commands all use the expedited-flow.

Network Services: Formatted function management data (can be viewed as commands) used on the SSCP-PU and SSCP-LU sessions. These messages always use the normal-flow.

Expedited Flow Data Control: Expedited-flow commands used to control data flow. These may be sent by the PLU or SLU on the LU-LU session.

Normal Flow Data Control: Normal-flow commands used to control data flow. These may be sent by the PLU or SLU on the LU-LU session.

Normal Flow: Normal-flow data messages that contain data generated by the PLU and SLU on the LU-LU session or by the SSCP and SLU on the SSCP-LU session (Network Services).

Responses: Messages that acknowledge the receipt of all types of normal and expedited messages on either the SSCP-PU, SSCP-LU, or LU-LU sessions.

LU-LU Session Messages

The three types of LU-LU session control commands are expedited-flow session control commands, normal-flow data control commands, and expedited-flow data control commands. The commands allowed during any type of LU-LU session depend on the transmission subsystem (TS) or Function Management (FM) profiles selected in the BIND parameters, as described under "SNA Profiles Supported," earlier in this chapter.

The controller does not enforce or restrict the commands allowed for a TS or FM profile, but allows the application program to read or write the commands listed. The following are descriptions of the commands and their types.

Session Control Commands

Bind: The PLU sends Bind to initiate a session. For non-negotiable Binds, the SLU checks the session parameters and returns a positive response if the Bind parameters are acceptable, or a negative response with sense codes if the Bind parameters are unacceptable. The negotiable Bind allows the SLU to return a positive response with at least 26 bytes of updated session parameters. These parameters indicate the incompatibility with the PLU session parameters. If the PLU finds the updated session parameters acceptable, it sends a Start Data Traffic (SDT) command. If the parameters are unacceptable, the PLU sends an Unbind.

The data accompanying the Bind comprises at least 26 bytes of parameters, a 1-byte field specifying cryptography options, a possible 1-to-9-byte cryptography field, a 1-byte binary count indicating the length of the resource (host application program) name, the 1-to-8-character resource name, and the user data. If the SLU configuration specifies either *OPTIONS=BIND* in the COMLINK CPGEN macro or *CPU=...,BIND* in the STATION macro, the SLU receives all Bind parameter and data; otherwise, the SLU receives only the resource name length, the resource name itself, and a byte of binary zeros. The total length of the Bind cannot exceed 256 bytes. The in-session bit is set in the SMSCST when the SLU generates a positive response.

Clear: The PLU sends the Clear command to clear all messages between the PLU and SLU. The SLU response, generated by the controller, purges all messages between the SLU and the PLU. Clear cancels all network messages and sets all sequence numbers in the network to zero. The SLU then enters data-flow-reset state. If the PLU issues a Clear command, your program may have to perform resynchronization with the host program.

Request Recovery: If the SLU detects a critical error, it sends Request Recovery to the PLU, which should then stop data transfer until the programs can be resynchronized.

Set and Test Sequence Numbers: The PLU issues Set and Test Sequence Numbers (STSN) to resynchronize data flow (for example, during session restart). Before sending this command, the PLU must stop network data flow with either with a Clear command or by ending and restarting the session. When the PLU issues STSN, each network element examines the action code in the first byte and resets the sequence numbers accordingly. This resets sequence numbers throughout the network.

Five bytes of data accompany STSN. The first byte contains action codes for the SLU and PLU sequence numbers. The next 2 bytes may contain a new SLU sequence number, and the last 2 bytes may contain a new PLU sequence number. Bits 0 and 1 of the action code refer to the SLU sequence number, bits 2 and 3 refer to the PLU sequence number, and bits 4 through 7 are reserved. The action codes (setting of bits 0 and 1, and bits 2 and 3) are:

00 (Ignore)	The sequence number is not affected by this command.
01 (Set)	The applicable number has been altered in the PLU and in the controller.
10 (Sense)	The PLU requests the current applicable sequence number in a response from the SLU.
11 (Set/test)	The applicable sequence number has been altered in the PLU and in the controller. The SLU's sequence number should be compared to the number supplied in the STSN; the result of the comparison should then be indicated in the response.

The SLU responds to STSN with either a negative response, or a positive response containing either no data or five bytes of data. A negative response causes a program check. A positive response without data means that the sequence number or numbers are acceptable. A positive response with data means the SLU is sending its version of the sequence number.

The 5 data bytes sent with the positive response contain at least a result code in the first byte, and can contain an SLU sequence number in the next two bytes and a PLU sequence number in the last two bytes. Bits 00 and 01 of the result code refer to the SLU sequence number; bits 02 and 03 refer to the PLU sequence number. Bits 04 through 07 are reserved. The result code meanings for each pair of bits are:

00 (Reset)	This is a user-defined result code for LU type 0, and is reserved for other LU types.
01 (Positive)	The sequence number received in the STSN is equal to the sequence number in use by the SLU.
10 (No Number)	The SLU sequence number requested by the PLU is invalid.
11 (Negative)	The applicable sequence number field contains the number requested (response to a 10 action code), or the PLU number should be changed to the number provided by the SLU (response to action code 11).

Start Data Traffic: The PLU sends Start Data Traffic (SDT) to indicate that the session is established and communication may begin. This command removes the SLU from data-flow-reset state.

Unbind: The PLU sends Unbind to end a session. Unbind removes the SLU from the in-session state.

Expedited-Flow Data Control (DC) Commands

Quiesce at End-of-Chain: An LU sends Quiesce at End-of-Chain (QEC) to request that the receiving LU stop sending data at the end of this data chain and reply with the Quiesce Complete message.

Release Quiesce: The same LU that issues QEC issues Release Quiesce (RELQ) to indicate that data flow, which stopped when the QC or Shutdown Complete message was received, should resume. RELQ resets the quiesce state in the SLU.

Stop Bracket Initiation: Either LU sends Stop Bracket Initiation to request that the receiving LU stop sending BB and Bid bracket commands. The receiving LU may continue to send BB and Bid until it receives the Stop Bracket Initiation command. See "Bracket Protocol," later in this chapter.

Request Shutdown: The SLU sends Request Shutdown to request an orderly end to the session. If the PLU agrees to end the session, it responds with a Shutdown command or Unbind command, depending on application protocol.

Shutdown: The PLU sends Shutdown as part of the orderly end of a session. It indicates that the SLU should stop sending data and prepare to end the session. When the SLU is ready to end the session, it sends a *Shutdown Complete* command.

Shutdown Complete: The SLU sends Shutdown Complete to tell the PLU that it is ready to end the session. The SLU then receives a Clear or Unbind, depending on the protocol used.

Signal: One LU can send Signal to the other LU, regardless of the status of the normal flow. Signal contains a four-byte code; the first two bytes are the signal field, and the last two bytes are the user field.

Normal-Flow Data Control (DC) Commands

Bid: Bid is sent by either LU to request permission to send data. Bid is used with bracket protocol (see "Bracket Protocol"). If the receiver of Bid (first speaker) cannot receive the data (for example, it is in a synchronous dialog), it sends a negative response. When the first speaker is ready to receive the data, it sends a *Ready to Receive* command.

Bracket Initiation Stopped: Either LU sends Bracket Initiation Stopped (BIS) to reply to a Stop Bracket initiation command. Once this command has been sent, the sender may not use BB or BID (see "Bracket Protocol," later in this chapter).

Cancel: Either LU sends Cancel to indicate that an error has occurred in the group of chained messages currently being sent (see "Data-Chaining Protocol" on page 2-32). The chained messages already sent should be discarded by the receiving LU. The next message sent should be an only- or first-in-chain message.

Chase: Chase is sent by either LU to ensure that the receiving LU actually receives and responds to all messages. When the sending LU reads the response to the Chase, the messages preceding Chase have been received and all response requirements have been satisfied.

Data (Normal-Flow): Either LU sends data after the LU-LU session is established. When transmitting data from the SLU, the data is written directly from the segment addressed by the LWRITE instruction. Data can be accompanied by a response request, a begin bracket, an end bracket, a change direction indicator, or chaining flag.

Before your program issues the LWRITE command to transfer data to the host, it must set the appropriate SMSCWF flags. Data transfer is complete when control returns to the SLU after execution of LCHECK CP. An LCHECK CP occurs automatically if the program issues an LWRITE CP and the number of outstanding messages has exceeded the number specified in the CPGEN.

The LREAD instruction places data received from the PLU in the addressed segment area. The received data can also include a response request, a begin bracket, an end bracket, a change direction indicator, a chaining flag, and a code selection indicator. Your program must test the SMSCRF flags to know what types of data was received.

Data and Control (Normal-Flow): Data and control is the same as the Data message, but also includes FM header control information at the beginning of the message.

Logical Unit Status: Either LU can send Logical Unit Status (LUSTAT), including four bytes of user-defined data. Such a command might be sent in place of a response where none is allowed by "no response" protocol.

Quiesce Complete: Either LU sends Quiesce Complete (QC) to indicate that data transfer is suspended. QC is normally a reply to a Quiesce at End-of-Chain (QEC) message. To resume data transfer, the LU that sent QC must receive a Release Quiesce (RELQ) message.

Ready to Receive (RTR): The "first speaker" LU in a bracket protocol issues RTR to cancel a negative response it sent to the bidder LU. If the first speaker originally answered the bidder's Bid command request to send data with a negative response, the RTR says the first speaker is now ready to receive data. The bidder responds with either a positive response and the data or it's own negative response and sense information saying the data is no longer available.

SSCP Network Services (NS) Commands

Network Services (NS) provides protocols that the PU, LU, and system services control point (SSCP) service managers can use to control the network. These service manager functions use NS commands (formatted FM data requests) and responses on the SSCP-LU and SSCP-PU session flows.

The following are descriptions of the NS commands and the SSCP sessions where they are used:

Initiate Self: The SLU sends Initiate Self to request the PLU to initiate the LU-LU session. Initiate accompanies a variable amount of data in the following format:

```
Header*Resource*X'0000'*Optional data.
```

Header

X'00404040404040404040F3'

Resource

A 1-byte binary count of the length of the resource name followed by the 1- to 8-byte symbolic name of the PLU application program.

X'0000'

Two bytes of binary zeros.

Optional data

A 1-byte binary count of the length of the optional data followed by the data. This data includes such things as passwords passed to the PLU application. If the optional data is not included, the field must be set to X'00'.

Terminate Self: The SLU sends Terminate Self to request the the end of the LU–LU session. Terminate Self accompanies a variable amount of data having the following format:

```
X'00F3'*Resource
```

Resource

A 1-byte binary count of the length of the resource name followed by the 1- to 8-byte symbolic name of the PLU application program.

In addition, one station can send or receive the following commands on the SSCP-PU session if the CPU= operand of the STATION macro specifies 'PU':

Request Maintenance Statistics (REQMS): The SSCP sends REQMS to the PU requesting that the PU return the specified maintenance statistics.

Record Formatted Maintenance Statistics (RECFMS): The PU sends RECFMS to the SSCP and includes 4700 maintenance statistics. If REQMS requests RECFMS, then RECFMS must include the specific maintenance statistics requested. The PU can also send an unsolicited RECFMS containing various maintenance statistics.

Responses

A *response* to a message contains information about the transmission and processing of a particular message. A response can be positive or negative. A *positive* response indicates that the message has been received successfully and its content is acceptable. A *negative* (or 'exception') response indicates either that the message's content is unacceptable or that the message was not received successfully.

The work station sends and receives responses in a manner similar to data messages:

- Before sending an SLU response, the program must set SMSCWT (the write type field) to indicate the response type (shown in Figure 2-28 on page 2-55), and then issue an LWRITE instruction to the host.
- After receiving a host's response, the program must read SMSCRT (the read type field) to know the type of response received (shown in Figure 2-26 on page 2-53).

Before sending a message, the sender must indicate the type of *response protocol* the receiver is to follow. SNA defines three types of response protocol: (1) *no response* protocol indicates that the sender of the message does not want (and will not accept) a response to the message; (2) *exception response* protocol indicates that the sender of the message wants, and will accept, a response only if the message is unacceptable; (3) *definite response* protocol indicates that the sender of the message always wants a response, regardless of whether the message was acceptable or not. The request type is indicated in the message's flags field:

- Before requesting a response from the host, you must set SMSCWF (the write type field) to indicate the response wanted, set SMSCWT to indicate the message type, and then issue an LWRITE instruction.
- To detect a host response request, you must issue an LREAD instruction, then test SMSCRF (read flags field) for the response type requested by the host.

The selection of a given response protocol depends on the nature of the data being sent. For example, a financial institution may decide that two types of data are transmitted:

1. Broadcast messages noting the status of the host resources, such as "SAVINGS DATA BASE NOT AVAILABLE UNTIL 9:00"; or–
2. Messages concerning a transaction, such as "DEP ACCT 02234 400.00."

The broadcast messages are of general interest only, and a lost message does not affect the data base; therefore the no-response protocol is appropriate. The transaction messages, on the other hand, affect the orderly operation of the branch and must be verified; therefore an exception response protocol or a definite response protocol is appropriate.

Every response, whether positive or negative, is further designated by its sender as a *definite response 1* (DR1) or a *definite response 2* (DR2). Some users may not need to distinguish between the two different kinds of positive and negative responses. For those that do, the distinction may be made for any purpose understood by the user. For example, DR2 could be used to indicate whether the message has been successfully received by the application program, while DR1 could indicate whether the message has been forwarded to its ultimate destination (a display device, perhaps).

SLU-Originated Responses

The SLU controls the response protocols for Data and Data-and-Control messages. The controller sets the *definite response* protocol and a request for a DR1 response for all other nondata messages.

For example, when your program sends an Initiate Self message (see "SLU Initiation" on page 2-16) it receives either a positive (X'09') or negative (X'0D') DR1 response appropriate for a nondata message. A positive response acknowledges receipt of the Initiate message and indicates that the initiation request was passed to the VTAM application program. A negative response also acknowledges receipt of the Initiate message, but indicates that the initiation request was not passed to the VTAM application program because VTAM found an error.

When sending data to the host, the SLU indicates the response protocol desired by setting bits in SMSCWF. The host then returns that type of response. Figure 2-1 shows the responses that are possible for each protocol and the SMSCRT and SMSCWF bits that are set for each response.

The controller sends any data received with the response to the SLU. This includes at least a one-byte SNA command code for both the session control and data-flow control commands, and three bytes for Network Services commands. More data can follow the command bytes, depending on the command type. For negative responses, the command code follows four bytes of sense information.

Response Protocol Requested				**Possible Response Types**	
To request this protocol . . .	. . . set SMSCWF Field to:			To determine receipt of this response type . . .	. . . test SMSCRT Field for:
	Bit 5 (Exception)	Bit 6 (DR2)	Bit 7 (DR1)		
No Response	0	0	0	–	–
Exception Response					
DR1 only	1	0	1	Negative DR1	X'05'
DR2 only	1	1	0	Negative DR2	X'06'
DR1 & DR2	1	1	1	Negative DR1 & DR2	X'07'
Definite Response					
DR1 only	0	0	1	Positive or Negative DR1	X'01' or X'05'*
DR2 only	0	1	0	Positive or Negative DR2	X'02 or X'06'
DR1 & DR2	0	1	1	Positive or Negative DR1 & DR2	X'03 or X'07'

*SMSCRT will contain X'09' or X'0D' to a non-data message.

Figure 2-1. Response Protocol Requests and Possible SLU Responses

Host-Requested Responses

All messages from the host include response protocol indicators. The controller responds automatically to most commands, but other responses, including those for Data and Data-and-Control, must come from the SLU. Figure 2-12 on page 2-38 shows messages to which the controller responds.

For example, when the PLU sends a Bind command (see “PLU Initiation” on page 2-16, later in this chapter), the SLU may send either a positive or negative DR1 response to the command (X‘01’ or X‘05’). If the SLU issues a positive response, it is acknowledging receipt of the Bind and telling the host to continue the session. A negative response, however, acknowledges receipt of the Bind but rejects the PLU’s request.

The PLU indicates the desired response protocol by setting the PLU’s SMSCRF field; this corresponds to the SMSCWF for messages that the SLU sends. In addition to setting SMSCRF, the controller maintains a copy of the TH, RH, and three bytes of the RU used to format the SLU’s response.

The SLU can send two types of responses: positive or negative. The controller interprets the response indicated by the SLU and sends a response that is appropriate for the type requested by the PLU. For example, if DR1 definite response protocol was requested (bits 5 and 6 set to 0, bit 7 set to 1), and the SLU indicates:

- Positive response, the controller will send a positive DR1 response.
- Negative response, the controller will send a negative DR1 response.

Figure 2-2 shows all responses that can be specified by the SLU and the responses that are actually sent.

Requested Response (SMSCRF) from the PLU Response Sent by SLU (SMSCWT)	Exception Response Protocol			Definite Response Protocol			No Response Protocol
	DR1	DR2	DR1 & DR2	DR1	DR2	DR1 & DR2	None
Positive	No Response Sent	No Response Sent	No Response Sent	Positive DR1	Positive DR2	Positive DR1 & DR2	No Response Sent
Negative	Negative DR1	Negative DR2	Negative DR1 & DR2	Negative DR1	Negative DR2	Negative DR1 & DR2	No Response Sent

Figure 2-2. Controller Responses

Responses When None Are Requested: If the SLU tries to send a response where none is required, no response is sent and the LWRITE ends with either normal status, or status indicating that another message has arrived from the host that the program can obtain with LREAD.

Automatic Responses: The controller enforces immediate response mode for all messages received by the SLU on the normal LU-LU session. The SLU issues any host-required response before the program senses the ending condition code for the related instruction.

Sending Data with Responses

The negotiable Bind allows the SLU to return a positive response containing a minimum of 26 bytes of updated session parameters. These parameters indicate the compatibility with the PLU session parameters. This positive response is the only one that contains data.

All negative responses contain four bytes of data; the first two bytes are controller-defined sense codes; your program defines the last two bytes. If the write operation specifies more data to be sent than necessary, the extra bytes are ignored and the LWRITE completes successfully. "Sense Codes" on page 2-61 describes the controller-defined bit settings.

Message Segmenting

In the 4700, segmenting is the dividing of a large basic information unit (BIU) into smaller portions for transmission by the PLU to the SLU. The 4700 system performs only PLU-to-SLU, or *outbound*, segmenting; SLU-to-PLU (inbound) segmenting is not supported by the 4700.

The SLU receives and reassembles a segmented message without any indication to the user that the message was segmented. However, you should be aware that if segmenting is used, the number of controller read buffers may have to be increased. Also, a segmented message that is too short (less than 46 bytes) has its first segment written in the system log, and all additional segments are discarded. Normal operation resumes when a complete message or another first segment equal to or greater than 46 bytes is received.

How a Session Operates

Several events must occur before data can be transferred between the PLU and SLU on the LU-LU session:

1. The SDLC line must be activated by either the IPL procedure, the STRLNK instruction, or by the system monitor.

2. A physical session must be established. When the controller becomes ready, the SSCP establishes an SSCP-PU session by sending an Activate Physical Unit (ACTPU) command and reading a positive response from the controller.

3. After the SSCP-PU session is established, the SSCP establishes an SSCP-LU session by sending an Activate Logical Unit (ACTLU) command and reading a positive response from the controller. The controller sends a positive response if the LU address referenced in the ACTLU corresponds to a STATION address. The controller presents a READY message to the station.

   If the configuration (CPGEN) specified LUASSIGN, the controller also sends a positive response to an ACTLU for an inactive LU address; however, the controller rejects all messages other than ACTLU or DACTLU for an inactive LU until the controller assigns the requested LU to a station. Once assigned, the controller then presents READY to the station and the session can continue.

   Note: An LU-LU session for an inactive LU cannot be established by the PLU as the controller will reject the Bind for the inactive LU. The LUASSIGN function should only be specified for those applications that initiate a LU-LU session from the SLU (Initiate-Self).

4. When the SLU receives the Ready indication, Either the PLU or the SLU can start an LU-LU session.

The work station indicates host contact in bits 0 and 1 of the global indicator byte (GMSIND) in segment 15. Bit 0 is the contact flag. When bit 0 is set to 0, the controller has contact with the host's communication controller. Bit 1 indicates the status of the communication line adapter and determines the validity of bit 0. When bit 1 is set to 1, the adapter is in communication with the modem. When tested as pairs, the bits have the meanings shown in Figure 2-25 on page 2-52.

SLU Initiation

You can choose to have the SLU begin all LU-LU sessions. To do this, the controller presents a Ready indication to the SLU after the SSCP-LU session is established. The SLU then requests session initiation by sending the Initiate Self command to the SSCP.

The SSCP receives the Initiate-Self command and optionally checks whether the named host application program is known and active. If the application program exists, the SSCP sends a positive response to the SLU and the PLU can now initiate the session. If the application program is not known, the SSCP sends a negative response and no session is created.

If the SSCP sends a positive response to an Initiate-Self command, but finds the session cannot be established for some reason, a Network Services Procedure Error (NSPE) command can be sent to the SLU to halt the attempt to establish a session. The SLU can still issue the Initiate Self command following the NSPE, if you choose. Refer to Figure 2-10 on page 2-36 and Figure 2-12 on page 2-38 for examples of the session initiation sequence.

PLU Initiation

You can allow the PLU program to begin LU-LU sessions. The PLU initiates sessions by generating a Bind command (a subsequent positive response establishes the agreement to communicate) and a Start Data Traffic (SDT) command (which signals the SLU that data transfer may begin). A data field associated with the Bind command contains the name of the PLU application program, and the session bind parameters. See "Messages" on page 2-5 and the *SNA Format and Protocol Reference Manual* (SC30-3112) for the format of this data field.

If the ability was specified during CPGEN, the SLU examines the program name and the parameters to determine if the session should be allowed. For non-negotiable binds, the SLU returns a positive response if the parameters are acceptable. If the parameters are unacceptable, the SLU returns a negative response with sense codes to the PLU.

The negotiable Bind allows the SLU to return a positive response with a minimum of 26 bytes of updated session parameters indicating compatibility with the PLU parameters. If the PLU finds the returned parameters acceptable, it sends a Start Data Traffic (SDT) command; otherwise, the PLU sends an Unbind command indicating that the negotiable bind parameters from the SLU are unacceptable. Note that if the CPGEN specified the LUASSIGN option and the SLU receives a Bind for an inactive station, the controller rejects it. Refer to Figure 2-3 on page 2-18 for an example of the programming needed to start a PLU-initiated session, and to Figure 2-13 on page 2-39 for the complete session initiation sequence.

Transferring Data

After the LU-LU session is established and the SLU program responds to SDT, data transfer can begin. To transfer data, the program must do the following steps:

- Prepare the data field in one of the SLU's segments.
- Set the write type field (SMSCWT) to X'10' or X'11' for LU-LU session data. X'11' indicates that FM header is included at the beginning of the data. The FM header is in a variable number of bytes in the station's segment.
- Set the write flags field (SMSCWF) as desired (refer to "Using the Read and Write Flags Fields" on page 2-30).
- Issue an LWRITE CP to write data from the desired field.
- Issue an LCHECK to ensure that the write operation succeeded. If the program uses a single data area for both writing and reading, LCHECK ensures that the written data is available for possible retransmitting, if required and that no later LREAD CP destroys the data.

Figure 2-4 on page 2-19 shows the processing performed by an SLU that is sending data and requesting a definite response on the LU-LU session.

When the program issues an LWRITE instruction to send data to the host, the controller sets the LWRITE condition code if any host messages are pending or if an error occurred. The program must test the status in SMSDST to determine whether an error occurred or a message is pending.

The SLU receives data by issuing LREAD CP in the host entry point routine. The LREAD CP instruction should select a user storage location in a segment other than 14. The program then should test SMSCRT for the type of message read.

```
         BEGIN          APBNM=EXAMPLE,ACP=CPUASYNC,DATE=999999
*
SMS      EQUATE         1              SEGMENT 1 NUMBER
SEGA     EQUATE         2              HOST INPUT SEGMENT NUMBER
         COPY           DEFSMS         GET SEGMENT 1 DEFINITION
INPUT    DEFLD          SEGA,,50       HOST INPUT BUFFER
CPUTAB   TABLE          (X'00',READY),(X'09',CMDFME),(X'10',DATA),
                        (X'0D',CMDERR),(X'82',SDT),(X'A0',BIND),
                        (X'01',FME),(X'05',ERRFME),LNG=1
*
CPUASYNC EQUATE         *              HOST ASYNCHRONOUS ENTRY POINT
         SETFPL         INPUT          SET FIELD POINTER FOR HOST READ
         LREAD          CP,SEGA        READ 50 BYTES MAXIMUM FROM HOST
         BRAN           ST,STATUSRD    IF STATUS RETURNED, GO PROCESS
         SETFPL         SMSCRT         SET FIELD POINTER FOR LSEEK
         LSEEK          SMS,CPUTAB     LOOKUP TYPE OF MESSAGE RECEIVED
*        EXIT IF MESSAGE TYPE NOT IN TABLE
BIND     EQUATE         *              LET CONTROLLER SEND POS RESPONSE, ACCEPT SESSION
CMDFME   EQUATE         *              SPECIAL MESSAGE RESPONSE OK, EXIT
READY    EQUATE         *              IGNORE READY INDICATION
         LEXIT                         EXIT
*
SDT      EQUATE         *              ENTRY POINT WHEN SESSION ESTABLISHED
*        WRITE MESSAGE TO OPERATOR 'IN SESSION'
         LEXIT                         EXIT
*
DATA     EQUATE         *              ENTRY POINT WHEN DATA RECEIVED
*        PROCESS DATA
         LEXIT                         EXIT
*
CMDERR   EQUATE         *              BAD RESPONSE ON SPECIAL MESSAGE
*        LOG ERROR AND SEND 'REQUEST RECOVERY'
         LEXIT                         EXIT
*
STATUSRD EQUATE         *              HOST STATUS ROUTINE
*        CHECK STATUS RETURNED ON LREAD INSTRUCTION
         LEXIT
*
FME      EQUATE         *              POS DATA RESP RETURNED FROM HOST
         LEXIT                         EXIT
*
ERRFME   EQUATE         *              NEG DATA RESP RETURNED FROM HOST
*        PROCESS ERROR - (DEFINED BY APPLICATION PROGRAMMER)
         LEXIT                         EXIT
         FINISH
         END
```

Figure 2-3. SLU Processing of a PLU-Initiated Session

I/O Area Management

The WRT operand of the COMLINK configuration macro determines the most LWRITE CP instructions that the controller can perform at any given time. The corresponding WRT operand of the STATION macro sets the maximum for the station, which cannot exceed that specified by COMLINK. If the SLU tries to execute an LWRITE CP instruction that exceeds this limit, the controller delays execution of the instruction until a previous one is completed. Because all data messages sent to the PLU are handled sequentially, the SLU can use this operation to control the use of multiple I/O areas.

For example, if the write limit is 3 the SLU can issue four LWRITE CP instructions, one for each of four I/O areas. When the fourth LWRITE CP is executed, the first LWRITE CP is completed, and its related I/O area can be reused.

```
*
*
         BEGIN          APBNM=EXAMPLE,ACP=CPUASYNC,DATE=999999
*
SEGB     EQUATE         3              HOST OUTPUT SEGMENT NUMBER
WORKSEG  EQUATE         4              WORK SEGMENT
         COPY           DEFSMS         GET SEGMENT 1 DEFINITION
DATAFME  DEFCON         X'0110'WRITE DATA,(10),REQ DR PROTOCOL(01)
OUTPUT   DEFLD          SEGB,,40       FORTY BYTE OUTPUT AREA
SW1      DEFLD          WORKSEG,,1     SET FIRST BYTE AS SWITCHES
AWAITFME EQUATE         X'80'  SWITCH FOR RESPONSE OUTSTANDING
*
CPUDATA  EQUATE         *              ENTRY POINT TO WRITE DATA TO HOST
*        SET MESSAGE IN OUTPUT AREA
*        PFP MUST POINT TO MESSAGE END + 1
         MVFXD          SMSCWC,DATAFME SET SEGMENT 1 TO WRITE DATA AND
*                                      REQUEST DEFINITE RESPONSE PROTOCOL
CPUWRITE SETSFP         SEGB,0         SET SFP TO START OF MESSAGE
         LWRITE         CP,SEGB        WRITE DATA TO HOST
         BRAN           ST,STATUSWT    STATUS RETURNED, GO CHECK
         LCHECK         CP             INSURE WRITE BUFFER FREE TO REUSE
         BRAN           ST,STATUSWT    STATUS RETURNED, GO CHECK
         INORI          SW1,AWAITFME   SET SWITCH FOR MESSAGE IN NETWORK
         LEXIT                         WAIT FOR RESPONSE
*
CPUASYNC EQUATE         *              HOST ASYNCHRONOUS ENTRY POINT
*
STATUSWT EQUATE         *              HOST WRITE STATUS ROUTINE
*        CHECK STATUS RETURNED ON LWRITE AND LCHECK COMMAND
         LEXIT                         EXIT
         FINISH
         END
```

Figure 2-4. SLU Processing for Data Transmission

Minimum Requirements to Transfer Data

The minimum processing required to transfer data between the PLU and the SLU includes session initiation by the PLU, data transfer, and session termination by PLU. The SLU must:

- Read the Ready indication.
- Read the Bind message and return a positive response.
- Read the Start Data Traffic message.
- Read the data sent by the PLU, process the data, and issue an LEXIT instruction to wait for additional data.
- Set the write type and flags fields (SMSCWT and SMSCWF), put the data to be sent in an output area, and issue an LWRITE CP.

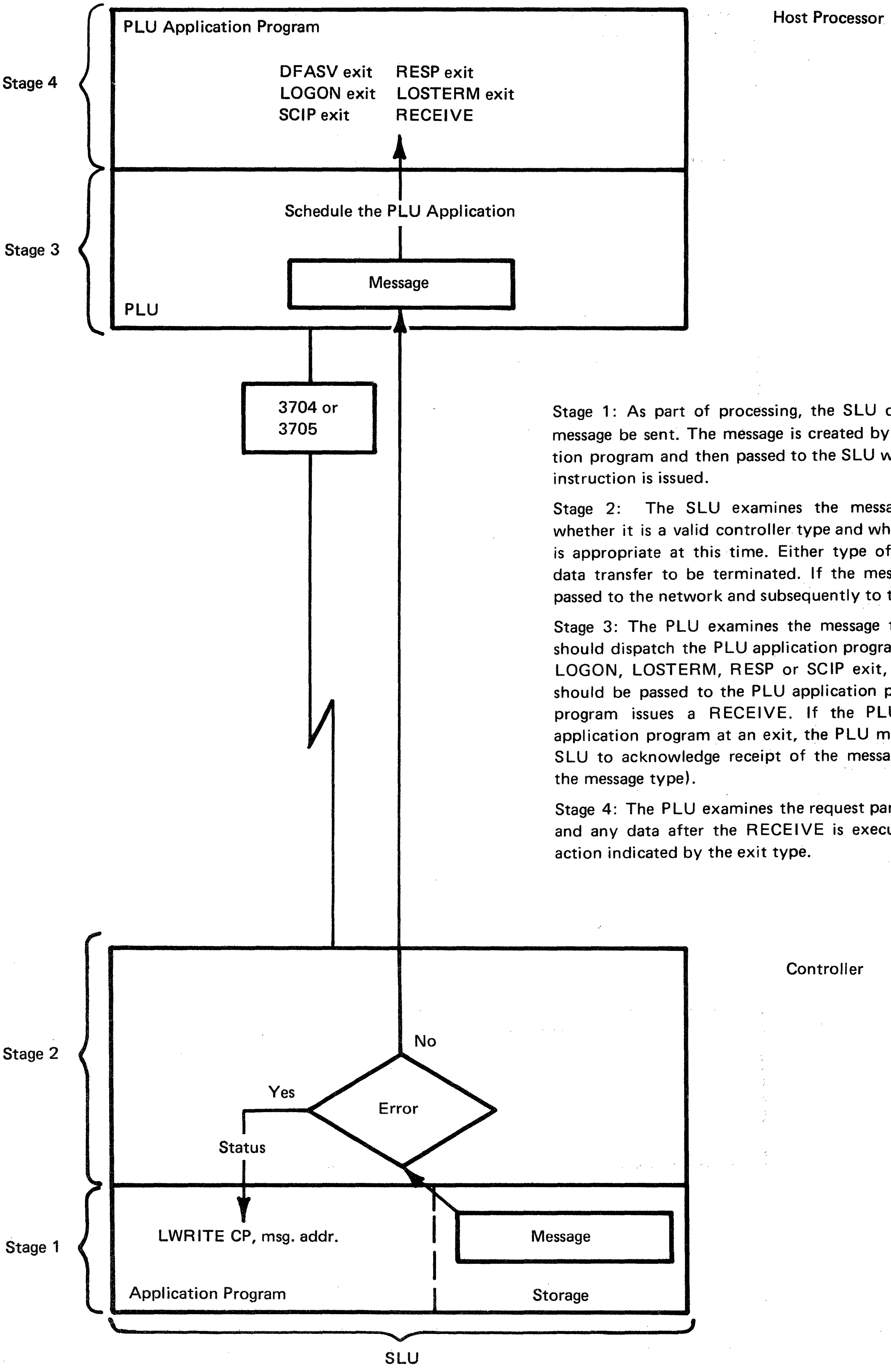
When the session ends, the application program must read and process the Clear command (if received), and then read the Unbind command. Figure 2-5 shows the minimum SLU processing required for a session.

```
         BEGIN          APBNM=EXAMPLE,ACP=CPUASYNC,DATE=999999
*
SMS      EQUATE         1              SEGMENT 1 NUMBER
SEGA     EQUATE         2              HOST INPUT SEGMENT NUMBER
SEGB     EQUATE         3              HOST OUTPUT SEGMENT NUMBER
         COPY           DEFSMS         GET SEGMENT 1 DEFINITION
INPUT    DEFLD          SEGA,,50       HOST INPUT BUFFER
OUTPUT   DEFLD          SEGB,,40       HOST OUTPUT BUFFER
DATAOUT  DEFCON         X'0010'WRITE DATA VALUE FOR SEGMENT 1
CPUTAB   TABLE          (X'10',DATA),(X'82',SDT),(X'80',CLEAR),LNG=1
*
CPUASYNC EQUATE         *              CP ASYNCHRONOUS ENTRY POINT
         SETFPL         INPUT          SET FIELD POINTER FOR HOST READ
         LREAD          CP,SEGA        READ 50 BYTES MAXIMUM FROM HOST
         BRAN           ST,STATUSRD    IF STATUS RETURNED,GO PROCESS
         SETFPL         SMSCRT         SET FIELD POINTER FOR LSEEK
         LSEEK          SMS,CPUTAB     LOOKUP TYPE OF MESSAGE RECEIVED
*        EXIT IF MESSAGE TYPE NOT IN TABLE
         LEXIT                         EXIT
*
SDT      EQUATE         *              ENTRY POINT WHEN SESSION ESTABLISHED
*        WRITE MESSAGE TO OPERATOR 'IN SESSION'
         LEXIT                         EXIT
*
DATA     EQUATE         *              ENTRY POINT WHEN DATA RECEIVED
*        PROCESS DATA
         LEXIT                         EXIT
*
CLEAR    EQUATE         *              ENTERED WHEN SESSION IS ENDING
*        WRITE MESSAGE TO OPERATOR 'SESSION ENDED'
         LEXIT                         EXIT
*
STATUSRD EQUATE         *              HOST READ STATUS ROUTINE
*        CHECK STATUS RETURNED ON LREAD INSTRUCTION
         LEXIT
*
CPUWRITE EQUATE         *              ENTRY POINT TO WRITE DATA
*        SET MESSAGE IN OUTPUT AREA
*        PFP MUST POINT TO MESSAGE END + 1
         MVFXD          SMSCWC,DATAOUT SET SEGMENT 1 TO WRITE DATA TO HOST
         SETSFP         SEGB,0         SET SFP TO START OF MESSAGE
         LWRITE         CP,SEGB        WRITE DATA TO HOST PROCESSOR
         BRAN           ST,STATUSWT    STATUS RETURNED,GO CHECK
         LCHECK         CP             INSURE WRITE BUFFER FREE TO REUSE
         BRAN           ST,STATUSWT    STATUS RETURNED,GO CHECK
         LEXIT                         EXIT
*
STATUSWT EQUATE         *              HOST WRITE STATUS ROUTINE
*        CHECK STATUS RETURNED ON LWRITE INSTRUCTION
         LEXIT                         EXIT
         FINISH
         END
```

Figure 2-5. Minimum SLU Session Processing

Data Transfer between the Controller and Host

Data transfer between the SLU and the PLU takes place in four stages, as shown in Figure 2-6 on page 2-22 and Figure 2-7 on page 2-23. The message always contains a type indicator, and can also contain a flags indicator and data. Note that for the LWRITE operation (Figure 2-6 on page 2-22), the message resides in the controller storage until it is transmitted, whereas for the LREAD operation (Figure 2-7 on page 2-23), the message is first placed in controller storage and then moved into SLU storage.



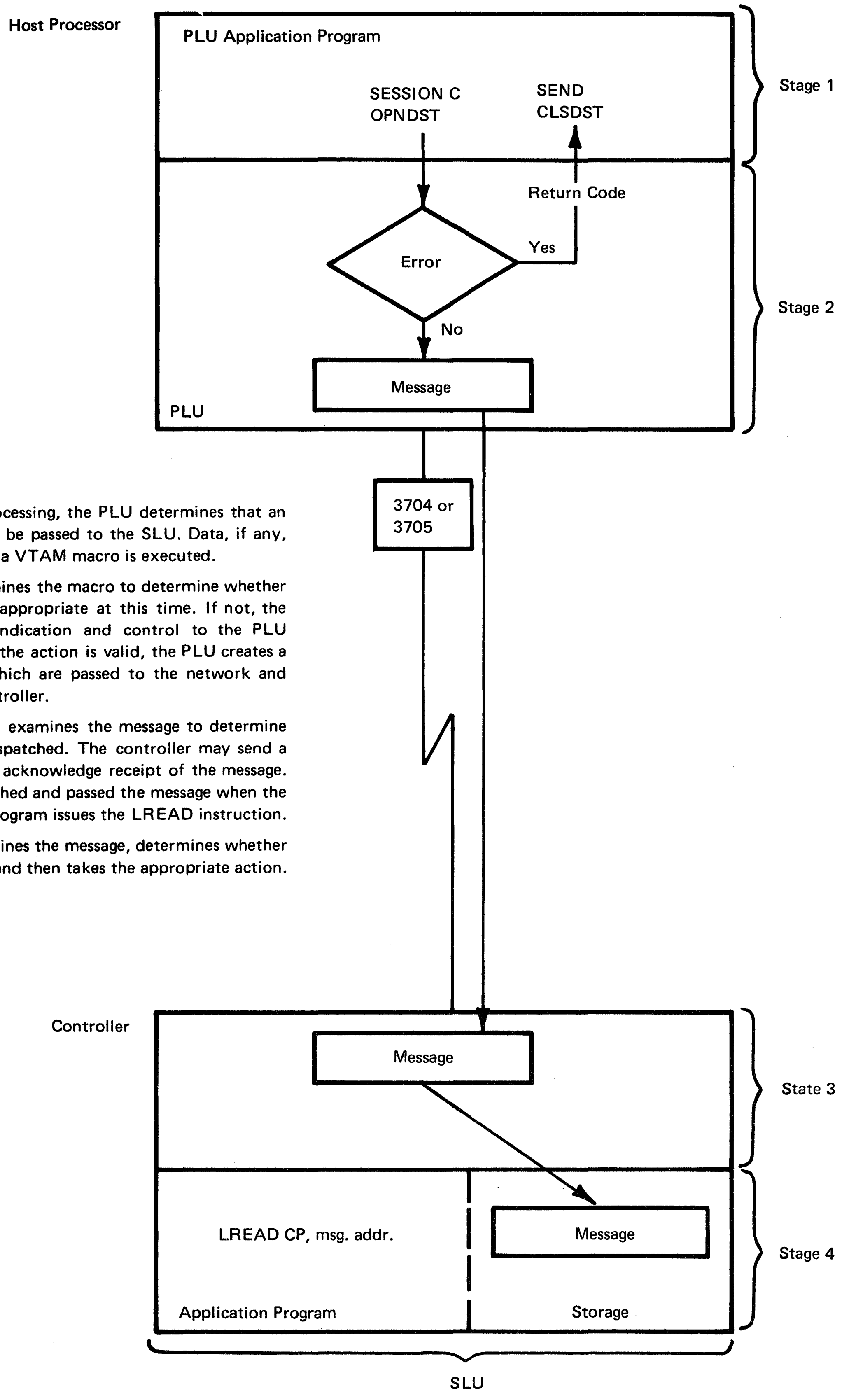
Stage 1: As part of processing, the SLU determines that a message be sent. The message is created by the SLU application program and then passed to the SLU when the LWRITE instruction is issued.

Stage 2: The SLU examines the message to determine whether it is a valid controller type and whether the message is appropriate at this time. Either type of error causes the data transfer to be terminated. If the message is valid, it is passed to the network and subsequently to the PLU.

Stage 3: The PLU examines the message to determine if it should dispatch the PLU application program at the DFASY, LOGON, LOSTERM, RESP or SCIP exit, or if the message should be passed to the PLU application program when the program issues a RECEIVE. If the PLU dispatches the application program at an exit, the PLU may respond to the SLU to acknowledge receipt of the message (depending on the message type).

Stage 4: The PLU examines the request parameter list (RPL) and any data after the RECEIVE is executed, or takes the action indicated by the exit type.

Figure 2-6. Transferring Data from the Controller to the Host



Stage 1: As part of processing, the PLU determines that an action be taken or data be passed to the SLU. Data, if any, is placed in a buffer and a VTAM macro is executed.

Stage 2: The PLU examines the macro to determine whether the indicated action is appropriate at this time. If not, the PLU passes an error indication and control to the PLU application program. If the action is valid, the PLU creates a message or messages which are passed to the network and subsequently to the controller.

Stage 3: The controller examines the message to determine which SLU must be dispatched. The controller may send a response to the PLU to acknowledge receipt of the message. The SLU is then dispatched and passed the message when the controller application program issues the LREAD instruction.

Stage 4: The SLU examines the message, determines whether a response is required, and then takes the appropriate action.

Figure 2-7. Transferring Data from the Host to the Controller

Data Transmission Status

The controller returns status from any LREAD, LWRITE, or LCHECK instruction to indicate any unexpected condition that occurred during message transmission. The application program must check the status by first testing for a condition code of '02' following the completion of the instruction, which indicates that status is in the 2-byte SMSDST field. Refer to Appendix D, "Link Status Codes" for an explanation of the status returned for an LWRITE CP or LCHECK CP instruction.

No more than one of the status bits in byte 0 of SMSDST can be set following an LWRITE CP or LCHECK CP instruction. After an LREAD CP instruction, the only byte 0 status bits that can be set with one another are the unit exception, incorrect length, and data check status.

Suspending Data Transfer

The LU-LU Data Flow Control (DFC) commands used to suspend the transfer of data on the LU-LU session are the Quiesce commands. Either the PLU or the SLU can issue these commands.

The three quiesce commands are:

- Quiesce at End of Chain (QEC) requests the receiver of this message to stop sending data after sending the last element in the chain. (A data chain is a series of related messages; see "Data-Chaining Protocol" on page 2-32, later in this chapter for additional information.)
- Quiesce Complete (QC) is an acknowledgment to QEC saying that data transfer is now suspended. When QC is sent by the SLU, the controller prevents the SLU from sending any normal-flow messages until the Release Quiesce message is received.
- Release Quiesce (RELQ) notifies the receiver that data can again be transferred.

Refer to Figure 2-8 on page 2-25 for the processing performed by the SLU when it receives the QEC message from the PLU. Refer to Figure 2-19 on page 2-46 for complete quiesce sequences.

```
*
*
        BEGIN         APBNM=EXAMPLE,ACP=CPUASYNC,DATE=999999
*
*       CP REQUEST TO SUSPEND DATA TRANSMISSION
*
        COPY          DEFSMS          COPY SEGMENT 1 DEFINITIONS
*       IN FIGURE 2-5 ADD (X'40', QEC) AND (X'43',RELQ) TO THE TABLE
*       INSTRUCTION
*
QC      DEFCON        X'22'  CODE FOR QUIESCE COMPLETE
*
QEC     EQUATE        *               ENTRY POINT WHEN QUIESCE IS RECEIVED
*       DETERMINE IF APPL PROG CAN QUIESCE
*       IF NO,SET SWITCH TO SHOW QUIESCE RECEIVED AND CONTINUE
        MVFXD         SMSCWT,QC       SET TO WRITE QUIESCE COMPLETE
        LWRITE        CP,SMSCWT       WRITE QC TO HOST
*       SMSCWT ON THE LWRITE IS A DUMMY FIELD TO ALLOW THE INSTRUCTION
*       TO EXECUTE PROPERLY,NO DATA WILL BE WRITTEN.
        BRAN          ST,STATUSWT     STATUS RETURNED,GO CHECK
        LCHECK        CP              CHECK WRITE
        BRAN          ST,STATUSWT     STATUS RETURNED,GO CHECK
*       WRITE MESSAGE TO OPERATOR 'SESSION SUSPENDED'
        LEXIT                         EXIT
*
RELQ    EQUATE        *               ENTRY POINT WHEN QUIESCE RELEASED
*       WRITE MESSAGE TO OPERATOR 'SESSION RESUMED'
        LEXIT                         EXIT
*
CPUASYNC EQUATE       *               HOST ASYNCHRONOUS ENTRY POINT
*       (see Figure 2-5)
*
STATUSWT EQUATE       *               STATUS PROCESSOR ROUTINE
        FINISH
        END
```

Figure 2-8. SLU Processing After Receiving QEC

Ending an LU-LU Session

When all data has been transferred and verified, the session can end. The SLU must end one session before it can begin a different session with either the same or another PLU. If the CPGEN specified LUASSIGN, the SLU must also end a session before it can be assigned a different LU address. The program can also end a session if an unrecoverable error occurs.

SLU Session Termination

You may decide that the SLU will end the session. The SLU must request this by issuing the Terminate message for an immediate ending or the Request Shutdown message for an orderly ending of the LU-LU flow. The SLU requests session end by doing the following:

- Preparing a data field to send with the message. The data field includes the name of the PLU application program. (Refer to "Messages" on page 2-5 for the format of this field.)
- Setting the write type field (SMSCWT) to X'A1'.
- Issuing an LWRITE CP that selects the data field.
- Checking for completion of the LWRITE CP by testing for a condition code of X'02', indicating unsuccessful completion status is in SMSDST.

- Optionally testing with LCHECK CP to ensure that the write operation completed successfully.

The SSCP receives the NS Terminate Self command and checks whether the named application program is the one participating in this season. If it is, the SSCP sends a positive nondata response, may send a Clear (depending on the SNA level being used) which purges all messages from the LU-LU session, and then sends an Unbind command which ends the session. Refer to Figure 2-22 on page 2-49 for the complete termination sequence.

For an orderly session termination, the SLU and PLU have a dialog that tells each to stop sending data and ensures that all data already sent is received. Figure 2-21 on page 2-48 is an example of the dialog used for orderly session termination.

PLU Session Termination

The programmer may decide that sessions will be terminated by the PLU. In this case, the PLU may issue an optional CLEAR command, and then an Unbind command. Refer to Figure 2-23 on page 2-50 for the complete session termination sequence. The sequence for an orderly session termination is the same as that shown in Figure 2-21 on page 2-48 , except that the sequence begins with the PLU sending shutdown (step 5).

Ending SSCP-LU and SSCP-PU Sessions

The SSCP-LU session ends when the host sends the Deactivate Logical Unit (DACTLU) command to the SLU. Note that ending the LU-LU session has no effect on the SSCP-LU session.

When the last SSCP-LU session for this controller ends, the SSCP-PU session can also end; the SSCP sends a Deactivate Physical Unit (DACTPU) command to the controller to end the SSCP-PU session.

Disconnecting the Link

When the host receives the DACTPU response, it returns a Set Disconnect Response Mode (SDRM) SDLC command to the controller. The SSCP can also disconnect immediately at any time, ending all sessions, by sending SDRM to the controller. When this occurs, all SLU's that previously received a ready indication will post loss-of-contact.

SLU Session States

The controller maintains indicators for each SLU that determine the SLU's ability to communicate with the PLU. If the SLU attempts to send a message type not currently allowed for the SLU, the controller rejects the write request and returns status bits in SMSCST. Figure 2-24 on page 2-52 shows the bit settings possible for SMSCST.

SMSCST indicates that:

- The SLU is allowed to communicate with the PLU (bit 1). This bit is set for all SLUs whose STATION macros specified CPU=Y or CPU=Y,PU during configuration.

- The station is in session (bit 0). This state is entered and the bit is set to 1 when the SLU issues a positive response to a Bind message. This state is reset when an Unbind message is received or when contact is lost.

- The station is in the quiesce state (bit 2). This state is entered (the bit is set to 0) when the SLU issues a Quiesce Complete or Shutdown Complete message. It is reset and the bit is set to 1 when a Release Quiesce message is received. When the SLU is in quiesce state, it cannot send any normal messages. This bit has no meaning if the station is not in session.

- The station is in data-flow-reset state (bit 3). This state is entered (the bit is set to 0) when a positive response is requested with a Bind command, or when the Clear message is received. It is reset (the bit is set to 1) when a Start Data Traffic message is received. This bit has no meaning if the station is not in session.

- A non-data message was sent to the PLU, but no replay was received (bit 4). The SLU enters this state when it sends any message except data and responses to the PLU; it leaves this state when it receives a response. The SLU can only send data messages or a response while in this state.

- Writing data is allowed (bit 7). This state occurs when the SLU is in session but not in quiesce or data-flow-reset state, and is permitted use of the telecommunications line by the other states (bits 0-3 set to 1).

Figure 2-9 on page 2-28 shows how the data-flow-reset, in-session, and quiesce bits change in SMSCST when various messages are transmitted or received and how loss of contact affects these bits. No other messages affect the bits in SMSCST.

Sequence Numbering

All normal flow messages transmitted between the SLU and the PLU during LU-LU flow are sequence-numbered. The SLU maintains a sequence number for normal messages from the SLU to the PLU and another sequence number for normal messages from the PLU to the SLU. Each normal message is given a sequence number one greater than the sequence number of the preceding normal message. There is one pair of sequence numbers for each session established between an SLU and a PLU. For LU-LU expedited-flow messages and for all SSCP-LU and SSCP-PU messages, identifiers are used instead of sequence numbers.

	Bit 3		Bit 0		Bit 2	
	= 0 Data Flow Reset	= 1 Not Data Flow Reset	= 1 In Session	= 0 Not In Session	= 0 Quiesced	= 1 Not Quiesced
Loss of Contact	N/A	N/A	—	→	N/A	N/A
Bind		●		●	N/A	N/A
Positive Response to Bind	←	—	←	—		
Start Data Traffic	—	→	●		N/A	N/A
Quiesce Complete		●	●		←	—●
Shutdown Complete		●	●		←	—●
Release Quiesce		●	●		—	→
Clear	←	—	●		N/A	N/A
Unbind	N/A	N/A	●—	→	N/A	N/A

Legend

● The SLU state is established or unchanged.

→ The SLU state makes this change.

N/A This state is not applicable.

Figure 2-9. Effect of Messages and Loss of Contact on Selected Bits in SMSCST

Sequence numbers are set to 0 in the network when a session is established or when a Clear message is sent. The sequence number or ID of each normal message read by the SLU is passed to SMSCRS. The sequence number of each normal message sent by the SLU is passed to the SLU at SMSCWS. The sequence numbers can be changed by the PLU with the Set and Test Sequence Numbers command so that a session can be recovered or restarted.

When the SLU is reading, and the controller encounters a sequence number error, it sends a negative response (if a response was requested) with sense code X'2001' to the PLU, and the LREAD is terminated with a data check.

When any response is read by the SLU, the response sequence number is set by the controller in SMSCSR. Note that a sequence number or ID is stored for every response. When a response is written by the SLU, the controller sets the correct sequence number in the message, but does not return the number to the SLU.

Restarting and Resynchronizing a Session

If the PLU or SLU encounters an unrecoverable error (for example, a line failure) the LU-LU session may need to be resynchronized after being restarted. This process includes returning to the last recoverable messages and optionally resetting the message sequence numbers accordingly. The application programs can include routines to retransmit lost messages.

The sequence numbers in VTAM, the NCP, and the controller are reset with the Set and Test Sequence Numbers (STSN) message; the read and write sequence numbers in segment 1 are not reset until the next read and write instructions for normal messages are executed.

The STSN message is sent by the PLU application program. It may contain either a new SLU sequence number (write), new PLU sequence number (read), or both. It also contains a set of flags for each sequence number. The flags may be set so that the applicable sequence number is ignored or set by the network, requested from the SLU, or set in the network with a request for verification by the SLU. If the STSN message is acceptable to the logical work station, the station sends a positive response without accompanying data if the STSN command specified two sets, two ignores, or a combination of both. Otherwise, the work station sends a positive response accompanied by 5 bytes of data. Refer to "Messages" on page 2-5, for additional information about STSN message and the responses.

When a session is restarted and resynchronized, the PLU will send Bind, STSN, and SDT messages. When the STSN command is sent, a dialog may occur to establish sequence numbers acceptable to both the PLU and the SLU; the dialog consists of a series of STSN messages and positive responses.

If the SLU discovers that resynchronization is required, the SLU may send either a Request Recovery command, a negative response, or an LU Status command with a description of the failure in the user sense bytes. If the PLU discovers the failure or receives a Request Recovery command from the SLU, the PLU sends a Clear message to purge all messages from the network, an STSN message to establish new sequence numbers, and then an SDT message.

Pacing

To avoid message flow at a rate that is too fast for the controller or host, you can issue an agreed-upon number of messages from a sender and wait for a response before sending more messages. This discipline, called *pacing*, can be done on just the host-controller flow, on just the controller-host flow, or on both flows. Pacing of messages to an LU is called *receive* pacing; on message flow from an LU, it is called *send* pacing.

Receive Pacing

Receive pacing protocol gives the PLU control over the number and frequency of messages sent from the SLU on an LU-LU session. When the COMLINK configuration macro specifies OPTIONS=PACE and the SLU receives pacing values in the Bind command, the controller automatically enforces pacing for each work station that communicates with the host.

During a positive response to a negotiable Bind, you can change the pacing values to anything but zero. When the first message of a sequence is sent, a bit is set in the request/response header (RH) indicating that a pacing response is to be returned. If the pacing count is exhausted before the controller receives a pacing response from the PLU, the SLU cannot send additional data messages. If the SLU issues a write operation and no pacing response is received, the write operation is deferred.

Note, however, that if a message is pending from the PLU a write operation ends with no data transfer and status of either X'4040' or X'4080'. This prevents loss of data if the controller read buffers become filled. If a message is received and a write operation is already pending, a later LCHECK instruction receives the same status and no write data is transferred.

Send Pacing

The controller controls send pacing automatically. If the pacing indicator is on in a PLU message to the SLU, the controller issues a pacing response. The pacing response can be included in a normal-flow message, or it can be an Isolated Pacing Response (IPR) if no response is required for the received message. The PLU can now send another block of messages; the controller sends a pacing response when it detects the next pacing request.

Using the Read and Write Flags Fields

The read and write flags fields (SMSCRF and SMSCWF) indicators are used for change-direction and bracket protocols, data-chaining, response protocols, and code selection. The SLU tests the bits in SMSCRF after executing an LREAD CP instruction; it sets the indicators in SMSCWF before issuing an LWRITE CP instruction. The change-direction and bracket indicators can be used with any message on the LU-LU session. Begin Bracket (BB) and End Bracket (EB) should only be set on first- or only-in-chain requests; Change Direction (CD) should only be set on last- or only-in-chain requests. Chaining and response protocol indicators (first-of-chain only) are used with Data or Data-and-Control (they are ignored for all commands). When writing a response, the controller ignores all chaining, bracket, and direction indicators.

Change-Direction Protocol

The change-direction (CD) indicator is used with the half-duplex flip-flop protocol, and optionally with the half-duplex contention protocol. A change-direction indicator tells the receiver that transmitting can begin.

For example, if all data transmission is initiated by the SLU it must begin by transmitting messages that completely describe a transaction. On the last message, the SLU must set the change-direction indicator to tell the PLU that it can begin transmitting a reply. If the PLU needs additional information to complete the transaction, it sends the inquiry and sets the change-direction indicator. The dialog proceeds in this half-duplex mode until the transaction completes. During a half-duplex dialog, the SLU can use the Signal message to tell the PLU to stop sending data, and change the direction of data flow. See Figure 2-16 on page 2-43 for an example of a change direction sequence.

Bracket Protocol

Brackets are an optional protocol that give the PLU and SLU context control of data transmission by indicating that a session concerns a single subject. Bracket protocol protects a current session from interruption by another session. The protected session is called a bracket.

The first message in the bracket contains a begin-bracket indicator, and the last message in the bracket contains an end-bracket indicator. A single message can be a bracket if it has both indicators.

For a bracketed session, the Bind parameters specify one of the LUs as the first speaker, the other as the bidder. The first speaker has the ability to begin a bracket without permission from the other LU. The bidder, however, must request and receive permission from the first speaker to begin a bracket.

Bid is a normal-flow request issued by the bidder to request permission to begin a bracket. A positive response to Bid indicates that the first speaker will not begin a bracket, but will wait for the bidder to begin a bracket.

A negative response to a Bid indicates that the first speaker has denied permission for the bidder to begin a bracket. A Ready to Receive (RTR) command may be sent later by the first speaker when permission to start a bracket is granted. If the first speaker will send RTR later, the sense data with the negative response to Bid is "Bracket Bid Reject-RTR Forthcoming". The bidder has the option of waiting for RTR or sending Bid again.

If the RTR will not be sent, the sense data is "Bracket Bid Reject—No RTR Forthcoming". In the latter case, the bidder must send Bid again if it still wants to begin a bracket.

Instead of sending Bid followed by first-in-chain with BB, the bidder may attempt to initiate a bracket by simply sending first-in-chain with BB. The first speaker grants the attempt (with a positive response) or refuses it (with a negative response indicating either Bracket Bid Reject-RTR Forthcoming or Bracket Bid Reject—No RTR Forthcoming). However, if the bidder terminates the chain that carries BB by sending Cancel, then, regardless of the response, the bracket is not initiated.

RTR may be issued by the first speaker to grant permission to the bidder to begin a bracket, or to find out if the bidder wants to begin a bracket.

A positive response to RTR indicates that the bidder will initiate the next bracket. If the bidder does not want to initiate a bracket, it issues a negative response with the sense code, "RTR Not Required".

For example, assume a transaction is being processed by the SLU. The first message sent to the PLU is accompanied by a begin-bracket indicator. During the dialog, a subroutine of the PLU is told to send a message to all SLUs noting that terminals will be shut down in 15 minutes. The subroutine sends a Bid message to all SLUs. The SLUs not in a bracket send a positive response, and they immediately receive the Shutdown message as a bracket (both begin-bracket and end-brackets indicators set); the SLU processing the transaction sends a negative response, which is noted by the subroutine. When the SLU sends the last message of the transaction, it includes an end bracket. The SLU then sends Ready to Receive, which is passed to the PLU subroutine; the subroutine then sends the Shutdown message. Figure 2-18 on page 2-45 shows message sequences using brackets.

Data-Chaining Protocol

Data chaining is an optional protocol used for transmitting a group of related messages. To send chained messages from the SLU, the program must set the message's chaining indicator to 1 to indicate the first message in a chain. For all messages between the first and the last in the chain the indicator is set to 0, and then to 1 again for the last message in the chain. The chaining indicator is also set to 0 to indicate an unchained message (a single message chain).

When the SLU is receiving messages, it must test the chaining indicator to determine whether the messages are chained or not. The status bits are set to indicate whether the message is first or last in a chain.

Only three types of chains may be sent:

- No-response chains: Each request in the chain is marked, "no response".
- Exception-response chains: Each request in the chain is marked, "exception response".
- Definite-response chains: The last request in the chain is marked, "definite response"; all other requests in the chain are marked, "exception response".

For example, assume that the diskette records all teller transactions for the day. At the end of the day, all records are compiled into a single file, but none of the work station's segments is large enough to assemble the entire file.

Instead, the work station sets the first record from the file into segment 3 and sends it as the first message in a chain. The work station reads the second record into segment 4 and transmits it as a middle-of-chain message. After checking the write operation from segment 3 for completion, the work station reads the third record into the segment and sends it as a middle-of-chain message, and so on.

This writing from alternate segments continues until end-of-file occurs; the work station then transmits either a message with a length of zero and the end-of-chain indication or sends the end-of-chain indication in the last message.

In another case, a work station receives a report to be printed on a 4710. The messages comprising each report page are chained. If an error occurs in the middle of a page, the work station sends a negative response to the message, and the VTAM application program sends a Cancel message to end the chain. The station positions the 4710 to a new page, and the VTAM application program resends the page as a new chain of messages.

For chained messages, an exception response protocol is typically used for all elements of the chain except the last. For the last element, a definite response protocol is typically used. When this procedure is used, the sender keeps the entire chain available, and retransmits it if necessary, until a positive response is returned.

When sending a message chain to the PLU, the SLU can send a Cancel command if either the SLU or the PLU finds a message error. The PLU should discard all messages in the chain that it has received. If the PLU sends a negative response to any element of a chain, the SLU should end the chain normally, or send Cancel.

Communication Sequences

The following figures show examples of the various types of dialogs that can occur between the SLU and the PLU. Figure 2-10 on page 2-36 is a flowchart of the processing performed by the SLU to initiate a session with a PLU. Figure 2-11 on page 2-37 through Figure 2-23 on page 2-50 are diagrams of communication sequences between the host and the controller.

The diagrams show the processing performed by the PLU, the controller, and the SLU. The diagrams also show the SMS fields that are set or tested by the SLU. The steps of each sequence are numbered to make them easier to follow.

Note: All responses shown are DR1 responses unless indicated otherwise.

Network Initialization and Session Initiation (Figure 2-10 on page 2-36). shows the steps performed by the SLU during network initialization and session initiation. Note that the resource name received with the Bind message is saved. This name is required if the SLU requests that the session be terminated. Note also that processing Set and Test Sequence Numbers (STSN) messages is represented as a user-defined process.

Network Initialization (Figure 2-11 on page 2-37). This diagram shows the network being initialized by the host. The controller passes the Ready indication to the SLU to indicate that communications with the host may begin.

Session Initiation by the SLU (Figure 2-12 on page 2-38). This diagram shows a session initiated by the SLU. The controller application program uses an LWRITE to send the Initiate message and the initiation data. The data consists of a header, the name of the PLU application program, 2 bytes of 0's, and an optional field which is passed to the PLU application program.

The optional field can contain passwords or any other information of use to the PLU during session initiation. (If the field is unused, 1 byte of 0's is required.) In step 6, the PLU application program issues the OPNDST macro. This results in a Bind message (containing the PLU application program name) being sent to the controller. When the controller receives Start Data Traffic (SDT), it sets the link status field at SMSCST, informing the SLU that session initiation is complete, and responds automatically to the SDT command.

Session Initiation by the PLU (Figure 2-13 on page 2-39). This diagram shows a session being initiated by the PLU. With the exception of the Initiate message, it is functionally similar to Figure 2-12.

Session Restart (Figure 2-14 on page 2-40). This diagram shows the variation of session initiation used for session restart. Before the SDT message is sent, the PLU sends an STSN message containing its version of the controller read and write sequence numbers. The SLU must make a positive response, but can include 5 bytes of data with the response which contain its version of the sequence numbers.

Data Flow (Figure 2-15 on page 2-41). This diagram shows the various response protocols that can be requested for Data messages. Part A shows data being sent with the no-response protocol indicated. Part B shows data being transmitted and a response being requested only if a failure occurs (exception response protocol). The response could be a negative DR1, DR2, or DR1 and DR2 response. If the data is received correctly, the receiver does not respond. Part C shows data being transmitted with the definite response protocol indicated. Both positive and negative responses are possible here.

Using Change Direction (Figure 2-16 on page 2-43). This diagram shows the change direction indicator in the message flag bytes to control the direction of data transfer. The change-direction indicator is used to tell the receiver that it may now begin sending.

Data Chaining (Figure 2-17 on page 2-44). This diagram shows how several messages are chained to indicate to the receiver that the entire unit of information is larger than the individual message.

One bit in the message flags indicator is used for chaining; the bit is set to 1 to indicate the beginning of the chain, set back to 0 to indicate the messages in the middle of the chain, and then set to 1 again to indicate the end of the chain. If chaining is not used, the bit is set to 0.

Part A shows a normal transmission using chaining. The first three messages ask for a response only if a failure occurs. The last message (end of chain) asks for an acknowledgment that the chain was received.

Part B shows the same transmission but with a failure during transmission of the second message. After sending the third message, the controller returns status to the SLU showing that a response is pending. The SLU reads the response, and sends a Cancel message.

Using Brackets (Figure 2-18 on page 2-45). This diagram shows how brackets separate data within a transmission. In part A, steps 1 through 5 show a transaction dialog. The PLU then finds that it has data unrelated to the transaction to send (such as a broadcast message) and sends a Bid message. Since the bracket has not ended, the SLU sends a negative response to the Bid message. When the bracket ends, the SLU sends a Ready to Receive message and then reads the unrelated data. If a bracket is not in progress, the sequence required to send the broadcast message would be that shown in Part B of the diagram.

Suspending Data Transfer (Figure 2-19 on page 2-46). This diagram shows how a data transfer may be suspended using the Quiesce at End-of- Chain message. The Release Quiesce message is used to inform the quiesced unit that data transmission can begin again.

Resynchronization and Recovery (Figure 2-20 on page 2-47). This diagram shows a recovery procedure that can be used when a network failure occurs. The SLU recognizes this problem when the status bits for a data check are set by the controller. As with session restart (Figure 2-14), the use of the STSN message is determined by the financial institution.

Orderly Termination of a Session (Figure 2-21 on page 2-48). This diagram shows an orderly termination of a session. A Chase message is used to ensure that all synchronous data has been received before processing is terminated.

Immediate Termination of a Session (Figure 2-22 on page 2-49 and Figure 2-23 on page 2-50). These diagrams show session termination. The sequence of messages starting with the CLSDST macro is required to terminate the session. When the Clear message is sent, all messages for this session that are in the network are purged.

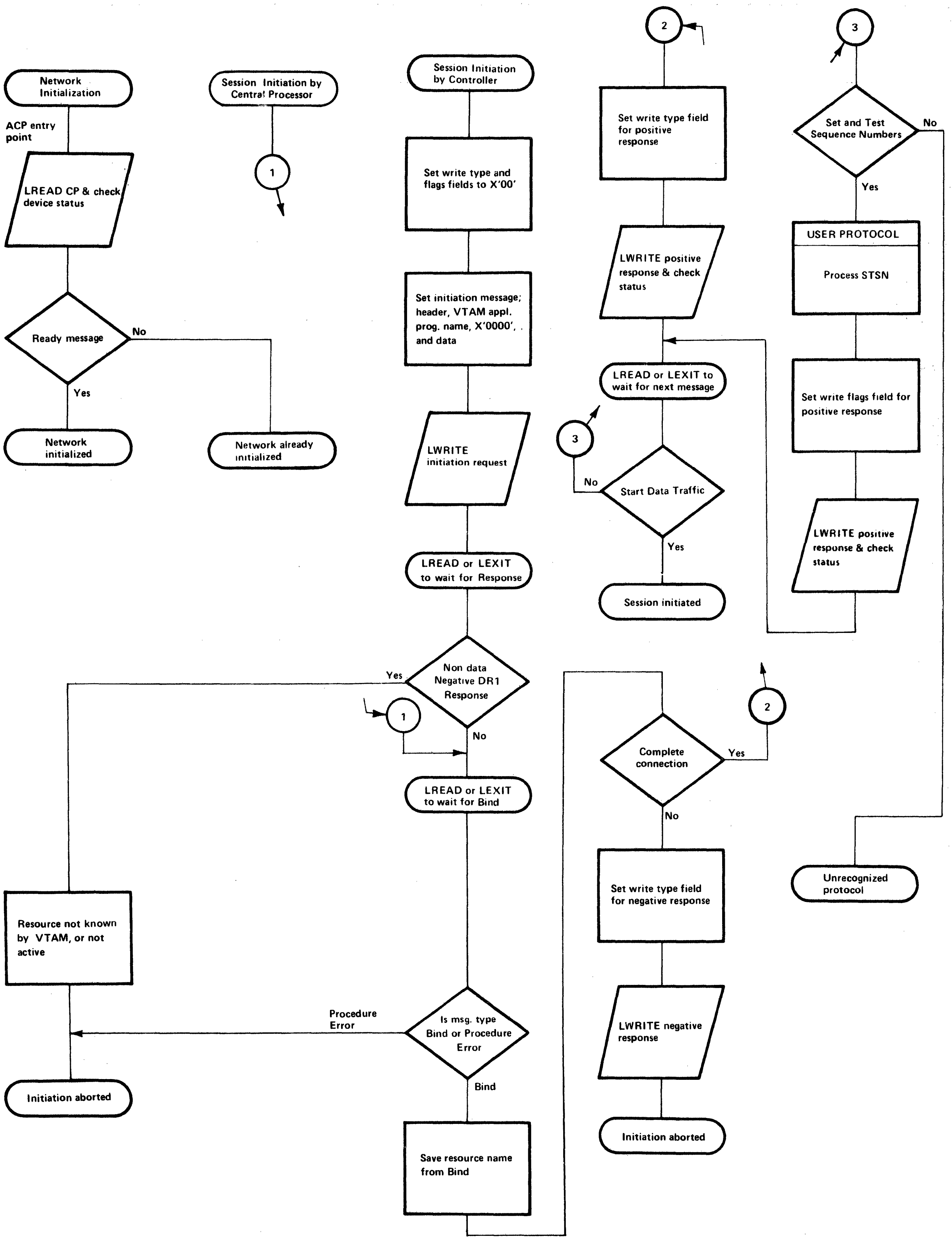


Figure 2-10. Network Initialization and Session Initiation by the SLU

VTAM Application Program	VTAM	Message Flow	Controller	Controller Application Program		Communication Status Field	Read Control Fields		Write Control Fields		Sequence Number Fields		
					Amount of Data	SMSCST X'40'	SMSCRF	SMSCRT	SMSCWF X'00'	SMSCWT X'00'	SMSCRS X'0000'	SMSCWS X'0000'	SMSCSR X'0000'
1	VARY ACTIVATE cluster or START VTAM	Activate PU →											
2	VARY ACTIVATE logical unit or START VTAM	Activate LU →	3 READY →	4 LREAD	None		X'00'	X'00'					

Figure 2-11. Network Initialization by the Host

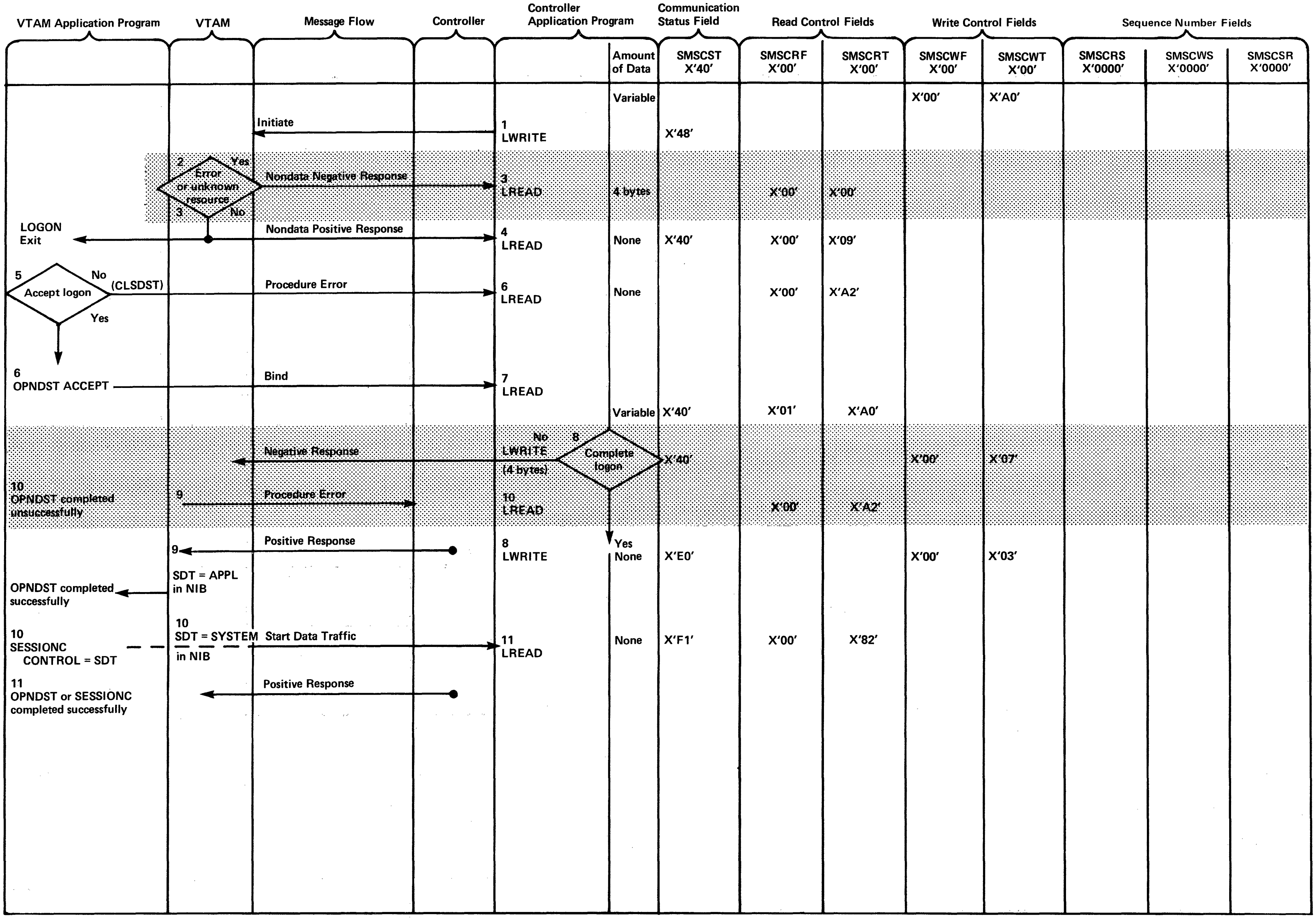


Figure 2-12. Session Initiation by the SLU

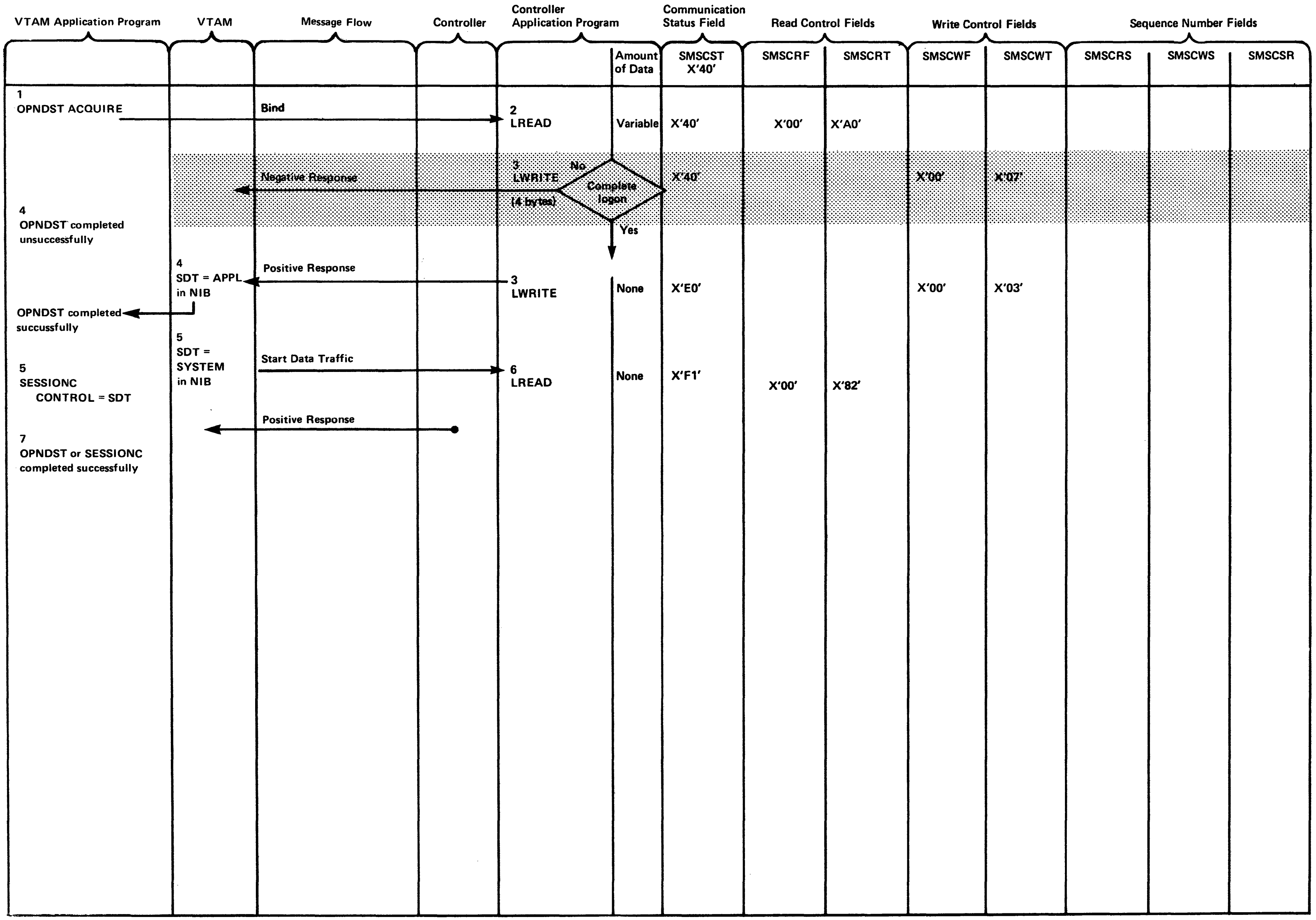
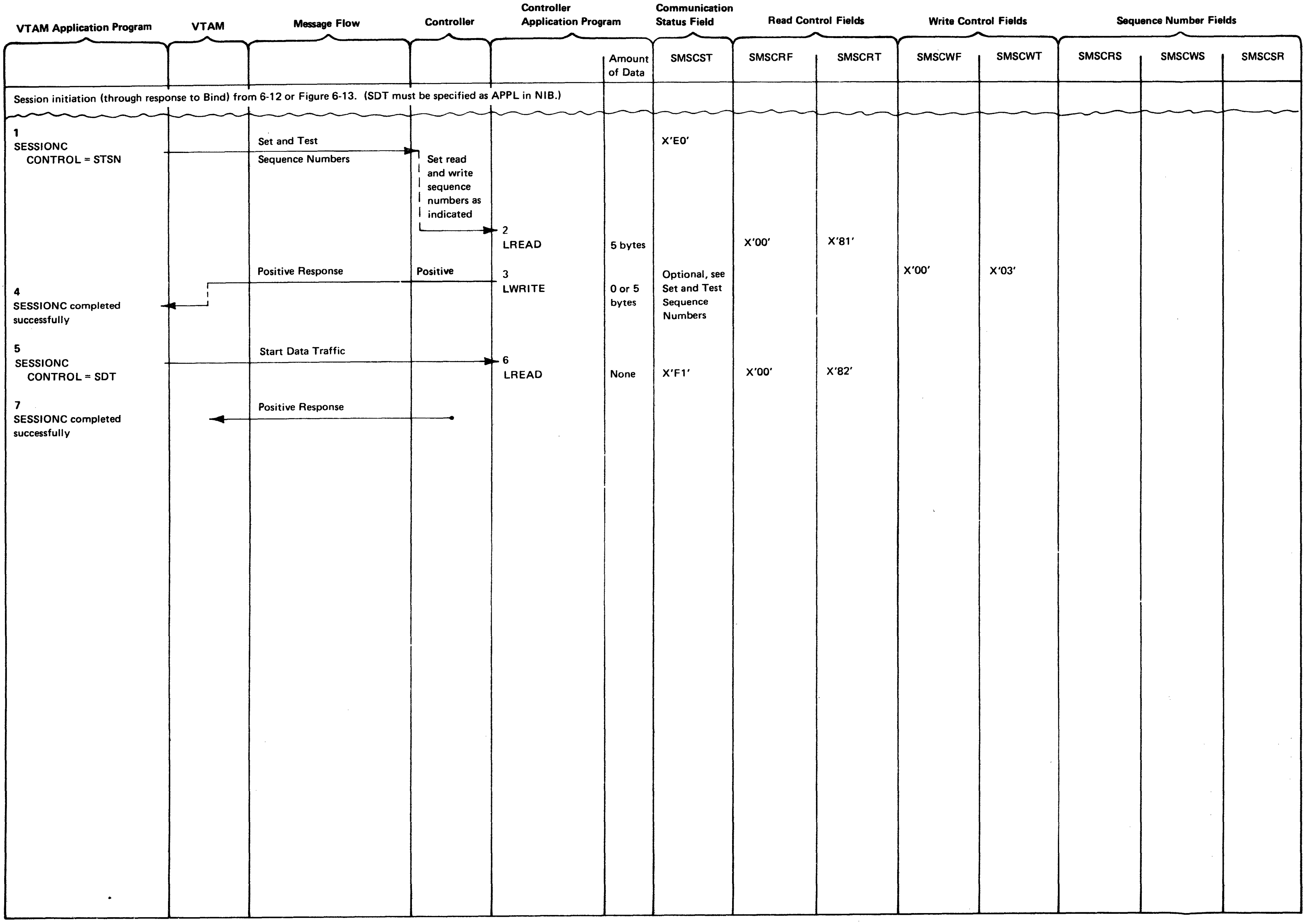


Figure 2-13. Session Initiation by the PLU

Figure 2-14. Session Restart



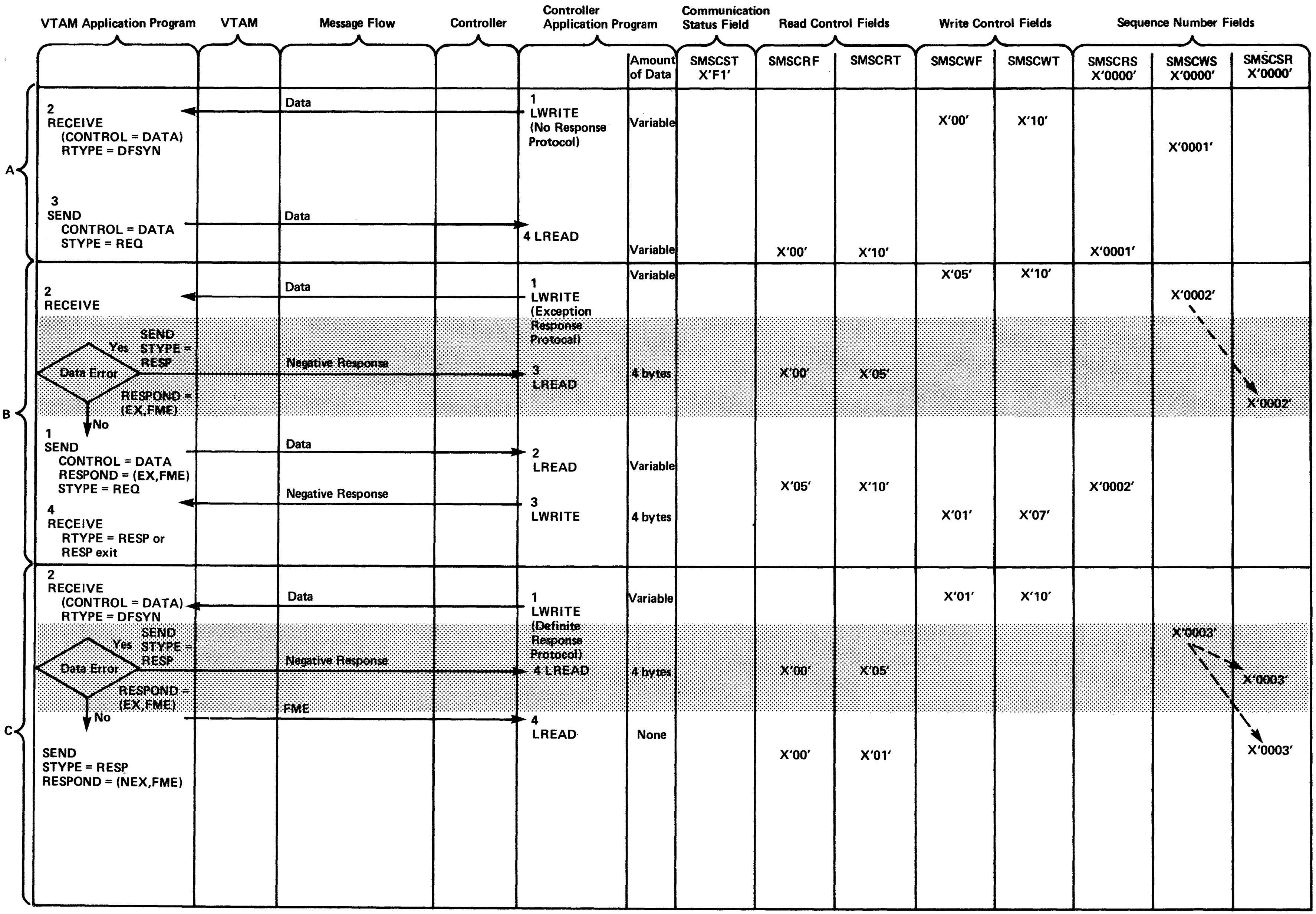


Figure 2-15 (Part 1 of 2). Data Flow with Sequence Numbers

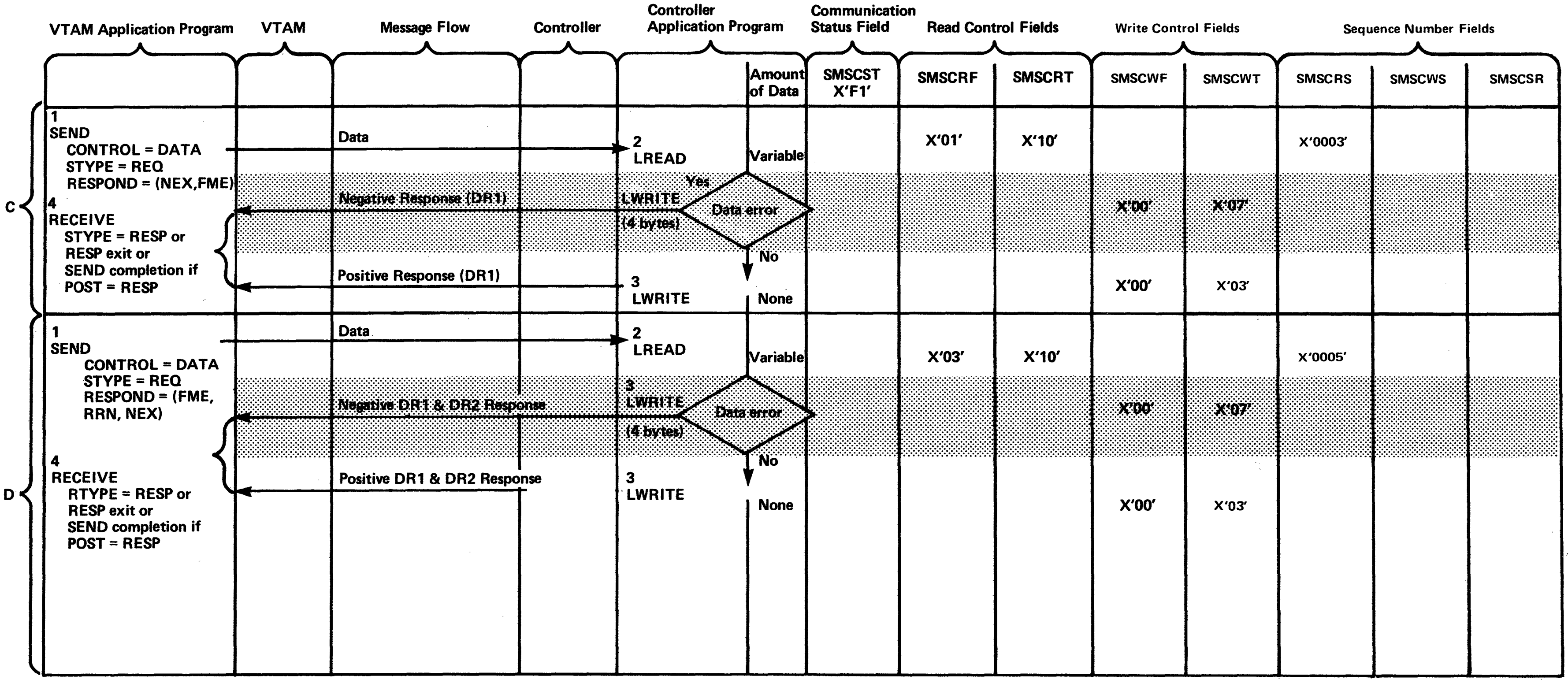


Figure 2-15 (Part 2 of 2). Data Flow with Sequence Numbers

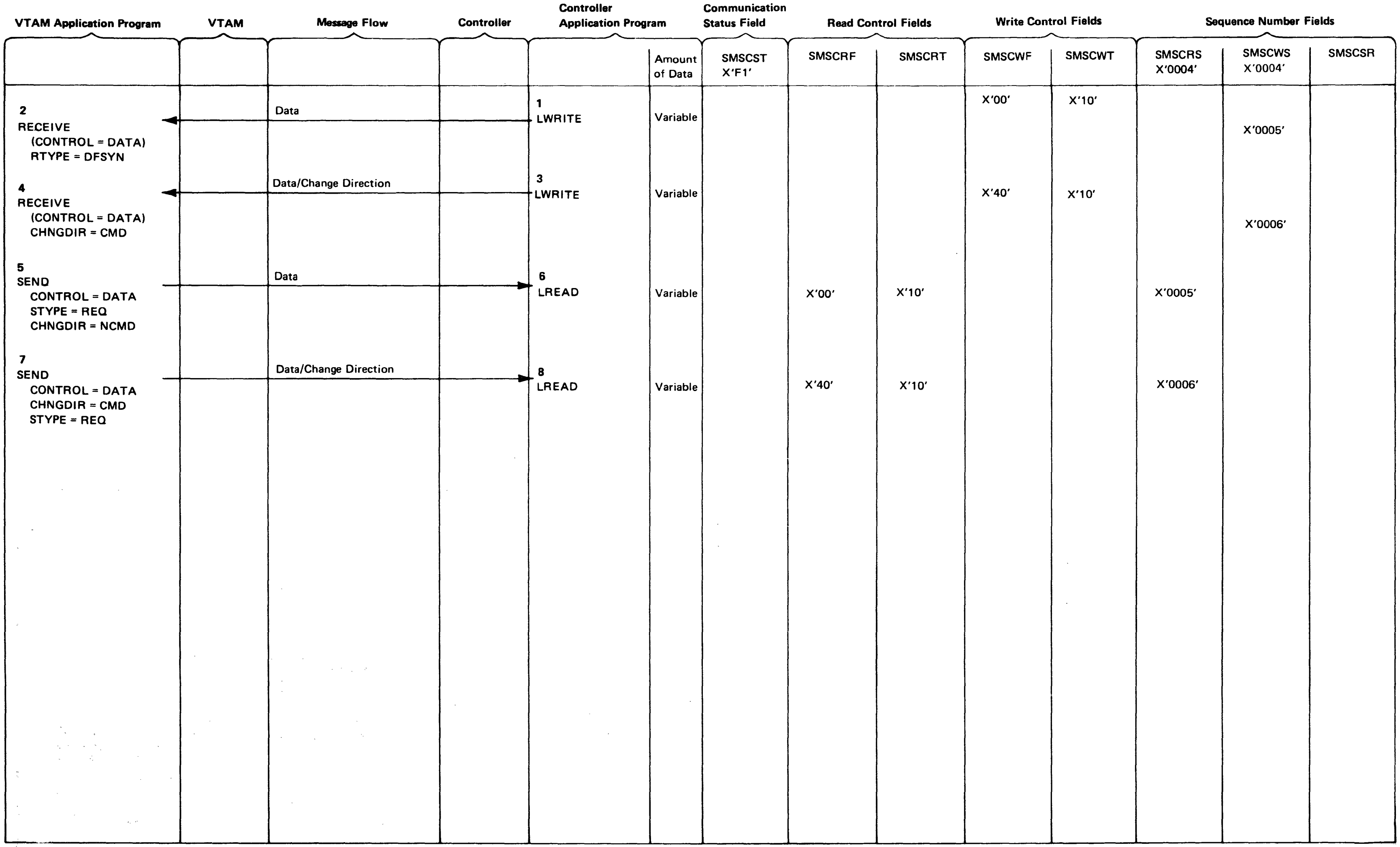
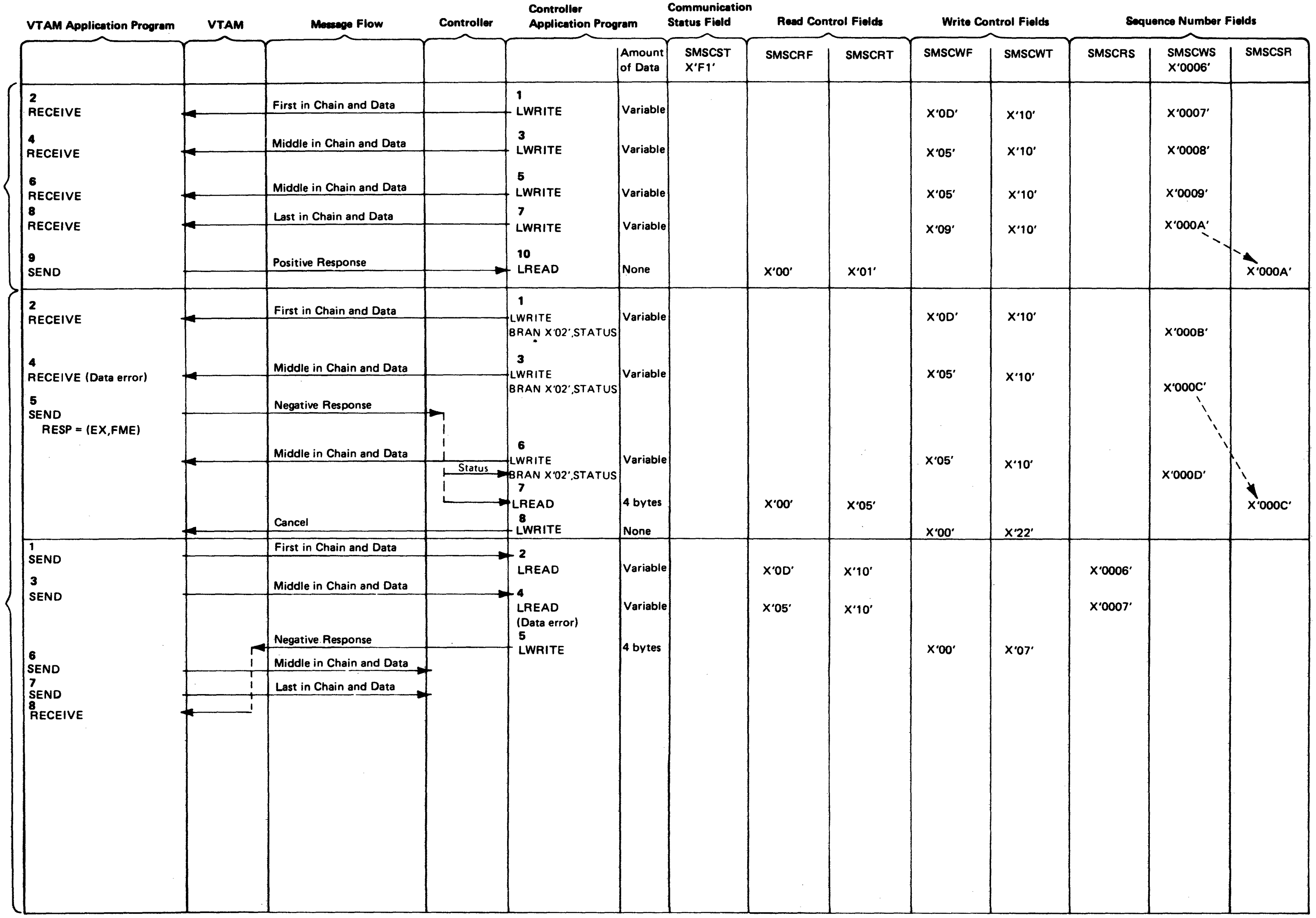


Figure 2-16. Using Change-Direction Protocol



	VTAM Application Program	VTAM	Message Flow	Controller	Controller Application Program	Amount of Data	Communication Status Field	Read Control Fields		Write Control Fields		Sequence Number Fields		
							SMSCST X'F1'	SMSCRF	SMSCRT	SMSCWF	SMSCWT	SMSCRS	SMSCWS X'0006'	SMSCSR
A	2 RECEIVE		First in Chain and Data		1 LWRITE	Variable				X'0D'	X'10'		X'0007'	
	4 RECEIVE		Middle in Chain and Data		3 LWRITE	Variable				X'05'	X'10'		X'0008'	
	6 RECEIVE		Middle in Chain and Data		5 LWRITE	Variable				X'05'	X'10'		X'0009'	
	8 RECEIVE		Last in Chain and Data		7 LWRITE	Variable				X'09'	X'10'		X'000A'	
	9 SEND		Positive Response		10 LREAD	None		X'00'	X'01'					X'000A'
B	2 RECEIVE		First in Chain and Data		1 LWRITE BRAN X'02',STATUS	Variable				X'0D'	X'10'		X'000B'	
	4 RECEIVE (Data error)		Middle in Chain and Data		3 LWRITE BRAN X'02',STATUS	Variable				X'05'	X'10'		X'000C'	
	5 SEND RESP = (EX,FME)		Negative Response											
			Middle in Chain and Data	Status	6 LWRITE BRAN X'02',STATUS	Variable				X'05'	X'10'		X'000D'	
					7 LREAD	4 bytes		X'00'	X'05'					X'000C'
			Cancel		8 LWRITE	None				X'00'	X'22'			
	1 SEND		First in Chain and Data		2 LREAD	Variable		X'0D'	X'10'			X'0006'		
	3 SEND		Middle in Chain and Data		4 LREAD (Data error)	Variable		X'05'	X'10'			X'0007'		
			Negative Response		5 LWRITE	4 bytes				X'00'	X'07'			
	6 SEND		Middle in Chain and Data											
	7 SEND		Last in Chain and Data											
	8 RECEIVE													

Figure 2-17. Using Data-Chaining Protocol

	VTAM Application Program	VTAM	Message Flow	Controller	Controller Application Program		Communication Status Field	Read Control Fields		Write Control Fields		Sequence Number Fields		
						Amount of Data	SMSCST X'F1'	SMSCRF	SMSCRT	SMSCWF	SMSCWT	SMSCSR X'0008'	SMSCWS X'000E'	SMSCRS
A	2 RECEIVE BRACKET = (BB,NEB)		← Begin Bracket and Data		1 LWRITE	Variable				X'20'	X'10'		X'000F'	
	3 SEND BRACKET = (NBB,NEB) STYPE = REQ		Data →		4 LREAD	Variable		X'00'	X'10'			X'0009'		
	6 SEND completed		← Data		5 LWRITE	Variable				X'00'	X'10'		X'0010'	
	7 SEND CONTROL = BID STYPE = REQ		Bid →		8 LREAD	None		X'00'	X'24'			X'000A'		
	10 SEND completed		← Negative Response		9 LWRITE	4 bytes				X'00'	X'07'			
	12 RECEIVE BRACKET = (EB,NBB)		← End Bracket and Data		11 LWRITE	Variable				X'10'	X'10'		X'0011'	
	14 RECEIVE CONTROL = RTR RTYPE = DFSYN		← Ready to Receive		13 LWRITE	None	X'F9'			X'00'	X'24'		X'0012'	
	15 SEND STYPE = RESP RESPOND = FME		Positive Response →		16 LREAD	None	X'F1'	X'00'	X'01'					
	17 SEND CONTROL = DATA BRACKET = (BB,EB)		Begin Bracket, End Bracket, and Data →		18 LREAD	Variable		X'30'	X'10'			X'000B'		
B	1 SEND CONTROL = BID STYPE = REQ		Bid →		2 LREAD	None		X'00'	X'24'			X'000C'		
	4 SEND completed		← Positive Response		3 LWRITE	None				X'00'	X'03'			
	5 SEND CONTROL = DATA BRACKET = (BB,NEB) STYPE = REQ		Begin Bracket and Data →		10 LREAD	Variable		X'20'	X'10'			X'000D'		

Figure 2-18. Using Bracket Protocol

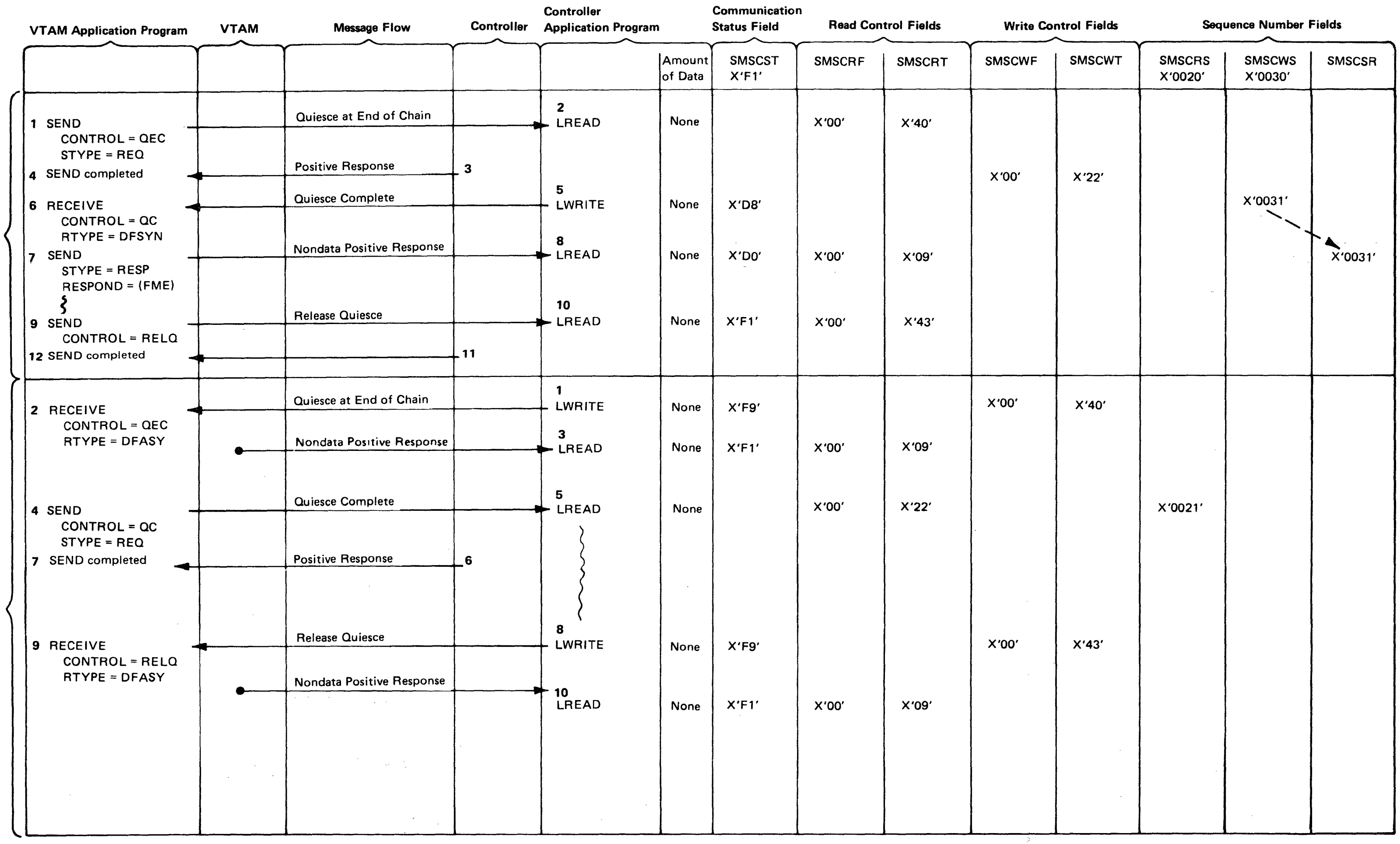
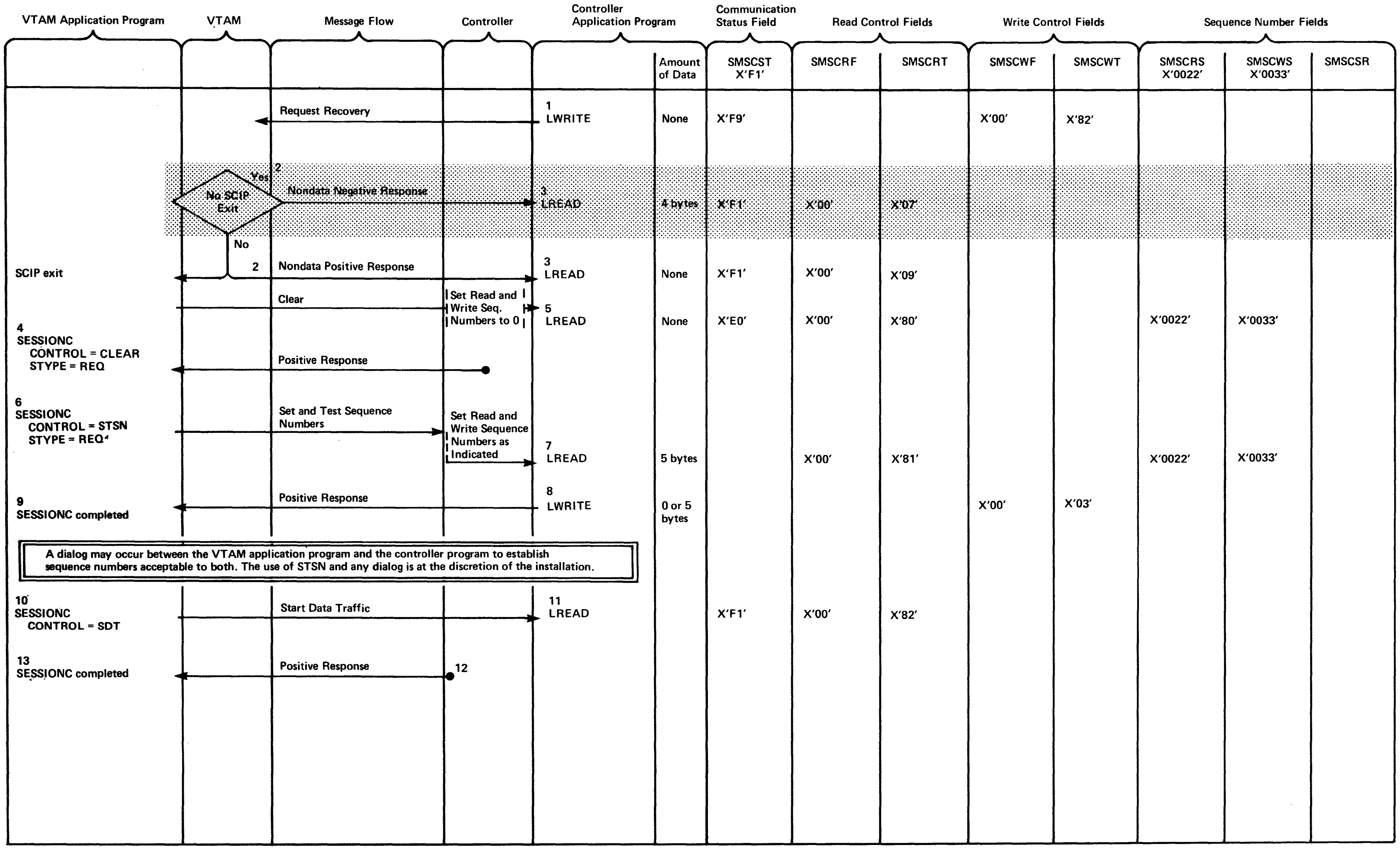


Figure 2-19. Suspending Data Transfer



A dialog may occur between the VTAM application program and the controller program to establish sequence numbers acceptable to both. The use of STSN and any dialog is at the discretion of the installation.

Figure 2-20. Resynchronization and Recovery

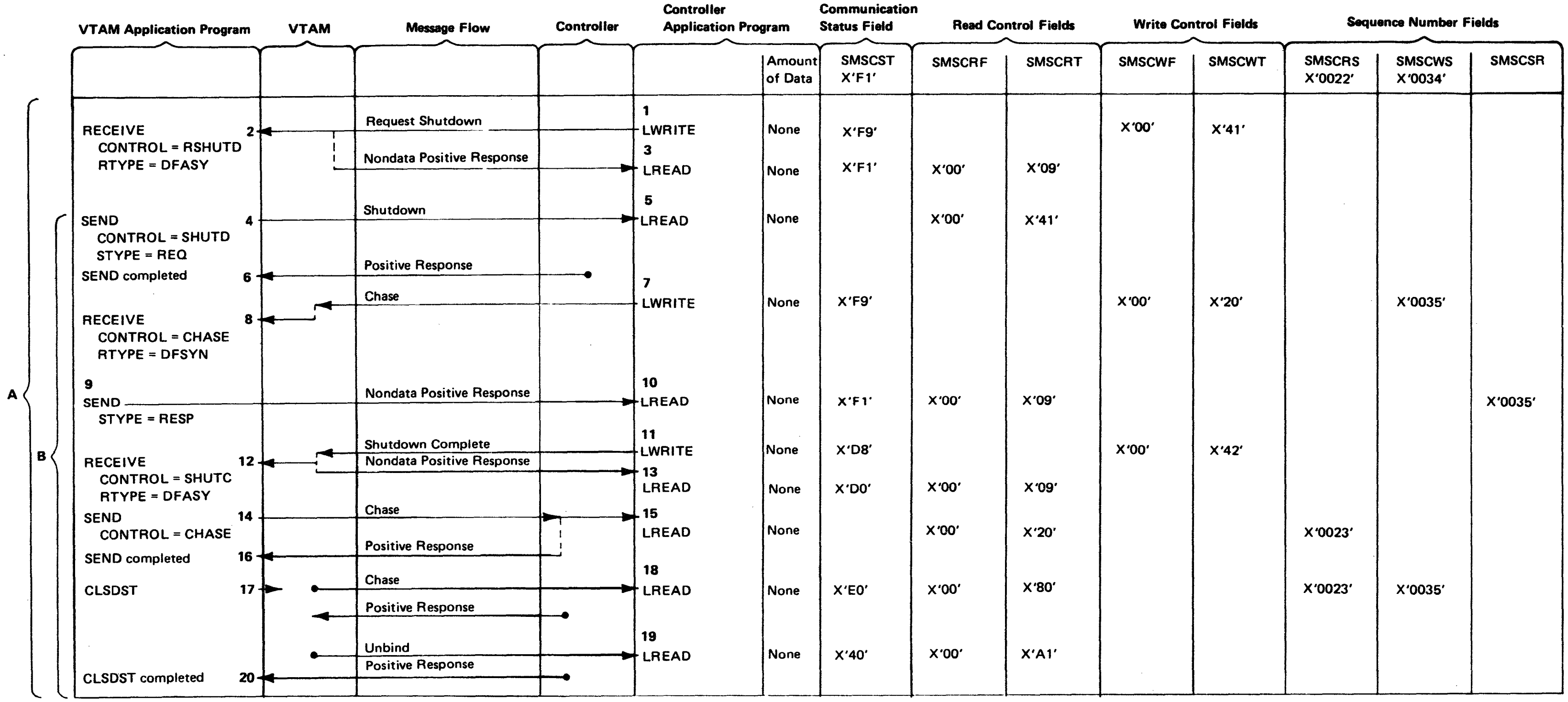
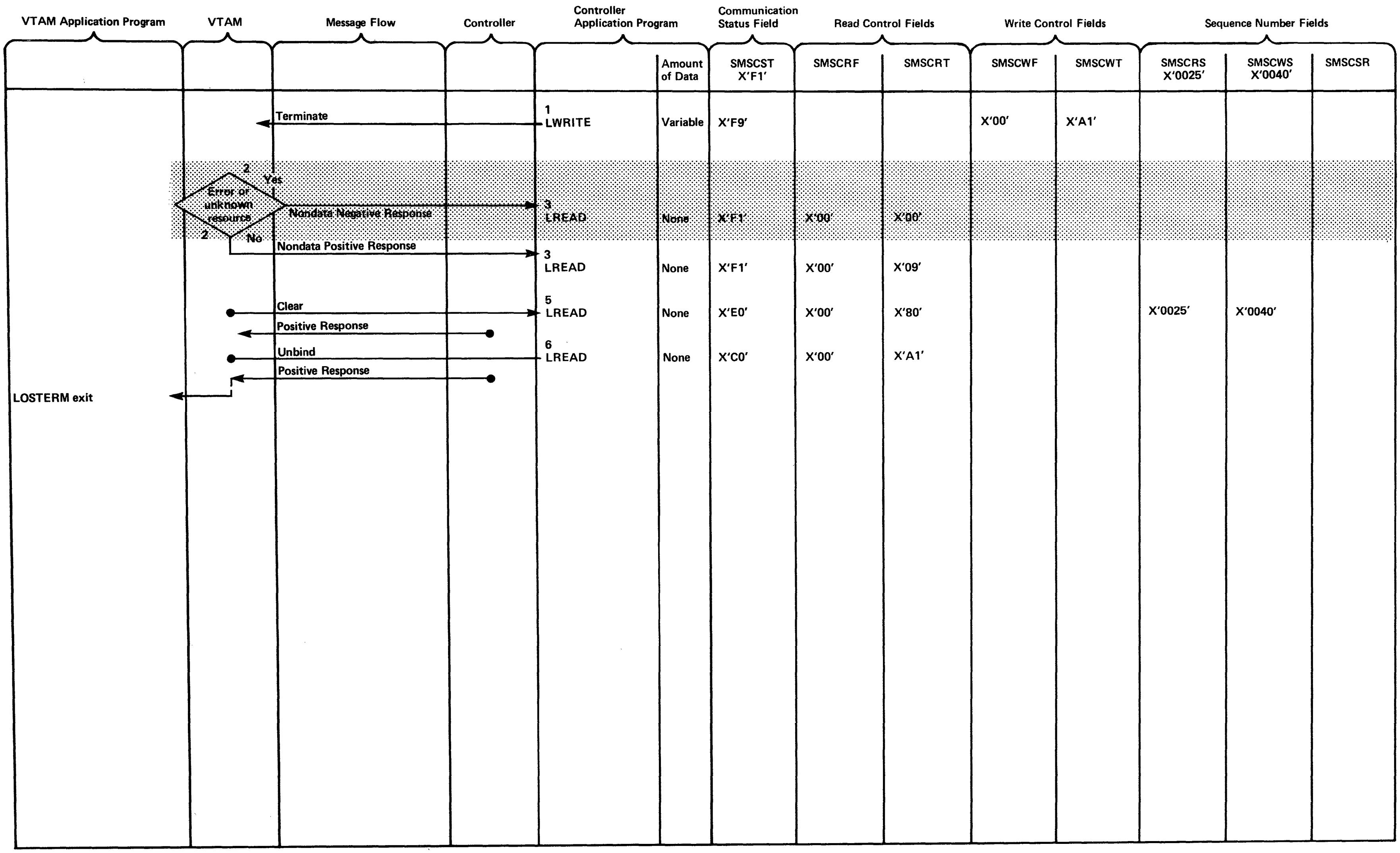


Figure 2-21. Orderly Ending of a Session



Controller Application Program	Amount of Data	Communication Status Field SMSCST X'F1'	Read Control Fields SMSCRF	 SMSCRT	Write Control Fields SMSCWF	 SMSCWT	Sequence Number Fields SMSCRS X'0025'	 SMSCWS X'0040'	 SMSCSR
1 LWRITE	Variable	X'F9'			X'00'	X'A1'			
3 LREAD	None	X'F1'	X'00'	X'00'					
3 LREAD	None	X'F1'	X'00'	X'09'					
5 LREAD	None	X'E0'	X'00'	X'80'			X'0025'	X'0040'	
6 LREAD	None	X'C0'	X'00'	X'A1'					

Figure 2-22. SLU-Initiated Session Ending

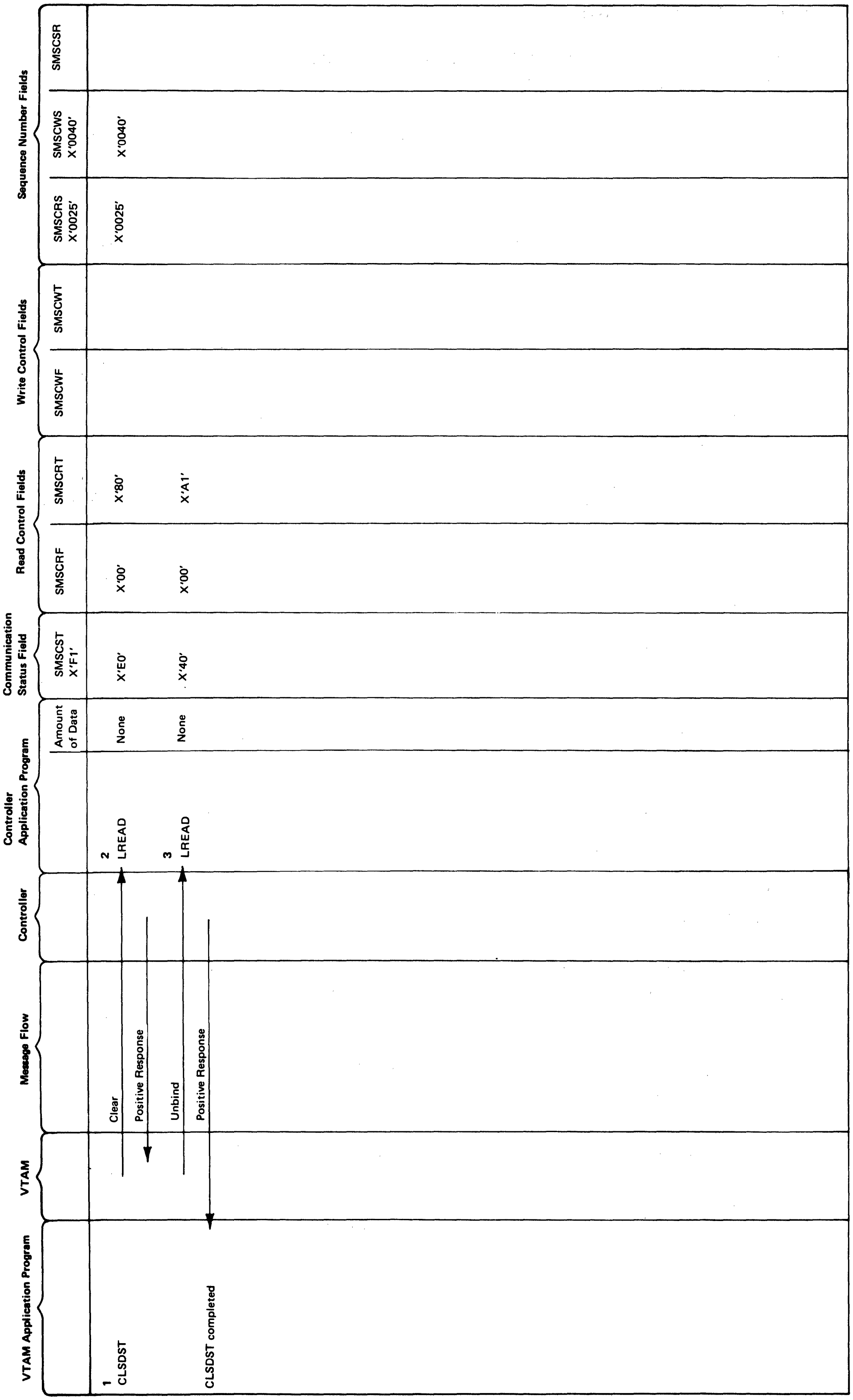


Figure 2-23. PLU-Initiated Session Ending

Controller Fields Used During Communication

The work station and controller use the following segment 1 (SMS) fields during host-controller communication:

The message length field (SMSIML): This 2-byte field contains the number of bytes read or the residual length after a write operation. This is the same field used for other I/O operation message lengths.

The device status field (SMSDST): This 2-byte field describes exceptional or error conditions when they occur. This is the same field used to report errors or exceptions during other I/O operations.

The read control field (SMSCRC): This 2-byte field describes the message received from the host. It is made up of the following two subfields:

> *The read flags field (SMSCRF)*: This 1-byte field describes the flags that accompanied the message received.
>
> *The read type field (SMSCRT)*: This 1-byte field describes the type of message received.

The write control field (SMSCWC): This 2-byte field is set by the SLU to describe the message being sent to the host. It is made up of the following two subfields:

> *The write flags field (SMSCWF)*: This 1-byte field specifies the flags that will accompany the message being sent.
>
> *The write type field (SMSCWT)*: This 1-byte field describes the type of message being sent.

The communication status field (SMSCST): This 1-byte field describes the SLU's status in the session.

The read sequence number field (SMSCRS): This 2-byte field contains the sequence number of any *normal* message received.

The write sequence number field (SMSCWS): This 2-byte field contains the sequence number of any *normal* message sent and is set by the controller when the LWRITE instruction is executed.

The response sequence number field (SMSCSR): This 2-byte field contains the sequence number of the response read.

Figure 2-24 through Figure 2-29 contain definitions of the fields used when communicating with the host system. Not shown are the sequence number fields (2 bytes each) which were discussed under "Sequence Numbering" on page 2-27. Figure 2-30 on page 2-57 shows the device status bits that may be set in SMSDST when communicating with the host system. Figure 2-31 on page 2-58 and Figure 2-32 on page 2-59 show the read and write types and possible responses.

Bit Position:	Meaning:
1nnn nnnn	Station in session
n1nn nnnn	Station configured to use the link
nn1n nnnn	Station not quiesced
nnn1 nnnn	Station not in data-flow-reset state
nnnn 1nnn	Nondata write in progress
nnnn n0nn	Reserved
nnnn nn0n	Reserved
nnnn nnn1	Data writes allowed

Figure 2-24. Link Status Definitions (SMSCST)

Bit Position:	Meaning:
01nn nnnn	Adapter enabled and contact established
11nn nnnn	Adapter enabled and contact not established
n0nn nnnn	Adapter disabled (contact flag ignored)

Figure 2-25. Adapter and Contact Flags (GMSIND)

Hex Value:	**Meaning:**
00	Ready
01	Positive DR1 response
02	Positive DR2 response
03	Positive DR1 & DR2 response
05	Negative DR1 response
06	Negative DR2 response
07	Negative DR1 & DR2 response
09	Positive (DR1) response on a nondata message**
0D	Negative (DR1) response on a nondata message**
10	Data
11	Data and control (e.g., FM header present)
20	Chase*
21	Cancel*
22	Quiesce complete (QC)*
23	LU status (LUSTAT)*
24	Bid (BID)
25	Bracket initiation stopped (BIS)*
40	Quiesce at end of chain (QEC)*
41	Shutdown (SHUTD)*
43	Release quiesce (RELQ)*
44	Signal (SIG)*)
45	Stop bracket initiation (SBI)*
80	Clear (CLEAR)*
81	Set and test sequence numbers (STSN)
82	Start data traffic (SDT)*
A0	Bind session
A1	Unbind session*
A2	Procedure error*
A3	Request maintenance statistics (REQMS)

* Response generated by the controller

**Received response caused by SNA including a 'request DR1 type response' with the last LWRITE.

Figure 2-26. Read Type Definitions (SMSCRT)

The following read control flags are in SMSCRF:

Response Protocol Requested (3 bits)

nnnn n000	No response
nnnn n001	Definite response protocol (DR1)
nnnn n010	Definite response protocol (DR2)
nnnn n011	Definite response protocol (DR1 & DR2)
nnnn n100	(Not allowed)
nnnn n101	Exception response protocol (DR1)
nnnn n110	Exception response protocol (DR2)
nnnn n111	Exception response protocol (DR1 & DR2)

Chaining Indicator (1 Bit)

nnnn 0nnn	Normal read or middle of chain
nnnn 1nnn	First or last in chain

Bracket, Direction, and Code Indicators (4 Bits)

nn00 nnnn	Middle of bracket
nn10 nnnn	Beginning of bracket
nn01 nnnn	End of bracket
nn11 nnnn	Only in bracket
n0nn nnnn	No change of direction
n1nn nnnn	Change direction indicator
0nnn nnnn	Not alternate code
1nnn nnnn	Alternate code

The following read control flags
are in extension field SMSCRE:

nn0n nnnn	Unpadded data
nn1n nnnn	Padded data
n0nn nnnn	Unexpedited message
n1nn nnnn	Expedited message
0nnn nnnn	Unenciphered data
1nnn nnnn	Enciphered data

Figure 2-27. Read Flags Definitions (SMSCRE/F)

Hex Value:	**Meaning:**
02	DR2 response
03	Positive response
07	Negative response
10	Data
11	Data and control (e.g., FM header present)
20	Chase*
21	Cancel*
22	Quiesce complete (QC)*
23	LU status (LUSTAT)*
24	Ready to receive (RTR)*
25	Bracket initiation stopped (BIS)*
40	Quiesce at end of chain (QRC)*
41	Request shutdown (RSHUTD)*
42	Shutdown complete (SHUTDC)*
43	Release quiesce (RELQ)*
44	Signal (SIG)*
45	Stop bracket initiation (SBI)*
82	Request recovery*
A0	Initiate session*
A1	Terminate session*
A3	Record formatted maintenance statistics (RECFMS)*

*A DR1 response is automatically requested.

Figure 2-28. Write Type Definitions (SMSCWT)

The following write control flags are in SMSCWF:

Response Protocol Requested (3 bits)

nnnn n000	No response protocol
nnnn n001	Definite response protocol (DR1)
nnnn n010	Definite response protocol (DR2)
nnnn n011	Definite response protocol (DR1 & DR2)
nnnn n100	Not allowed
nnnn n101	Exception response protocol (DR1)
nnnn n110	Exception response protocol (DR2)
nnnn n111	Exception response protocol (DR1 & DR2)

Chaining Indicator (1 Bit)

nnnn 0nnn	Normal write or middle of chain
nnnn 1nnn	First or last in chain

Bracket, Direction, and Code Indicators (4 Bits)

nnn1 nnnn	End of bracket
nn1n nnnn	Beginning of bracket
n1nn nnnn	Change direction
0nnn nnnn	Not alternate code
1nnn nnnn	Alternate code

The following write control flags are
in extension field SMSCWE:

nn0n nnnn	Unpadded data
nn1n nnnn	Padded data
0nnn nnnn	Unenciphered data
1nnn nnnn	Enciphered data

Figure 2-29. Write Flags Definitions (SMSCWE/F)

Device Status (Byte 1)

1nnn nnnn	Intervention required (write only)
n1nn nnnn	Unit exception
nn1n nnnn	Data check (read only)
nnn1 nnnn	Reserved
nnnn 1nnn	Attention (read only)
nnnn n1nn	Command reject
nnnn nn1n	Loss of contact
nnnn nnn1	Incorrect length (read only)

Device Status (Byte 2)

1nnn nnnn	Not ready (intervention required)
n1nn nnnn	Not in session (intervention required)
1nnn nnnn	First in chain (read unit exception)
n1nn nnnn	Last in chain (read unit exception)
nn1n nnnn	Begin bracket (read unit exception)
nnn1 nnnn	End bracket (read unit exception)
nnnn 1nnn	controller staging buffer too small (incorrect length)
nnnn n1nn	User buffer too small (incorrect length)
1nnn nnnn	Data pending (write unit exception)
n1nn nnnn	Response pending (write unit exception)
nn1n nnnn	Command pending (write unit exception)
1nnn nnnn	Station quiesced (command reject)
n1nn nnnn	Station in data-flow-reset state (command reject)
nn1n nnnn	Station not allowed access to link (command reject)
nnn1 nnnn	Invalid command sequence (command reject)
nnnn 1nnn	No application program asynchronous entry point (command reject)
nnnn nnn1	Dynamic recovery attempted and failed

Figure 2-30. Device Status Set During Host Communication

Message	Hex Code	Type	Response	Comments
DR2 response	02	–	None	Response type dependent on request received
Positive response	03	–	None	
Negative response	07	–	None	
Data	10	S	All	01/02/03/05/06/07 response possible (SMSCAT)
Data and control	11	S	All	01/02/03/05/06/07 response possible (SMSCAT)
Chase	20	S	09/0D	
Cancel	21	S	09/0D	For chaining only; indicates end of chain
Quiesce complete	22	S	09/0D	Quiesce state entered
LU status	23	S	09/0D	
Ready to Receive	24	S	09/0D	
Bracket initiation stopped	25	S	09	
Quiesce EOC	40	A	09	
Request shutdown	41	A	09	Request central processor to end session
Shutdown complete	42	A	09	Response to shutdown; quiesce state entered
Release quiesce	43	A	09	Resets quiesce state
Signal	44	A	09	
Stop bracket initiation	45	A	09	
Request recovery	82	SC	09	
Initiate	A0	XX	09/0D	Issued in ready state only
Terminate	A1	XX	09/0D	Issued in ready state only
Record statistics (RECFMS)	A3	S	09	

Notes:

Hex Code – Contents of write type field (SMSCWT)

Type –

XX = Independent data flow
SC = Session control
S = Normal data flow control
A = Expedited data flow control

Figure 2-31. Controller Write Types and Possible Responses

Message	Hex Code	Type	Response	Comments
Ready	00	XX	None	3601 application program can issue initiate
Positive DR1 response	01	–	None	Response type dependent on request sent
Positive DR2 response	02	–	None	Response type dependent on request sent
Positive DR1 & DR2 response	03	–	None	Response type dependent on request sent
Negative DR1 response	05	–	None	
Negative DR2 response	06	–	None	
Negative DR1 & DR2 response	07	–	None	Response type dependent on request sent
Nondata Positive (DR1) response	09	–		
Nondata Negative (DR1) response	0D	–		
Data	10	S	–	02/03/07 response allowed
Data and control	11	S	–	02/03/07 response allowed
Chase	20	S	None	
Cancel	21	S	None	For chaining only; indicates end of chain
Quiesce complete	22	S	None	
LU status	23	S	None	
Bid	24	S	03/07	07 if data is not to flow
Bracket initiation stopped	25	S	None	
Quiesce EOC	40	A	None	Quiesce complete requested
Shutdown	41	A	None	Shutdown complete requested
Reserved	42			
Release quiesce	43	A	None	Resets quiesce state
Signal	44	A	None	
Stop bracket initiation	45	A	None	
Clear	80	SC	None	Resets quiesce state and enters data flow reset state
Set/Test sequence numbers	81	SC	03	Bind/Clear previously received
Start data traffic	82	SC	None	Data flow reset is reset
Bind	A0	SC	03/07	User must respond; if response is 03 in-session and data-flow-reset states are entered
Unbind	A1	SC	None	Resets in-session state
Procedure error	A2	XX	None	Initiate failure
Request statistics (REQMS)	A3	S	03/07	

Notes:

Hex Code – Contents of read flags field (SMSCRF)

Type –

XX = Independent data flow
SC = Session control
S = Normal data flow control
A = Expedited data flow control

Figure 2-32. Controller Read Types and Possible Responses

Controlling the Communication Link

STPLNK and STRLNK control the operation of the communication link during error recovery. The STPLNK instruction stops a running link. The STRLNK instruction activates a stopped link, or initiates a link wrap test.

STRLNK refers to a parameter list describing the manner in which the controller is connected to the telecommunications line. The parameter list can also contain a selection sequence field for connecting with a switched network requiring X.21 protocols. The parameters set with STRLNK are effective until controller is reloaded or the link is stopped again. The parameter list defined by the previous STRLNK or during controller configuration is used if no new parameters are specified.

The start link function is also available in the system monitor, so that these instructions do not have to be used in the application program. Both STRLNK and the start-link function of the system monitor can be used to override some of the parameters specified in the COMLINK configuration macro.

The following features are described by STRLNK:

- NRZI and non-NRZI data encoding: The type of data encoding used by the modems during data transmission. Specify NRZI encoding unless NRZI it is not compatible with the modem being used.

- Wrappable or non-wrappable: Whether the modem is wrappable. Wrap capability is a feature used to diagnose failures. It effectively bypasses the telecommunications line so that the failure can be isolated to the modem or the link.

- Speed selection: Whether the modem is capable of operating at two speeds. Some modems can operate at two different transmission speeds (for example, 2400 and 4800 bps). One of these bits is set to indicate whether the higher or lower speed is being used. If it is a single-speed modem, specify that the higher speed is being used.

- Switched or nonswitched: Whether the modem is attached to a switched or nonswitched communication line.

- Control unit address.

- Data terminal ready: Used to indicate to the modem that the controller is ready. Operation cannot continue until the modem returns data-set-ready to the controller.

- Permanent request to transmit: Used when the controller and 3704 or 3705 are attached by a full-duplex point-to-point line. Not effective if switched lines are being used.

- Controller request to transmit: Used when the controller and 3704 or 3705 are attached by a half-duplex point-to-point line, or the controller is part of a multipoint network.

- XID: The transmission identifier (a 20-bit binary number) that identifies this controller on the switched network. This ID must correspond with the ID specified for this controller during VTAM system definition (the IDNUM operand of the PU macro definition statement). The BLKID for 4700 is X'57'.

For X.21 switched networks, the parameter list defines the following:

- The network features to be used (autoanswer, autocall, or direct call).
- The selection sequence field length and content.

The X.21 version (VERSION), retry (RETRIES), and delay (DEL1 and DEL2) parameters specified by the COMLINK macro cannot be overridden by the STRLNK parameter list.

Sense Codes

Every negative response must contain by four bytes of data; either sense information provided by the application program, or four bytes of 0's if the sense information is not used. The first two bytes are network-defined and have the meanings described in Figure 2-33. The last two bytes can contain any information defined by the financial institution.

All valid sense codes are listed in Figure 2-33; all other combinations are reserved. The controller always sets bit 0, of sense data byte 0 from the application program to 0. With the exception of those codes listed as being sent by the controller, all codes are sent by the controller application program or the VTAM application program.

Figure 2-33 lists a sense code, an explanation, and an action code. The action code is a number that is referenced in Figure 2-34 on page 2-66 along with suggested actions.

Every LU status message must also be accompanied by 4 bytes of data. The first 2 bytes must be X'0000'; the last two bytes are defined by your financial institution.

Sense Code	Explanation	Action Code
X'4000'	Protocol error: The requested protocol (response, chaining, bracket, or change direction) is not supported: This may be sent rather than a more definitive sense code.	2
X'4003'	Begin bracket not allowed: The begin bracket was set along with middle–of–chain or end–of–chain.	2
X'4004'	End bracket not allowed: The end bracket was set along with middle–of–chain, or sent by the host when only the controller may send end bracket.	2

Figure 2-33 (Part 1 of 5). Communication Sense Codes and Explanations

Sense Code	Explanation	Action Code
X'4006'	Exception not supported: An exception response was requested, but is not supported.	2
X'4007'	Definite response not supported: A definite response was requested (DR1 or DR2) but is not supported.	2
X'4009'	Change direction not allowed: Change direction was set along with first–in–chain or middle–in–chain.	2
X'400A'	No–Response not allowed: No–Response was specified in the message, but this protocol is not allowed.	2
X'400B'	Multiple element chaining not supported: A chained message was received, but chaining is not supported.	2
X'400C'	Brackets not supported: A begin bracket or end bracket was received, but bracket protocol is not supported.	2
X'400D'	Change direction not supported: Change direction was requested, but is not supported.	2
X'400F'	Data–end–control not allowed. Data–end–control is not supported in this session or is not supported when set with first–in–chain.	2
X'2000'	Protocol used incorrectly. Session initiation, quiesce, sequence number, bracket, chaining, or change direction protocol used incorrectly. This may be sent rather than a more definitive sense code.	1 3 4
X'2001'	Sequence number error: The sequence number of the last message received was not the next sequential sequence number: a message has been lost in the network. This code may be received by the controller application program, but is sent by the controller.	1
X'2002'	Chaining error: The chaining indicators were not in the first–in–chain, middle–in–chain, last–in–chain sequence.	3
X'2003'	Bracket error: The rules for bracket protocol were not followed.	3
X'2004'	Direction error: The half–duplex protocol established by change direction was not followed.	3

Figure 2-33 (Part 2 of 5). Communication Sense Codes and Explanations

Sense Code	Explanation	Action Code
X'2005'	Data traffic not started: A CLEAR or BIND was sent and then a message other than START DATA TRAFFIC.	4
X'2006'	Quiesce error: A normal message has been received after a QUIESCE COMPLETE and before a response has been sent for a RELEASE QUIESCE.	3
X'1000'	Message type or data error: A message has been received that is not valid type or contains invalid data. This may be sent rather than a more definitive sense code.	5
X'1001'	Data error: The data contains unrecognized character codes or is not formatted properly.	5
X'1002'	Data length error: The message is too long or too short.	5
X'1003'	Invalid function: The message references an unrecognized function.	5
X'1005'	Invalid parameters: A message, such as BIND or SIGNAL, contains parameters not recognized by the receiver.	5
X'1008'	Invalid FM header: The FM header is not understood or translatable by the receiver, or an FM header is expected but not present.	5
X'0800'	Requested function cannot be completed. This may be sent rather than a more definitive sense code.	3 5 6
X'0801'	Resource not available: The VTAM application program referenced in the INITIATE message is not started.	9
X'0802'	Intervention required: The requested output terminal is not available.	1
X'0803'	Missing password: The expected security password was not supplied.	5
X'0804'	Invalid password: The security password is incorrect or does not match the requestor ID.	5

Figure 2-33 (Part 3 of 5). Communication Sense Codes and Explanations

Sense Code	Explanation	Action Code
X'0805'	Session limit exceeded: The requested session cannot be bound since either the logical work station or VTAM application program has reached its session limit.	1
X'0806'	Resource unknown: The VTAM application program referenced in the INITIATE message is not known to VTAM.	5
X'0809'	State inconsistency: A requested function cannot be performed because of the current state of the receiver.	5
X'080A'	Permission rejected: An implied or explicit request has been rejected.	5
X'080C'	Procedure not supported: A procedure (dump, resynchronization, test, or other) is not supported.	5
X'080E'	Not authorized: The requestor is not allowed access to the requested resource.	6
X'080F'	End user not authorized: The requesting end user does not have access to the requested resource.	6
X'0810'	Missing requestor ID: The requestor identification was missing.	5
X'0811'	Break: Tells the receiver to terminate the current chain with CANCEL or last-in-chain.	8
X'0812'	Insufficient resources: Sufficient resources are not available to act on the request.	9
X'0813'	Bracket contention: BID or begin bracket received while in a bracket, READY TO RECEIVE will not be sent. Permission to begin a bracket has been denied.	7
X'0814'	Bracket contention: BID or begin bracket received while in a bracket, READY TO RECEIVE will not be sent. Permission to begin a bracket has been denied.	7
X'0819'	READY TO RECEIVE not required: The VTAM application program received READY TO RECEIVE but has nothing to send.	8

Figure 2-33 (Part 4 of 5). Communication Sense Codes and Explanations

Sense Code	Explanation	Action Code
X'081A'	Message sequence error: An invalid sequence of messages was received.	3
X'081C'	Function not executable: The requested function is supported but cannot be executed at this time.	3
X'0822'	Link procedure failure: A link-level procedure has failed due to equipment failure, loss of contact with a link station, or an invalid response to a link command.	1
X'0824'	Component aborted: The LU selected has been aborted due to an error condition or depletion of resources.	6
X'0825'	Component not available: The LU component is not available.	9
X'0826'	Function not supported: The function requested in the message is not supported.	5
X'0827'	Intermittent error, retry requested: The message was lost by the receiver. The failure has been corrected and the message (or chain) should be re-sent.	9
X'0828'	Reply not allowed: A reply has been requested, but the receiver is quiesced or shut down and has no delayed-reply capability.	5

Figure 2-33 (Part 5 of 5). Communication Sense Codes and Explanations

Action Code	Suggested Actions
1	Inform the network operator that the failure occurred.
2	Be sure that the session was established with the correct VTAM application program, or change either the SLU or VTAM application program to establish a consistent protocol.
3	Restart the sequence in which the failure occurred or correct the application program.
4	The VTAM application program should issue a START DATA TRAFFIC or the programming error should be corrected.
5	Revise or reformat the data or message type and retransmit.
6	Terminate the session and correct the application program.
7	Wait for the operation to complete.
8	Stop the operation and wait for further information.
9	Retry the operation.

Figure 2-34. Suggested Sense Code Actions

The following listed sense codes are not shown in Figure 2-33 on page 2-61, because they are sent between components in the SNA network and appear only in the system log.

Sense Code:	Explanation:
X'8000'	Path error. The received message cannot to delivered to a work station. The errors indicate network or hardware problems.
X'8001'	Intermediate node failure. A machine or program check occurred in an intermediate node. The message is discarded.
X'8002'	Link failure. The data link failed.
X'8003'	Logical unit inoperative. The specified logical unit has not been enabled for the link.
X'8004'	Invalid destination address field. The specified logical unit does not exist.
X'8005'	No session. No logical unit to logical unit session exists between the sender and receiver.
X'8006'	Invalid format identification. The received transmission header was not of type 2.
X'8007'	Mapping field error. Segments have been lost or have arrived in improper order.

Sense Code:	**Explanation:**
X‘8008’	Physical unit inactive. No activate physical unit signal has been received.
X‘8009’	LU inactive. No Activate Logical Unit has been received.
X‘800B’	Incomplete transmission header. The message is too short to contain a complete transmission header.
X‘800C’	Invalid data count field. The data count field is inconsistent with the transmission length.
X‘800D’	Lost contact. Contact with the receiver has been lost, but the link has not failed.
X‘4005’	Incomplete request header. The transmission was shorter than a complete transmission header/request header.

Chapter 3. Programming the Alternate SNA/SDLC Line (ALA)

The SNA/SDLC alternate (ALA) line allows an IBM 4701 controller work station to operate as an SNA primary logical unit (PLU) when attached to an ALA device, which operates as an ALA logical unit (SLU). This chapter explains how you control communication between the controller work station and the ALA device. The ALA device can be a 4730 Personal Banking Machine, or it can be a device attached via RPQ 820132.

Note: 4700 system users of the RPQ 8V0132, Systems Network Architecture—Primary (SNA-Primary) attachment, should refer to this *4700 Controller Programming Library* and the *4700 Subsystem Operating Procedures* rather than to the *Systems Network Architecture—Primary Custom Feature Description* manual, GC31-2509, which is now obsolete. Information in this manual that refers to the ALA link applies to the SNA-Primary link and its attached SLU devices, as well.

The ALA link, also called the alternate link in this chapter and the associated instruction descriptions, connects an SNA/SDLC-attached device to the 4701 controller as an SNA secondary logical unit (SLU). The 4701 controller operates as a primary logical unit (PLU) when communicating with the SLU, as does the host system when communicating with the 4701.

The 4701 controller controls sessions with the SLU device in much the same way as the host processor controls sessions with the 4701 over the host link.

System Structure

Although the ALA link supports both Type 1 and Type 2 Physical Units (PUs), the 4701 controller and the ALA device are PU-Type 2 units. You must define the PU type during configuration. The ALA link provides the following support:

- PU-Types 1 and 2 attachment (defined during configuration)
- Transmission Subsystem (TS) Profile 2, 3, 4, or 7 (defined when the session becomes active)
- Function Management (FM) Profile 2, 3, 4, 7, or 18 (defined when the session becomes active).

Network Linkage

The ALA link uses the SDLC (Synchronous Data Link Control) protocol. Additional characteristics of ALA link communication include:

- Locally-attached (fan-out) or remotely-attached (through modems)
- Multipoint
- Nonswitched
- Half-duplex

Work Stations and Logical Units

A 4700 logical work station comprises an application program and an allotted portion of controller storage. Each logical work station is an SNA *logical unit* (LU).

The ALA link permits attaching several types of logical units. The logical unit type—which defines such things as valid commands, use of brackets, use of function management headers, and so on— is not defined at CPGEN time. It is dynamically established during the LU-LU session between the two logical units.

Normally, the controller attached to the host processor is the ALA logical unit (SLU), and the host processor acts as the primary logical unit (PLU). However, when a device is attached to the host-connected controller, the device assumes the role as SLU whenever the two are communicating; the host-attached controller becomes the PLU whenever it communicates with the device.

Several types of LUs are possible within SNA. In the case of the ALA link, LU types 0, 1, 2, 3, 4, and 6 can be supported. An application program determines the LU type dynamically by specifying certain parameters when your program begins a controller—ALA device LU-LU session. It follows, then, that your program must support the proper sequence of commands, requests, replies, and options for the LU type that it specifies. You, therefore, must understand SNA before you program the ALA link.

ALA Link Terminology

Before continuing, you should understand the following definitions as they apply to the ALA link:

SSCP (System Service Control Point): That portion of the SSCP function provided by the link that is responsible for the SSCP-PU and SSCP-LU sessions, including their activation and deactivation.

SSCP/Application Program: That portion of the SSCP function that is in the application program. This includes support for unformatted data on the SSCP-LU sessions.

PU (Physical Unit): That portion of each supported node responsible for the SSCP-PU session.

PLU (Primary Logical Unit): That portion of your program used to support the LU-LU session flow.

SLU (Secondary Logical Unit): That portion of a supported node responsible for an SSCP-LU and an LU-LU session. Multiple SLUs can reside in each supported node.

Application Program: Your application program code containing both the SSCP/application program and PLU functions. A program can be unique to a 4700 station or can be shared by many 4700 stations.

SEC: An ALA SDLC station residing in a supported PU-T1/2 node and supporting the exchange of traffic with the PU and the SLUs within the node.

Ownership of Network Components

An ALA logical unit (SLU) can be connected logically to one or more logical work stations in the PLU; this is referred to as "ownership" of the SLU by the work station. Ownership of an SLU is determined during configuration of the ALA link network and attached devices. However, the link itself cannot be owned.

Programming Considerations

Work stations in a 4700 system communicate with the ALA device. However, there is a requirement that the application program contain some of the details of the protocol for those control units and devices. The path between the 4700 work station and the ALA device comprises the following components (some are equipment, others are programming):

- An application program that operates on behalf of one or more work stations.
- The link controller data that defines the required protocol and is used by one or more work stations.
- A nonswitched telecommunication line comprising:
    1. An SDLC half-duplex telecommunication adapter
    2. An EIA interface to a CCITT local interface
    3. A 2- or 4-wire telecommunication line
    4. Modems (if remote attachment).

A control unit, a control unit/terminal configuration, or a stand-alone terminal that operates with SDLC protocol.

For either 2- or 4-wire telecommunication lines, the controller data supports half-duplex mode only. Contact your IBM representative for compatibility information.

SDLC Protocol

The supported protocol is an IBM SDLC local line multipoint protocol. *Multipoint* refers to the control of the telecommunication line by polling sequences.

The interface to the ALA SDLC stations (not to be confused with the controller work station) is a half-duplex interface. This implies that the link can either send or receive data, but not both, simultaneously. Only one I-Frame can be sent to an SDLC station before a confirmation is requested. This is done by setting the poll flag in the I-Frame before transmission. When the SDLC station responds with either an I-Frame or SDLC response, acceptance of the message is determined.

ALA Link SNA Principles

A good familiarity with SNA principles is necessary when programming the alternate link:

Network Definition: In some of the macro instructions and statements used for network definition and application programming, you describe the physical and logical capabilities of the subsystem in SNA terms. You can also specify certain actions that are to occur when the network is activated and deactivated, again in SNA terms.

Application Programming: The portion of SNA dealing with establishing sessions and exchanging data has a significant impact on the design of application programs (both those in the controller and in the attaching unit).

The Systems Network Architecture publications listed in the preface provide a detailed description of SNA principles and the protocols that can exist in a network containing SNA communication products. However, these publications do not describe the protocols that each specific SNA product supports; refer to the product's component description for this information. This manual section describes the subset of SNA protocols used in the 4700 system to support the ALA link. You must understand these protocols to perform network definition and write application programs.

With the ALA link in the 4700 system, a work station can assume the role of either a Secondary Logical Unit (SLU), a Primary Logical Unit (PLU), or both. As an SLU, the station uses the current 4700 communication support to continue a single session started by a PLU. As a PLU, the station establishes one or more sessions, each session with a unique SLU.

The ALA link provides a Physical Unit Type 5 (PU T5) node, containing a boundary function, which resides in the 4700 controller. Functions performed by the System Services Control Point (SSCP), a part of the PU-T5 support, are shared between the 4700 application and the ALA link. The SSCP is responsible for establishing a session between itself and each supported Physical Unit (PU) (referred to as the SSCP-PU session). Additionally, the SSCP establishes sessions with each SLU in the PU-T1 and PU-T2 nodes (referred to as SSCP-LU sessions). After these sessions are established, various Network Services (NS) commands can be used to perform optional SNA functions, such as "Initiate-Self" on the SSCP-LU flow and "Request Maintenance Statistics" on the SSCP-PU flow. Also, depending on the type of SLU, data can be sent and received on the SSCP-LU flow.

A third type of session is established between the SLU and the PLU (referred to as the LU-LU session). The LU-LU session transfers the actual "working" data.

Device and Network Addressing

As with other devices, you must define the ALA-attached device to the 4700 system. The 4701 selects the ALA device using the address you assign during that configuration (CPGEN) definition.

Network ID (NID)

You must specify an ALA device network identifier, or NID, during the CPGEN process. The NID is associated with the physical device or line address, and must be set into a specific field (AMSNID) before an application program can send data to, or receive data from, the ALA device.

The two-byte NID may be any value except all zeros or all ones, and must differ from any NIDs for other devices. Associated with the NID are two one-byte addresses: the device poll address and the device select address. The application program may interrogate these addresses, which are also defined at configuration (CPGEN) time.

Poll and Select Addressing

Each physical control unit on the ALA network has an address. This one-byte address is used for polling--requesting input from a control unit. The poll address for a control unit is specified at CPGEN time with the ALACU macro.

Each ALA link SLU also has a select address. The PLU uses the one-byte select address to write to the SLU. The PLU transmits the select address as the destination address field (DAF) of the transmission header (TH). If the attached SLU is another 4700 controller work station, the select address is the same as the work station number that represents the SLU. The select address for an SLU is specified during CPGEN by the ALATERM macro.

An application program can query the ALA NID to find the poll and select addresses for a device. The program can also change the select address. Both operations are done with the LCNTRL command, described in Chapter 3.

Polling

The ALA link polls each ALA-attached device serially. Each device is polled once for each pass through the polling list. The polling operation begins automatically following an LCNTRL Start Line instruction if there is an available read buffer, and at least one device is in the Vary Online Pending state.

A normal poll sequence begins when the controller issues either a Receive Ready (RR) command with the poll bit set or an I frame with the poll flag set. The controller then issues a Set Normal Response Mode (SNRM) to establish initial contact with the SLU. If the controller detects an out-of-buffers condition, it continues polling with the Receive Not Ready (RNR) command. This causes the secondary station to remain active but prevents it from sending more data.

The ALA link follows a write-over-read priority on a station-by-station basis. This means that if the controller finds a station with a pending write request during polling operation, the controller sends a message having the poll bit set to that station. If no stations have write requests pending, the controller continues normal polling as described before.

Two conditions are necessary to permit polling to an ALA control unit to begin:

- The ALA link must be started.
- The ALA device or terminal on the device must be in the "online pending" state.

These two conditions can occur in either order. After polling begins, the link begins a select sequence when your application program issues a write command.

The ALA link operates in either normal or slow poll mode. The controller removes a device from the normal poll list and places it in slow poll mode if "loss of contact" status occurs during normal polling. Slow polling means a device receives an SNRM poll once for every "n" passes through the normal poll list (you define "n" on the ALALINE macro during the CPGEN process). The controller returns the station to the normal poll list when it receives a nonsequenced acknowledgment (NSA) response, indicating that contact was reestablished.

An ALA device also enters the slow poll list when it is first polled. It remains on the slow poll list until the controller establishes contact with the device and receives a positive acknowledgment. If the device on the normal poll list does not acknowledge a poll or select sequence, the controller decreases the retry count established during the CPGEN process. When the retry count reaches zero, the controller posts the station that owns the device with "loss of contact," and moves the device to the slow poll list.

Polling to a control unit ends when the control unit or the last terminal on the control unit enters the offline state or if the ALA line enters the nonoperational state. If the line stops with devices still online, all devices that were being polled at the time of failure are re-polled when the line restarts.

Device and Line States

Your application program uses the device and line states to determine what action to take. The program issues the LCNTRL SENS instruction, and receives ten bytes of status information. The first status byte represents the following device and line states:

Device States

Offline: The ALA device enters offline state if CPGEN or a Vary Offline instruction placed the device in the inactive state. To place the device in online or online pending state, your program must issue a Vary Online instruction.

Online Pending: The ALA device enters online pending state if it was designated active by the CPGEN process, by a Vary Online instruction, or if the device posted “loss of contact” while online. The device goes to the Offline state after a successful completion of a Vary Offline instruction or after an Online status X‘0800’ is reported.

Online Status Pending: This state means the controller made contact with a secondary controller or attached device. Online Pending state is always present with the Online Status Pending state. The polled controller or device leaves Online Status Pending state when it presents Online status to your application program. The device enters the Online state at this time.

Online: A secondary controller or attached device enters this state from either the Online Pending state, by sending Online status (X‘0800’) to the 4700 controller, or directly from Offline state when a Vary Online instruction completes. The controller or device returns to either Offline state after a Vary Offline instruction completes, or to Online Pending state if Loss of Contact status occurs.

Loss of Contact Status Pending: The device enters this state when contact is lost with, or before contact is presented to, your application program. The Online Pending state is always present with this state. The device leaves this state when the link presents Loss of Contact status to your application program.

Owned: You specify a device as "owned" either during the CPGEN process or by assigning it from the free device pool using the ASSIGN instruction. Any combination of devices can be owned. A device can be owned even though it might not be varied online (for example, a control unit in Message Routing mode can be owned).

Unowned: A device is "unowned" if either it was not assigned during CPGEN or you released it with the ASSIGN instruction to the free device pool.

Line States

Nonoperational: An ALA line leaves this state when either the system monitor, operating during startup, or the application program performs a successful Start Line instruction. The station reenters nonoperational state when it either detects loss-of-contact, or executes a Stop Line instruction. The line adapter does not set nonoperational state; the adapter can be either enabled or disabled.

Operational: The ALA line enters this state after starting successfully. It leaves this state when either a line loss of contact occurs, or when the controller program issues a Stop Line instruction.

ALA Link Operating Mode

A 4700 system with the ALA link RPQ installed operates in message-routing mode. Message-routing is the ability to send a message from a station to a terminal owned logically by (but not physically unique to) that station, or from the terminal to the owning station.

You must define the terminals with ALATERM macros during CPGEN. These macros define the terminal addresses and associate them with a logical work station. When writing to or reading from one of the terminals, the link creates the appropriate headers, inserts the physical address required, and routes the message through the network. When a message arrives at the primary station from a terminal, the link removes the headers before placing the message in your program segment. Appropriate information from the headers is placed in appropriate fields in the ALA link's alternate machine segment (AMS), described later in this chapter.

Read Buffer Allocation

Although messages are written directly from a user's segment, data is read into buffers defined for each ALA line. You must define the number and size of the buffers during CPGEN with the ALABUFF macro. The ALALINE macro defines the use of either single or dynamic buffer allocation.

Single Buffer Allocation

When you specify single-buffer allocation at CPGEN time, only one buffer will be used to hold an input message. Therefore, you should specify the buffer size as a value equal to or greater than the length of the largest input message from the ALA link devices. If the message does not fit into a single buffer, overflow status is reported to your program.

Dynamic Buffer Allocation

Dynamic read buffer allocation, set by specifying DBA=Y in the ALALINE macro, causes the controller to use as many available buffers as needed to contain an incoming message. If not enough buffers are available, a "buffer overflow" condition occurs, and the 4700 controller does not acknowledge the message.

If this condition occurs "n" (specified in CPGEN) times in a row, data is lost; the controller signals a "data count exceeded" condition to your program. You should not specify dynamic buffering if you use inbound message segmenting. (See "Message Segmenting" on page 3-31 for more information.) Assuming no errors occur, the link control presents the complete message after receiving "end of message". You should ensure that the buffer size, specified by the CNL= operand of the ALABUFF macro, is large enough to hold the average input message received during normal operations, and that the number of buffers specified can contain at least the maximum size message.

When you use dynamic buffering, it is possible that an incorrectly received message may be written in the station's storage. This occurs because your program execution resumes at the asynchronous ALA link program entry point when the first buffer of the message becomes available. The station's program issues LREAD, causing data to enter storage. When your program receives the part of the message in error, it issues the X'4000' status.

Note that if the address portion of the message is in error, the link control may route the message to the wrong work station. Therefore, your program should issue LEXIT if it detects the X'4000' status after issuing LREAD, because the message may not be for your station.

The ALA Link Machine Segment (AMS)

You must define a new communication area in your segment having fields similar to those used to communicate with the host. This area, called the alternate machine segment (AMS), is like the normal System Machine Segment (SMS) in Segment 1. Figure 3-1 shows the AMS format. The following is a description of the contents and purpose of the major fields in the AMS:

Network Identification Field (AMSNID): This two-byte field contains the network ID (NID) of the SLU or PU that was just read, or to which your program issues LWRITE or LCNTRL instructions. If your program writes back to the same NID from which a read was just received, your program does not have to reinitialize AMSNID. The CPGEN defines the NID values you enter in AMSNID.

Read Control Field (AMSARC): This two-byte field contains information about a message just read from an ALA device. AMSARC comprises two subfields:

1. The Read Type Subfield (AMSARCT): This one-byte field describes the type of message received. The meanings of the values set into this subfield are defined under "ALA/AMS Control Fields and Indicators" on page 3-34, later in this chapter.

2. The Read Flags Subfield (AMSARCF): This one-byte field describes the flags that accompanied the message received. The meanings of the codes set in this subfield are described under "ALA/AMS Control Fields and Indicators" on page 3-34, later in this chapter.

The Write Control Field (AMSAWC): Your application program must set this field to describe the message being sent to the ALA SLU. The AMSAWC comprises the following two subfields:

1. The Write Type Subfield (AMSAWCT): This one-byte field describes the type of message being sent. The meanings of the codes set in this subfield are described under "ALA/AMS Control Fields and Indicators" on page 3-34, later in this chapter.
2. The Write Flags Subfield (AMSAWCF): This one-byte field specifies the flags that will accompany the message being sent. The meanings of the codes set in this subfield are described under "ALA/AMS Control Fields and Indicators" on page 3-34, later in this chapter.

The Write Event Number (AMSAEN): This one-byte field assigns an ID to a write issued by an application program.

The Write Event Failure Number Field (AMSAFN): This one-byte field identifies a failing write operation.

The Read Sequence Number Field (AMSASQ): This two-byte field contains the sequence number or message ID of any message received or written. The controller assigns the sequence number to all LWRITE operations. After the LWRITE completes successfully, the ALA link should return the correct sequence number. If you do not provide enough AMS space for sequence numbers, none are returned and the ALA link sets a status of X'040F' in SMSDST. PU-T1 devices do not use AMSASQ.

If you specify a nonzero *ssid* parameter on the STATION CPGEN macro, AMS contains the following extension fields:

The Read Flags Extension (AMSARCE): This field contains additional flags that accompany a received message. Refer to "ALA/AMS Control Fields and Indicators" on page 3-34 for a description of the flags.

The Write Flags Extension (AMSAWCE): This field contains additional flags that accompany a message written to the ALA device. See "ALA/AMS Control Fields and Indicators" on page 3-34 for a description of the flags.

The Next Read Type (AMSANRT): This field describes the type of message in the buffer for this work station. The LREAD AL instruction description describes the search continue indicator, which determines the contents of this field.

<——— Byte n ———>	<——— Byte n+1 ———>
NETWORK IDENTIFICATION FIELD (AMSNID)	
READ CONTROL FIELD (AMSARC)	
Read Flags Field (AMSARCF)	Read Type Field (AMSARCT)
WRITE CONTROL FIELD (AMSAWC)	
Write Flags Field (AMSAWCF)	Write Type Field (AMSAWCT)
WRITE EVENT NUMBER (AMSAEN)	WRITE EVENT FAILURE NUMBER (AMSAFN)
SEQUENCE NUMBER FIELD (AMSASQ)	
READ FLAGS EXTENSION (AMSARCE)	WRITE FLAGS EXTENSION (AMSAWCE)
NEXT READ TYPE (AMSANRT)	(reserved)
(reserved)	

Figure 3-1. The Alternate Link Machine Segment

System Machine Segment (SMS) for the ALA Link

The SMSIML, SMSDST, and SMSAFL fields in the basic SMS are used by the ALA link as well as by other 4700 devices. Defined here is the ALA link's use of these SMS fields. For a total description of the SMS and its use, see Volume 1 of the *4700 Controller Programming Library.*

Message Length Field (SMSIML): This two-byte field contains the length of data placed in a station's segment as a result of an LREAD AL instruction, the amount of data *not* transmitted (residual length) by an LWRITE AL instruction, or the displacement into a parameter list identifying the entry last processed after an LCNTRL instruction operation.

Device Status Field (SMSDST): This two-byte field contains a code indicating the status that resulted from an I/O operation. The codes are described in detail in Appendix D, "Link Status Codes"of this manual.

Asynchronous Input Indicator (SMSAAP): This indicator occupies the third high-order bit position (X'20') in the SMSAFL field in Segment 1. It is set on whenever input or status is pending for a work station and reset when the station issues an LREAD AL (non-read search) for the latest input or status pending.

Two new SMS fields are used to provide your application program with the location of the alternate machine segment (AMS) specified for the station during the configuration process.

AMS Segment Field (SMSAMS1): This one-byte field contains the segment number in hexadecimal of the segment containing the AMS.

AMS Displacement Field (SMSAMS2): This two-byte field contains the hexadecimal displacement, from the start of the segment addressed by SMSAMS1, of the AMS.

Message Structure

Messages (requests and responses) have almost the same structure as host link messages. However, the ALA link allows two TH formats: format identifiers 2 (FID2) and FID3. Request/response headers are supplied by both your application program and the ALA link. The program supplies this information in the write control field for write operations; the link supplies the RH in the read control field on read operations. (See "ALA/AMS Control Fields and Indicators" on page 3-34.)

Message Flow

The ALA link permits the same two routing modes as the SNA/SDLC host link: Normal and expedited flow. Normal-flow messages occur sequentially. Expedited-flow messages also occur sequentially, but are sent by the ALA link ahead of any pending normal-flow messages. The messages on the ALA link are discussed later in the chapter.

Message and Command Types

Listed below are messages that can be sent or received by the PLU work station over the ALA link. See "ALA/AMS Control Fields and Indicators" on page 3-34 for the direction that each command flows. Many of the commands received by your program are responded to by the ALA link; this information is also under that same heading.

LU-LU Session Messages

The session control commands allowed during an LU-LU session depend on the TS profile selected in the Bind parameters. The Bind and Unbind session commands are used in all TS profiles. Bind commands start sessions; Unbind commands end sessions. Figure 3-2 shows the valid session control commands for any profile supported by the ALA link. If the PLU attempts to write an invalid session control command, status is returned; if an invalid session control command is received, the session is ended. (See the description of terminal/control unit counter 11 in Chapter 6.)

TS Profile	SDT	CLEAR	STSN	RQR
2		X		
3	X	X		
4	X	X	X	
7				

X — Command allowed for specified profile

Figure 3-2. LU-LU Session Commands Allowed by TS Profiles

Bind: The PLU application program sends Bind to initiate a session. If the SLU wants to reject the session, it sends a negative response; if the SLU wants to accept the session, it sends a positive response.

The data accompanying the Bind in the PLU's segment consists of 26 bytes of parameters, a one-byte binary count indicating the length of the resource name (PLU application program name), the resource name (from 1 to 8 characters), and user data. The length of the bind parameters cannot exceed 256 bytes.

Clear: Clear purges all messages between the PLU and SLU and the response purges all messages between the SLU and the PLU, if in fact any messages are encountered. All messages and responses in the network are lost, and all sequence numbers in the network are set to zero. The SLU is placed in data-flow-reset state. Resynchronization may be required, depending on your application.

Request Recovery: The SLU sends Request Recovery when it detects a critical error. The PLU should then suspend data transfer until resynchronization has taken place.

Set and Test Sequence Numbers: The PLU sends Set and Test Sequence Numbers (STSN) to resynchronize data flow (for example, during session restart). Before sending this command, the PLU must stop the flow of data in the network either with a Clear command or by ending the old session and starting a new one. Each network element examines the action codes in the first byte of the data; the sequence numbers throughout the network are reset as indicated by the action codes.

Five bytes of data accompany this command. The first byte contains action codes for the SLU and PLU sequence numbers. The next 2 bytes may contain a new SLU sequence number, and the last 2 bytes may contain a new PLU sequence number. Bits 0 and 1 of the action code refer to the SLU sequence number, bits 2 and 3 refer to the PLU sequence number, and bits 4 through 7 are reserved. The action codes (setting of bits 0 and 1, and bits 2 and 3) are:

00 - Ignore: The sequence number is not affected by this command.

01 - Set: The applicable number has been altered in the PLU and in the controller.

10 - Sense: The PLU is requesting that the SLU supply the current applicable sequence number in the response.

11 - Set and test: The applicable sequence number has been altered in the PLU and in the controller. The program must compare the SLU's sequence number with the one supplied in the STSN; the result of the comparison should then be indicated in the response.

The SLU responds to this command with a positive response and either 0 or 5 bytes of data. A negative response causes a Data Check SNA error. A positive response without data indicates that the sequence number or numbers are acceptable. A positive response with data indicates that the logical work station is sending its version of the sequence number. The 5 bytes of data sent with the response contain a result code (the first byte), and can contain an SLU sequence number (the next 2 bytes) and a PLU sequence number (the last 2 bytes). Bits 0 and 1 of the result code refer to the SLU sequence number, and bits 2 and 3 refer to the PLU sequence number; bits 4 through 7 are reserved. The result codes (setting of bits 0 and 1 and bits 2 and 3) are:

00 - Reset: This result code is reserved.

01 - Positive: The sequence number received in the STSN command is equal to the sequence number in use by the SLU.

10 - No number: The PLU requested the sequence numbers, but the SLU does not have a valid number.

11 - Negative: The applicable sequence number field contains the number requested (response to a 10-action code), or the number should be changed to the one supplied by the SLU (response to action code 11).

Start Data Traffic: The PLU application program sends Start Data Traffic (SDT) to indicate that the session is established and communication can begin. This command causes the controller to remove the station from data-flow-reset state. Depending on the TS Profile, this command might not be required.

Unbind: The PLU application program sends Unbind to end a session. This command causes the controller to remove the SLU from the in-session state.

Expedited-Flow Data Flow Control (DFC) Commands

The expedited-flow DFC commands during an LU-LU session depend on the FM profile selected in the Bind parameters. Figure 3-3 shows the expedited-flow DFC commands for any profile supported by the ALA link. If the PLU attempts to write an invalid expedited-flow DFC command, status is returned. Enforcement of the FM profile for commands read is the responsibility of the PLU.

	FM Profile				
	2	3	4	7	18
QEC			X		
RELQ			X		
RSHUTD		X	X	X	X
SBI*					X
SHUTC		X	X		
SHUTD		X	X		
SIG		X	X	X	X

X — Command allowed for specified profile
* — Allowed only if brackets are used

Figure 3-3. Expedited-Flow DFC Commands Allowed by FM Profiles

Quiesce at End-of-Chain: An LU sends Quiesce at End-of-Chain (QEC) to request the receiving LU to stop sending data at the end of this data chain, and reply with the QC message.

Release Quiesce: An LU sends Release Quiesce (RELQ) to indicate that data flow, which stopped when the QC or Shutdown Complete message was received, should resume. RELQ resets the quiesce state in the SLU.

Request Shutdown: An SLU sends Request Shutdown (RSHUTD) to request an orderly ending of the session. If the PLU application program wishes to end the session, it responds with a Shutdown or Unbind command, depending on application protocol.

Stop Bracket Initiation: An LU sends Stop Bracket Initiation (SBI) to request that the receiving LU stop sending Begin Bracket (BB) and Bid. The receiving LU may continue to send BB and Bid until the "Bracket Initiation Stopped" message is sent in reply (see "Bracket Protocol," later in this chapter).

Shutdown: The PLU application program sends Shutdown (SHUTD) as part of the orderly ending of a session. It indicates that the SLU should stop sending data and prepare for session ending. When the SLU is ready to end, it sends a "Shutdown Complete" (SHUTC) command.

Shutdown Complete: The SLU sends Shutdown Complete (SHUTC) to indicate that it is ready for session ending. The next message sent can be Clear or Unbind depending on the protocol used.

Signal: Signal (SIG) is an expedited command the PLU can send to the SLU, regardless of the status of the flow. It carries a four-byte signal code; the first two bytes are the signal field and the last two bytes are for program use.

Normal-Flow Data Flow Control (DFC) Commands and Data

The normal-flow DFC commands allowed during an LU-LU session depend on the FM profile selected in the Bind parameters. Data can be included on LU-LU sessions, regardless of the FM profile you select. Figure 3-4 shows the valid normal-flow DFC commands for any profile supported by the ALA link. If the PLU attempts to write an invalid normal-flow DFC command, status is returned. Enforcement of the FM profile for commands read are the responsibility of the PLU.

	FM Profile				
	2	3	4	7	18
BID*		X	X		X
BIS*					X
CANCEL		X	X	X	X
CHASE		X			X
LUSTAT		X**	X	X	X
QC			X		
RTR*		X	X		X

X — Command allowed for specified profile
* — Allowed only if brackets are used
** — SLU-to-PLU only

Figure 3-4. Normal-Flow DFC Commands Allowed by FM Profiles

Bid: The half-session wishing to start data transmission, called the *bidder*, issues Bid to request permission to send data. Bid is used with bracket protocol (see "Bracket Protocol," later in this chapter). If the receiver of Bid (first speaker) does not want to receive the data (for example, while it is in a synchronous dialog), it sends a negative response. When the first speaker wants to receive the data, it sends a "Ready to Receive" command.

Bracket Initiation Stopped: Bracket Initiation Stopped (BIS) is sent by an LU in reply to a Stop Bracket Initiation command. After sending this command, the sender cannot issue Begin Bracket (BB) or BID (see "Bracket Protocol," later in this chapter).

Cancel: An LU sends Cancel to indicate that an error has occurred in the current group of chained messages (see "Data Chaining" on page 3-33). The LU that receives Cancel should discard the messages already received. The next message sent should be an unchained or first-in-chain message.

Chase: Your application program sends Chase to ensure that all messages have been received and responded to by the receiving LU. When the station reads the response to this message, the messages preceding Chase have been received and all response requirements have been satisfied.

Data (Normal-Flow): An LU sends data after the LU-LU session is established. When transmitting data, the PLU writes the data directly from the segment indicated by the LWRITE instruction. Data can be accompanied by a response request, begin bracket, end bracket, or change direction indicator, a chaining flag, or a code selection indicator. Your program sets the flags in AMSAWCF before issuing LWRITE AL. The operation is complete when control returns to the PLU after an LCHECK AL executes, or after exceeding the count of LWRITEs allowed for the PLU or SSCP/application program, plus one.

The PLU places received SLU data in the segment indicated by the LREAD instruction. Data can be accompanied by a response request, a begin bracket, an end bracket, a change direction indicator, a chaining flag, or a code selection indicator. AMSARCF indicates which flags are set.

Data and Control (Normal-Flow): Data and Control is the same as Data, except that the beginning of the message includes control information in the form of an FM header.

Logical Unit Status: An LU can send Logical Unit Status, combined with four bytes of user-designated information. Logical unit status is an alternative message the LU can use if it cannot send a negative response.

Quiesce Complete: Quiesce Complete (QC) indicates that data transfer is suspended. QC is normally a reply to a Quiesce at End-of-Chain (QEC) message. To resume data transfer, the LU must receive a Release Quiesce message.

Ready to Receive: The first speaker sends Ready to Receive (RTR) to tell the bidder that it can now receive data. The SLU sends RTR after sending a negative response to a PLU's Bid command. When the PLU/application program (bidder) receives RTR, it should send a positive response followed by data, or a negative response with sense information that indicates that data is no longer available.

SSCP-PU/LU Session Messages

Data (Normal-Flow): The SSCP/application program sends data after the SSCP-LU session is established. When transmitting data from the SSCP/application program, the SSCP writes data directly from the segment specified in the LWRITE instruction. The write control field flags (AMSAWCF) are unused and a DR1 response is always requested. The operation completes the same as the Normal Flow Data operation, described earlier.

The SSCP places data received from the SLU on the SSCP-LU session in the segment specified in the LREAD instruction. Data is accompanied by a DR1 response request only. No other bits in AMSARCF are significant. The SSCP-PU session cannot send or receive data.

Network Services Commands (Normal-Flow): NS commands can be sent or received (solicited or unsolicited) on the SSCP-LU or SSCP-PU sessions, depending on the type of device being supported. For some devices, the data received in the ACTLU or ACTPU responses indicate the type of NS commands allowed. For those devices that do not use the extended response format, no unsolicited NS commands are allowed. In either case, the type of NS commands allowed depends on the device being supported.

Note: Because PU-T1 nodes do not use ACTPU and the ALA link cannot use XID, you cannot determine the NS commands allowed for these devices during the session. Therefore, you must determine whether or not NS commands will be used before writing the application program.

Logical Unit Status (Normal-Flow): This command is used on the SSCP-LU session when a device utilizes the "LUs readiness to accept Binds and other requests" field in the ACTLU response. If this value is set to 1, an LUSTAT must be received by the SSCP/application program with the code X'0001' before an LU-LU session can be allowed. An LUSTAT with code X'0831' can be encountered after receipt of an LUSTAT with code X'0001' or a ACTLU response with the readiness flag set to zero. This means that an LU-LU session cannot be allowed until a LUSTAT (X'0001') is received. LUSTAT X'0831' has no affect on the current LU-LU session.

Responses

A response to a message is itself a message containing information about the transmission and processing of the original message. A response can be positive or negative. A positive response indicates that the message has been received successfully and its content is acceptable. A negative (or 'exception') response indicates that the message was not received or that its content was unacceptable. Each negative response carries a four-byte sense code that indicates the exception condition that was encountered.

The program sends and receives responses in a manner similar to that used for data messages:

- When your program sends a response, it sets the AMSAWCT (the write type field) to the type of response required and then issues an LWRITE instruction.
- When your program receives a response from an SLU, it issues an LREAD instruction; when the read is complete, your program tests AMSARCT (the read type field) to determine the type of response read.

The message sender indicates a response protocol for the receiver to follow. The ALA link defines three types of response protocols:

1. No response protocol indicates that the sender of the message does not want (and will not accept) a response to the message.
2. Exception response protocol indicates that the sender of the message wants, and will accept, a response only if the message is unacceptable.
3. Definite response protocol indicates that the sender of the message always wants a response, regardless of whether the message was acceptable or not.

The message that requires the response contains a response protocol indicator defining the required response. The request type is indicated in the control field flags in the AMS.

- When requesting a response from an SLU, your program sets AMSAWCF (the write flags field) to indicate the type of response desired, sets AMSAWCT to indicate the message type, and then issues an LWRITE instruction.
- The program detects a response by first issuing an LREAD instruction and then testing AMSARCF (the read flags field) for a response code.

Every response, whether positive or negative, is further designated by its sender as a definite response 1 (DR1), a definite response 2 (DR2), or a combination of both (DR1 and DR2). Some LUs might not need to distinguish between the different kinds of positive and negative responses. For those that do, the distinction is made for purposes recognized by the LU.

Responses to Messages Sent by the Program/PLU

While all messages sent by your program (except responses) include response protocol indicators, your program can set these indicators only for data and data-and-control messages. The controller sets the response protocol indicators for all other messages (commands) so that definite response protocol is used and a DR1 response is requested.

For example, when the PLU sends a Bind Command (see "Beginning a Session" on page 3-21 later in this chapter), your program receives either a positive or negative DR1 response to the command (hexadecimal "09" or "0D"). If a positive response is received, the SLU is acknowledging receipt of the Bind and indicating that the session is permitted. If a negative response is received, the SLU is acknowledging receipt of the Bind, but indicating that the request is rejected.

When your program sends data, the response protocol and type of response requested are indicated by setting bits in the AMSAWCF write flags subfield. A response of the requested type will then be returned by the SLU. Figure 3-5 shows the seven possible response protocols and the AMSAWCF fields that are set for each. The figure also shows the responses possible for each protocol and the AMSARCT bits that are set for each response.

Response Protocol Requested:				Possible Response Types:	
To request this protocol:	Set AMSAWCF to: Bits 5	6		To determine receipt of:	Test AMSARCT field for:
No Response	0 or 1	0			
Exception					
DR1	1	0	1	Negative DR1	X'05'
DR2	1	1	0	Negative DR2	X'06'
DR1 & DR2	1	1	1	Neg. DR1 & DR2	X'07'
Definite					
DR1	0	0	1	Pos or Neg DR1	X'01' or X'05'*
DR2	0	1	0	Pos or Neg DR2	X'02' or X'06'
DR1 & DR2	0	1	1	+ or − DR1 & DR2	X'03' or X'07'

*The AMSARCT is set to X'09' or X'0D' for a response to any write request in which the link has automatically requested a DR1 type response. See VVWrite Type Field (AMSAWCT) Codes on page 3–37.

Figure 3-5. PLU Response Protocol Requests and SLU Responses

Any data received on a response is presented to your program. This includes a one-byte SNA command code for Session Control and Data Flow Control commands and three bytes for Network Services Commands. Depending on the type of command, additional data may follow the command bytes. For negative responses, the command codes follow four bytes of sense information.

Responses to Messages Received by the Program/PLU

All messages sent by the SLU include response protocol indicators. The response to most commands is automatically sent by the controller; but responses to others, including data and data-and-control, must be sent by your program. The description under "Message Flow" on page 3-11 describes to which messages the controller responds automatically.

Response protocol indicators are included in the AMSARCF field, which is set by the controller when the message is received. The bit settings and position within the field correspond to the AMSAWCF field, which is set by your program when a message is sent. In addition to setting AMSARCF, the controller maintains a copy of the TH, RH, and three bytes of the RU that it uses to format the response written by your program.

The program can send two types of responses: positive and negative. The controller interprets the response indicated by your program and sends a response that is appropriate for the type requested. For example, if a DR1 definite response protocol was requested (bits 5 and 6 set to 0, bit 7 set to 1), the PLU can indicate:

- Positive response, and the controller will send a positive DR1 response.
- Negative response, and the controller will send a negative DR1 response.

Figure 3-6 shows the responses that the SLU program can request, and the responses that the PLU controller gives.

Requested Response Resp sent by PLU	Exception Response Protocol			Definite Response Protocol			No Response Protocol
	DR1	DR2	1&2	DR1	DR2	1&2	NONE
Positive	No Msg Sent	No Msg Sent	No Msg Sent	Pos DR1	Pos DR2	Pos DR1 & 2	No Msg Sent
Negative	Neg DR1	Neg DR2	Neg DR1 & 2	Neg DR1	Neg DR2	Neg DR1 & 2	No Msg Sent

Figure 3-6. Responses Sent by the Controller

Unrequested Responses

The application program might attempt to send a response when none is required. In this case no response is sent and the LWRITE completes successfully; that is, a no-operation occurs.

Sending Data with Responses

Data is not sent with any positive responses, but data must accompany all negative responses. The data sent with a negative response will be 4 bytes long; the first two bytes contain sense codes defined by ALA, and the last two bytes can be defined by you. If the write specifies more data to be sent than necessary, the superfluous bytes are ignored and the LWRITE completes successfully. Chapter 2, "SNA/SDLC Host Link Programming," describes the sense bit settings defined by the ALA link.

Control Modes

The ALA link enforces immediate response mode for all messages received by your program. These messages can be on the normal or expedited flow in either the SSCP-PU, SSCP-LU, or LU-LU sessions. The enforcement is accomplished by not allowing your program to read from the same SLU or PU without sending a response if one is owed. If a read is issued by your program and a response is outstanding for the SLU or PU presenting data, the ALA link generates a positive response. Note that if no data is available and X'4000' status is returned, no response is generated by the ALA link.

If your program issues a general read, one which can be satisfied by a message from any SLU or PU owned by the station, the ALA link generates a positive response only if a response is required for the SLU or PU whose message is being read. Note that once it reads a message, your program cannot respond to a previous message for the same SLU or PU.

For messages sent by your program, the control mode enforced depends on the session, and whether the flow in that session is normal or expedited. Immediate control mode is enforced on the LU-LU session for the expedited-flow and for all SSCP-PU and SSCP-LU session flows. This implies that only one session control command or expedited data flow command can be written before the response to the command is read; that is, only one outstanding request of this type is allowed per SLU. On the LU-LU session, Unbind and Clear are exempt from this enforcement and can be issued by the PLU at any time during the session. If your program attempts to write a second command without reading the response to the first, the LWRITE is discontinued with X'0441' status.

The request control mode used on an LU-LU session is specified in the Bind parameters. It is the responsibility of the PLU to enforce this mode.

Beginning a Session

Several events must occur before data move between the PLU and SLU on the LU-LU session:

1. The SDLC line must be active. You may specify the line as active either during the CPGEN process, or by coding on LCNTRL/Start Line statement in your program.

2. An SLU must be owned. You can assign an SLU to a station with the LCNTRL/Assign instruction or during the CPGEN process.

3. An SLU must be varied online. This also can be done with either an LCNTRL/Vary Online instruction after ownership has been established or during the CPGEN process.

   Note: Before varying and SLU on- or offline with LCNTRL VONL or VOFF, you must load the network ID (NID) defined by the NETID= operand of the ALATERM macro into AMSNID. Do not use the NID defined by PUNID= on the ALACU macro, or the SSCP-PU session will fail.

4. After an SLU is owned and varied online and the SDLC line is started, an initial contact poll must be transmitted; an NSA response indicates that the SDLC station is ready.

5. When the SDLC station becomes ready, the SSCP-PU session is established. For a PU-T2 (FID2) node, the SSCP does this by sending an Activate Physical Unit (ACTPU) command to the PU and reading the positive response. (The first five characters of the CPGEN identifier are used as the SSCP identifier in the ACTPU data.) A negative response (SNA protocol error) is placed in the system log along with the sense data received, and the PU remains in the online pending state. SLUs associated with the PU are not varied online and no ready status is received. For PU-T1 (FID3) nodes, the SSCP-PU session is established implicitly when the SDLC station reports ready with the NSA link response.

6. After the SSCP-PU session is established, the SSCP establishes an SSCP-LU session with each SLU varied online. The SSCP does this by sending an Activate Logical Unit (ACTLU) command to each SLU and reading the positive response. When the SSCP receives the positive response, a Ready indication is presented to the owning station. If the SSCP receives a negative response (usually LU not defined), it writes the response and any sense information received, in the system log. The SLU remains in the online pending state, but the owning station does not receive ready status.

If you select the option during CPGEN, your program passes the parameters received from the response to the ACTLU to the SSCP/application program on a subsequent LREAD X'16' in AMSARCT). This is done on the next LREAD directed to the NID that presented ready status.

7. When the work station receives the Ready indication, the LU-LU session begins. The PLU can start the session after receiving ready, or the SLU can start the session after the SSCP-LU session has been established.

Session Initiation by the SLU

You may decide that sessions will be initiated by the SLU. When the SSCP-LU session is established, the controller passes the Ready indication to the SSCP/application program. The SLU can now request session initiation by sending the Network Services (NS) Initiate Self command to the SSCP/application program. Alternatively, some SLUs may use FM data on the SSCP-LU session to request an LU-LU session. The application program name will be part of the message because the SLU must be owned before it can be made ready, the name has little value. However, the SSCP/application program may wish to reassign the SLU to another station, based on the application name.

The SSCP/application program receives the NS Initiate Self command and optionally checks whether the named application program is known and active. If your program is known and active, the SSCP/application program sends a positive response to the SLU and prepares to create a session. If your program is not known, the SSCP/application program sends a negative response and no session is created.

If the SSCP/application program sends a positive response to an Initiate Self command, but the session cannot be established for whatever the reason, a Network Services Procedure Error (NSPE) command is sent to the SLU to halt the attempt to establish a session.

Session Initiation by the PLU

You might decide that sessions will be initiated by the PLU application program. The PLU initiates a session by generating a Bind command (the subsequent positive response establishes the agreement to communicate) and a Start Data Traffic (SDT) command (which signals that data transfer can begin). Some SLUs might not require an SDT, depending on the TS profile specified in the Bind parameters. A data field associated with the Bind command contains the name of the PLU application program and the session bind parameters. The SLU can examine this name and bind parameters to determine if the session should be allowed.

The SLU examines the bind parameters to determine the protocols to use for this session. If the session is to be allowed, the SLU returns a positive response; if not, the SLU returns a negative response. The following example is part of an application program that begins a PLU-initiated session with the SLU. For this example, either the SLU was previously varied online by another portion of your program, or it was specified as online in CPGEN.

```
MVDI       AMSNID,X'0001'        NETWORK ID OF SLU
MVDI       AMSAWC,X'00A0'        BIND TO WRITE CONTROL
MVFXD      OUTSEG,BINDPARM       BIND PARAMETERS TO OUTPUT SEG
LWRITE     AL,OUTPUT             WRITE THE BIND
JUMP       ST,ERROR              HANDLE INITIAL STATUS
LCHECK     AL                    WAIT FOR WRITE COMPLETE
LEXIT                            EXIT, WAIT FOR RESPONSE
```

If execution began at your program's ALA= entry point, the following code could be used to read the BIND response and then write the SDT:

```
LREAD      AL,INPUT              READ INPUT FROM THE SLU
CCDI       AMSAWCT,X'03'         IS IT A RESPONSE?
JUMP       NE,OTHERT             NO, SEE WHAT IT IS
SETFPL     CHAR1                 POINT TO FIRST INPUT CHAR
CCDI       CHAR1,X'A0'           WAS THIS BIND RESPONSE?
JUMP       NE,OTHERR             NO,SEE WHAT IT WAS
MVDI       AMSAWC,X'0082'        SDT TO WRITE CONTROL
SETFPL     OUTPUT,0,0            NO DATA WITH SDT
LWRITE     AL,OUTPUT             WRITE SDT
```

.

.

.

Bind Parameters

When the PLU or SLU writes a bind command, it can send up to 256 bytes of bind parameters with the command. The ALA link requires that you supply at least bytes 1 through 9, described below. Most of the Bind parameters are controlled at the LU level; however, several are link-level parameters, and are discribed below. Refer to the *IBM SNA Format and Protocol Reference Manual: Architecture Logic*, SC30-3112, for a complete description of the parameters.

Byte 0 - Request Code: set to X'31'. To set, place X'A0' in the write control field.

Byte 1 - Request Format and Type: This is the first byte of data in the user segment.

Byte 2 - FM Profile: Must be set to one of the following decimal values: 2, 3, 4, 7, or 18. If you fail to set this field to a valid value, the ALA link rejects the LWRITE and sets appropriate status.

Byte 3 - TS Profile: Must be set to one of the following decimal values: 2, 3, 4, or 7. If you do not set this field to a valid value, the LWRITE is rejected.

Bytes 4-7 -	Reserved
Byte 8 -	SLU Send Pacing Count (bits 2-7): This value is not used by the ALA link, but the value specified has significance in the overall network performance as explained in the section entitled "Inbound Pacing" on page 3-31. This value should be the same as byte 13, but this is not checked by the ALA link.
Byte 9 -	SLU Receive Pacing Count (bits 2-7): The ALA link uses this value to determine the number of data messages to be sent on the LU-LU session flow before waiting for a pacing response. The first message of the sequence carries the pacing count. If you specify this value, it changes the value specified by CPGEN. Only a subsequent Bind restores the CPGEN value. Although it is not checked by the ALA link, this value should be the same as parameter byte 12. If you specify zero, but the pacing parameter in the ALATERM macro is not zero, the ALA link sets this value to the CPGEN value and the SLU begins pacing mode. If you specify X"FF", the ALA link replaces it and the CPGEN pacing value with zero.

Using the Sessions

Network Services (SSCP-LU/PU)

Network Services (NS) provides protocols through which the services managers of the SSCP and of the PUs and LUs in the network can interact with monitor and control LUs, links, and sessions. NS commands (formatted FM data requests) and their responses are used on the SSCP-PU and SSCP-LU session flows by the SSCP/application program. A description of the NS commands is in the *IBM SNA Format and Protocol Reference Manual: Architecture Logic* (SC30-3112).

The types of NS commands supported depend on the type of device attached and individual user requirements. The ALA link accepts these commands and passes them to the SSCP/application program with an indication in the AMS read control field that a Network Services (NS) command was received. To write an NS command, you must set the appropriate value in the AMS write control field and issue LWRITE AL. The data you supply depends on the type of command being written. In reading or writing an NS command, the first three bytes of data identify the command.

NS commands on the SSCP-LU flow are routed to and from the station owning the NID associated with the SLU. NS commands on the SSCP-PU flow are routed to and from a single station owning the NID associated with the PU. This station, called the *designated station*, is controlled by the SSCP/application program. The designated station can write NS commands on this flow using the PU NID specified by the PUNID= operand of the ALACU macro. If there is no designated station, you cannot write NS commands during the SSCP-PU flow. If the controller receives an NS command on the SSCP-PU flow and there is no designated station, the controller writes the command to the system log and disconnects the connection.

You must specify the NID for the PU in the ALACU macro. You can optionally specify the designated station. In lieu of specifying a designated station in CPGEN, you can use the LCNTRL ASSIGN instruction to assign the NID to the designated station.

When the SSCP establishes an SSCP-PU session by issuing a vary online to an SLU, a ready status is indicated to the designated station. If no designated station has been specified, no NS commands can be read or written on the SSCP-PU flow and ready status is not reported.

Optionally, if specified during CPGEN, the parameters received from the response to the ACTPU (PU-T2 only) will be presented to the SSCP/application program on a subsequent LREAD X'16' in AMSARCT). This will be the next LREAD if the LREAD is directed to the PU NID.

Transferring Data (SSCP-LU)

Some devices allow unformatted FM data to be transferred on the SSCP-LU session (for example, an operator entered logon message). This session can be used when the Ready indication is read by the SSCP/application program, that is, when the SSCP-LU session is established. Writing on this flow is identical to writing on the LU-LU flow with the exception that an X'12' is set in the write type field (AMSAWCT) and the write flags field (AMSAWCF) is unused.

Transferring Data (LU-LU)

After the PLU or SLU establishes the LU-LU session and the PLU receives the response to SDT (SDT is not required, and is not allowed for TS Profiles 2 and 7), the PLU may begin transferring data on that session. Data is sent in the following manner:

- Prepare the data field in one of the station's segments.

- Set the write type field (AMSAWCT) to X'10' or X'11' for LU-LU session data. Code X'11' indicates that FM header is included at the beginning of the data. The FM header is the first n bytes (user defined) of the PLU segment.

- Set the write flags field (AMSAWCF) as desired. (Refer to "ALA/AMS Control Fields and Indicators" on page 3-34.)

- Set AMSNID to the NID of the desired SLU.

- Issue an LWRITE AL that refers to the data field.

- Check to ensure that the write was successful and that the data area can be reused.

The following shows the processing performed by a PLU application program that is sending data and requesting a definite response on the LU-LU session. These instructions prepare data in your segment:

```
      MVFXD     OUTSEG,DATA          MOVE DATA TO USER SEGMENT
      MVDI      AMSAWCF,X'0110'      SET UP AMSAWC FOR LU-LU
      * COMMENT LINE                 DATA, POS DR1 RESPONSE
      LWRITE    AL,OUTPUT            WRITE TO SLU FROM USER SEG
      JUMP      ST,ERROR             CHECK INITIAL STATUS
      LCHECK    AL                   WAIT FOR WRITE TO FINISH
      JUMP      ST,ERROR             CHECK ENDING STATUS
      LEXIT                          EXIT AND AWAIT RESPONSE
                .
                .
                .
SNAEP EQUATE *                       ASYNCH ALA ENTRY POINT
      MVDI      AMSNID,X'0000'       SET UP GENERAL READ
      LREAD     AL,INPUT             READ DATA
      CCDI      AMSARC,X'0103'       IS IT A POSITIVE DR1 RESP?
      JUMP      NE,OTHERARC          NO, SEE WHAT IT IS
      CCDI      AMSNID,X'0001'       IS IT FROM SLU WRITTEN TO?
      JUMP      NE,OTHERNID          NO, SEE WHO SENT IT
                .
                .
                .
```

The SLU may also begin sending data. Data is received by issuing an LREAD AL instruction at the ALA link entry point. The read should point to a segment (not 14) where the data will be placed. AMSARCT will indicate the type of data or command read.

Ending Data Transfer

The LU-LU Data Flow Control (DFC) commands used to suspend or resume data transfer on the LU-LU session are quiesce commands. Either the PLU or the SLU application programs can issue quiesce commands.

There are three quiesce commands:

- Quiesce at End of Chain (QEC), which requests the receiver of this message to stop sending data after sending the last element in the chain. (A data chain is a series of related messages; see "Data Chaining" on page 3-33 for additional information).
- Quiesce Complete (QC), which is an acknowledgment to QEC saying that data transfer is now suspended. When QC is sent by the PLU, the controller prevents the station from sending any normal-flow messages until the Release Quiesce message is received.
- Release Quiesce (RELQ), which notifies the receiver that data may again be transferred.

Ending Sessions and Disconnecting

When data transfer is completed, the session may be terminated. The session must be terminated before this PLU or another PLU can establish a different session with the same SLU. The session may also be terminated if an unrecoverable error occurs.

SLU-Initiated Ending

You may decide that the SLU will request session termination. Termination is requested by the SLU by issuing the NS Terminate Self command for an immediate termination (SSCP-LU flow) or the Request Shutdown command for an orderly termination (LU-LU flow).

The SSCP/application program receives the NS Terminate Self command and checks whether the named application program is the one participating in this session. If it is, the SSCP/application program sends a positive response, may send a Clear (if allowed by the TS profile) which purges all messages on the LU-LU session, and an Unbind command which ends the session. Because the Terminate Self command is presented to the SSCP/application program as a network services command, the response must be written by the SSCP/application program and will be sent on the SSCP-LU flow.

PLU-Initiated Ending

The programmer may decide that sessions will be terminated by the PLU application program. In this case the PLU issues a Clear command (optional) followed by an Unbind command. The session can also be terminated abnormally by simply varying the SLU offline.

Ending the SSCP-LU Session

The SSCP-LU session is ended when an SLU is varied offline by the LCNTRL/Vary Offline command. Note that ending the LU-LU session has no effect on the SSCP-LU session. When an SLU is varied offline, the SSCP sends a Deactivate Logical Unit (DACTLU) command to the SLU and the SLU enters the offline state. The response to the DACTLU has no effect on the state; it simply clears the "response outstanding" condition and allows the SSCP-PU session to end, if this is the last SLU to go offline. If a negative response is received, it is written to the system log, the response requirement is satisfied, and the error is ignored.

If you vary the SLU offline before the end of the LU-LU session, the LU-LU session ends because it cannot exist without the SSCP-LU session. Any information on the LU-LU session at the time the SLU is varied offline is lost. The link control places any purged messages in the system log.

Ending the SSCP-PU Session

When the last SSCP-LU session ends for this PU, the SSCP-PU session ends. The SSCP sends a Deactivate Physical Unit (DACTPU) command to the PU and the SSCP-PU session ends. The response to DACTPU has no effect on the session except that it allows the disconnect to flow.

If a negative response is received, the response requirement is satisfied; the response is written to the system log, and the error accompanying the negative response is ignored.

Disconnecting

When the SSCP receives a response to DACTPU, it sends a Set Disconnect Response Mode (SDRM) SDLC command to the SDLC station. The ALA link waits as long as specified in CPGEN for a response. Either an NSA response or a three-second time-out removes the SDLC station from the poll list.

An application program can cause an immediate disconnect by varying the SDLC station or control unit (NID specified in the ALACU macro) offline. In this case, all SNA sessions are ended abruptly and an SDRM is sent to the SDLC station. When this occurs, all stations that previously received a ready indication are posted with loss of contact.

Minimum Requirements for Transferring Data

The minimum processing required to transfer data between the PLU application program and the SLU includes session initiation by the PLU application program, data transfer, and session termination by the PLU application program. The PLU must:

- Receive the Ready indication.
- Send the Bind command and read a positive response.
- Send the Start Data Traffic command and read the response (if required).
- Read the data sent by the SLU, process the data, write any required responses, and issue an LEXIT instruction to wait for additional data.
- Set the write type and flag fields (AMSAWCT and AMSAWCF), set the device's NID into AMSNID, put the data to be sent in an output area, and issue an LWRITE AL.

To end the session, the PLU must send the Unbind command and read the response.

Logical PLU Session States

The controller maintains internal indicators for each SLU which determines the PLU's ability to communicate with the SLU. If your program attempts to send a message type not currently allowed for the PLU, the controller rejects the write request and returns status bits in SMSDST. The bits set in SMSDST are shown later in this chapter. Figure 3-7 shows state changes for various commands and for responses to commands that are sent or received by the PLU. The PLU enters a given state when the described condition occurs.

	Data Flow Reset State:		Session State:		Quiesced State:	
	IN	OUT	IN	OUT	IN	OUT
Loss of Contact	N/A	N/A	•——	——>	N/A	N/A
Bind Rsp Received	•		<——	——•	N/A	N/A
TS 3 & 4 Bind Rsp Received		•	<——	——•		•
TS 2 & 7 SDT Rsp Received	•——	——>	•			•
TS 3 & 4 QC Rq Sent		•	•		<——	——•
RELQ Rq Received		•	•		•——	——>
Clear Rq Sent	<——	——•	•		N/A	N/A
Clear Rq Received	——	——>	•			•
TS 2 Unbind Rq Sent or Received	N/A	N/A	•——	——>	N/A	N/A

• ———Beginning state (with arrow), or state unchanged
<———State changes
N/A ——State not applicable

Figure 3-7. PLU State Changes

Sequence Numbering

For PU-T2 devices, all normal flow messages transmitted on the LU-LU session-flow between the PLU and SLU are sequence-numbered. A sequence number is maintained for the PLU-to-SLU flow and another sequence number for the SLU-to-PLU flow. Each message is given a sequence number one greater than the sequence number of the preceding normal-flow message. For expedited-flow messages on the LU-LU flow and for all messages on the SSCP-LU and SSCP-PU flows, identifiers are used instead of sequence numbers.

Sequence numbers are set to 0 in the network when a session is established for when a Clear message is sent. The sequence number or ID of each message read by your program is passed in AMSASQ. The sequence number or ID of each message sent by your program is also passed in AMSASQ. The sequence numbers on the LU-LU session can be changed by the PLU using the Set and Test Sequence Numbers (STSN) command, if this command is allowed by the TS profile.

When the PLU is reading and encounters a sequence number error, it sends a negative response (if a response was requested) with sense code X'2001' to the SLU, and the LREAD is terminated with a data check.

When any response is read by your program, the response sequence number is set by the controller in AMSASQ. Note that a sequence number or ID is stored for every response. When a response is written by your program, the controller sets the correct sequence number in the message, and returns the number to your program in AMSASQ. Sequence numbers are not used for PU-T1 devices.

If the PLU or SLU encounters an unrecoverable error (for example, loss of a line), the LU-LU session may need to be resynchronized after being restarted. This process includes returning to the last recoverable messages and optionally resetting the sequence numbers accordingly. The application programs may include routines to retransmit lost messages.

The STSN message is sent by the PLU application program (AP). The STSN may contain either a new PLU sequence number for a write, a new SLU sequence number for a read, or both. It also contains a set of flags for each sequence number. The flags may be set so that a sequence number is ignored or set by the network, requested from the SLU, or set in the network with a request for verification by the SLU. Five bytes of data are always returned on the response to STSN unless the STSN command contained two 2-bit action codes defining two sets, two ignores, or a combination of set and ignore. In these cases, a positive response without data may be received.

When a session is restarted and resynchronized, the PLU application program will send clear, STSN, and SDT. When the STSN command is sent, a dialog may occur to establish sequence numbers acceptable to both the SLU and the PLU; the dialog consists of a series of STSN commands and positive responses.

If the SLU discovers that resynchronization is required, the SLU may send either a Request Recovery command, a negative response, or an LU Status command with a description of the failure in the sense bytes. If the PLU application program discovers the failure or receives a Request Recovery command from the SLU, the PLU should send a Clear to purge all messages from the network, an STSN to establish new sequence numbers, and then an SDT.

Pacing

Outbound Pacing

Outbound pacing is a protocol that gives the SLU control over the number and frequency of normal flow messages it receives from the PLU on the LU-LU session. The specification of pacing values is performed during CPGEN, and can be altered when the LU-LU session is established. When the first message of the sequence is sent, a bit is set in the request header indicating that a pacing response is to be returned. If the pacing count is exhausted before a pacing response is read by the ALA link, the PLU cannot send additional data messages. If a write is issued and a pacing response has been requested and not received, the station is to be deferred until the response is received.

Inbound Pacing

When a message is received from an SLU, it is placed in a queue for that SLU. As your program reads these messages, the pacing indicator is checked for each LU-LU normal flow message. If the pacing indicator is detected and this is the last message in the queue, an Isolated Pacing Response (IPR) is generated. If the Queue is not empty, an IPR is not generated until the last message in the queue is passed to your program.

An example best shows how this algorithm works. If a value of 3 was specified in byte 8 of the Bind parameters and the SLU supports pacing, the pacing count in the SLU is set to 3. This means that up to 3 messages will be sent before the SLU waits for a pacing response—in this case, an IPR.

When the first normal flow message arrives, the pacing indicator is on, the queue is probably empty, and an IPR is generated when the PLU reads. This allows 5 more messages to flow. If these messages arrive at a slow enough rate, the queue is always empty and IPRs are returned immediately, causing no slowdown of the data flow. If the messages arrive rapidly, the queue does not remain empty, and an IPR is not returned to the SLU.

When the queue is empty, an IPR is generated. Because the next message received carries the pacing indicator and the queue is now empty, a second IPR is generated allowing up to 5 more messages to flow. Hence, specifying the value 3 for inbound pacing allows bursts of 5 messages to flow. Generally, if "*n*" equals the value specified in the Bind parameters, then groups of *n* +(*n* -1) or 2 *n* -1 messages will flow (for example, 1, 3, 5, 7, 9, etc.).

Message Segmenting

Segmenting is the dividing of a large basic information unit (BIU) into smaller portions for transmission as separate basic link unit (BLU) messages.

Segmenting applies to data on any session flow. The ALA link supports segments from 20 bytes to 500 bytes for both inbound and outbound segmenting; however, outbound segment sizes of 128 bytes or greater are recommended to ensure that your program receives a Bind response.

Outbound Segmenting

Segmenting is performed automatically by the ALA link, based on the 'SEGMENT' parameter in the ALATERM macro. When a message is written by your program which exceeds the parameter value, the message is segmented. Each element of the segment contains the same TH; the first segment contains the RH. If insufficient buffers are available to write all of the segments, the work station is placed in the defer state until sufficient buffers are available.

Inbound Segmenting

If a segmented message is received, it is passed to your program as if it were a dynamically buffered message. In other words, the receiver reassembles the message in your segment without indication that the message was segmented. You should be aware that if inbound segmenting is used, the number of the ALA link input buffers may have to be increased. If you receive a segmented request that is less than 20 bytes, the first segment is written to the system log, an error is recorded for the station, and the connection with the device is ended.

Using the AMS Read and Write Flags

The indicators in the read and write flag fields (AMSARCF and AMSAWCF) are used for change-direction and bracket protocols, data-chaining, response protocols, and code-selection. The PLU application program tests the bits in AMSARCF after executing an LREAD AL instruction and sets the indicators in AMSAWCF before issuing an LWRITE AL instruction. The change-direction, bracket indicators, and the code selection indicator can be used with any message on the LU-LU session. Begin Bracket (BB) and End Bracket (EB) should only be set on first-in-chain requests; Change Direction (CD) should only be set on last in chain requests. Chaining and response-protocol indicators (first-in-chain only) are used with data or data-and-control (they are ignored for all commands). All indicators in AMSAWCF are ignored when a response is written.

Change Direction Protocol

The change direction (CD) indicator is used with the half-duplex flip-flop protocol and, optionally, with the half-duplex contention protocol. The ALA link transports this indicator for use by the PLU that is responsible for enforcing the half-duplex protocols. A change-direction indicator signifies to the receiver that he is to begin transmitting.

Assume, for example, that all data transmission is initiated by the SLU. The SLU begins transmitting messages that completely describe a transaction; with the last message, the SLU sets the change-direction indicator to tell the PLU application program that it may now begin transmitting a reply.

If the PLU application program needs additional information to complete the transaction, it sends the inquiry and sets the change-direction indicator. The SLU then replies to the inquiry and again sets the change-direction indicator. The dialogue continues in this half-duplex mode until the transaction is completed. (During a half-duplex dialogue, the SLU can use the Signal message to indicate that the PLU application program should stop sending data, and change the direction of data flow.) Consult the *IBM SNA Format and Protocol Reference Manual, Architectural Logic* (SC30-3112) for a more detailed explanation.

Bracket Protocol

Brackets are an optional protocol that can be used to give the PLU or SLU control of data transmission, to indicate that a monologue or dialogue is about a single subject, and to prevent unrelated messages from being transmitted during the monologue or dialogue. The monologue or dialogue is referred to as a bracket; the first message in the bracket is accompanied by a begin-bracket (BB) indicator, and the last message in the bracket is accompanied by an end-bracket (EB) indicator. A single message can be a bracket if both the BB and EB indicators are set. Bracketing can also be done in chained messages, as described later in "Data Chaining" on page 3-33.

If brackets are used in a session, the Bind parameters specify one of the LUs as first speaker and the other as bidder. The first speaker has the freedom to begin a bracket without requesting permission from the other LU to do so. The bidder must request and receive permission from the first speaker to begin a bracket.

Bid is a normal-flow DFC request issued by the bidder to request permission to begin a bracket. A positive response to Bid indicates that the first speaker will not begin a bracket, but will wait for the bidder to begin a bracket.

A negative response to Bid indicates that the first speaker has denied permission for the bidder to begin a bracket. A Ready to Receive (RTR) command may be sent later by the first speaker when permission to start a bracket is granted. If the first speaker will send RTR later, the sense data with the negative response to Bid is, "Bracket Bid Reject--RTR Forthcoming." The bidder has the option of waiting for RTR or sending Bid again. If the RTR will not be sent, the sense data is "Bracket Bid Reject--No RTR Forthcoming." In the latter case, the bidder must send Bid again if it still wants to begin a bracket.

Instead of sending Bid followed by first-in-chain with BB, the bidder may attempt to initiate a bracket by sending just first-in-chain with BB. The first speaker grants the attempt (via positive response) or refuses it (via negative response indicating either Bracket Bid Reject--RTR Forthcoming or Bracket Bid Reject--No RTR Forthcoming). However, if the bidder terminates the chain that carries BB by sending Cancel, then the bracket is not initiated no matter what the response.

RTR may also be issued by the first speaker to find out if the bidder wants to begin a bracket. A positive response to RTR indicates that the bidder will initiate the next bracket. If the bidder does not want to initiate a bracket, it issues a negative response with the sense code, "RTR Not Required."

For example, assume that a transaction is being processed by the SLU. The first message sent to the PLU is accompanied by a begin-bracket indicator. During the dialogue, a subroutine of the PLU application program is told to send a message to all SLUs, advising that terminals will be shutdown in 15 minutes.

The subroutine sends a Bid message to all SLUs. The SLUs not in a bracket send a positive response, and they immediately receive the shutdown message as a bracket (both BB and EB indicators set); the SLU processing the transaction sends a negative response, which is noted by the subroutine. When the SLU sends the last message of the transaction, it includes an end bracket. The SLU then sends RTR, which is passed to the PLU subroutine; the subroutine then sends the shutdown message.

Refer to the *IBM SNA Format and Protocol Reference Manual, Architectural Logic* (SC30-3112) for a more detailed explanation.

Data Chaining

Data chaining is an optional protocol that can be used to transmit a group of related messages. When the PLU is sending messages (LU-LU flow), the chaining indicator is set to 1 to indicate the first message in a chain. It is set to 0 to indicate the middle messages in the chain and then set to 1 again to indicate the last message in the chain. The chaining indicator is also set to 0 to indicate a nonchained message (a single message "chain"). When the PLU is receiving messages, the chaining indicator is 1 for the first or last message and 0 for the middle message or a nonchained message (the status bits indicate whether the message is first or last in a chain).

When a chain is being sent to an SLU, the PLU application program may send a Cancel message to indicate that an error was detected in one of the chained messages. All elements of the chain received by the SLU should be discarded. If the SLU sends a negative response to any element of a chain, the PLU should terminate the chain normally or by sending Cancel.

You can combine bracket protocol with data chaining. A bracket is delimited by use of Begin Bracket (BB) in the first request of the first chain, and End Bracket (EB) in the last request of the last chain in the bracket.

The PLU application program may receive a Cancel message to indicate all elements of the chain previously received should be discarded. The PLU can send a negative response to a chain. Any messages received after the response is written should be discarded.

The type of response used for a given chain is provided by the user with the first chain. The AMSAWCF field is then ignored until a following first-in-chain or only-in-chain request is made. Only chains of the following three types can be sent:

- No-response chain: Each request in the chain is marked no-response.
- Exception-response chain: Each request in the chain is marked exception-response.
- Definite-response chain: The last request in the chain is marked definite-response; all other requests in the chain are marked exception-response.

Refer to the *IBM SNA Format and Protocol Reference Manual, Architectural Logic* (SC30-3112) for a more detailed explanation.

ALA/AMS Control Fields and Indicators

The following contains definitions of the AMS field values used when the 4700 controller communicates using the alternate line attachment. Not discussed is the sequence number field (2 bytes of the AMSASQ field) which was discussed under "Message and Command Types" on page 3-11 and "Sequence Numbering" on page 2-27. All values in AMSARCT and AMSAWCT are given in hexadecimal values.

Read Type Field (AMSARCT) Codes

Hex Value:	**Meaning:**
01	Positive DR1 response
02	Positive DR2 response
03	Positive DR1 & DR2 response
05	Negative DR1 response
06	Negative DR2 response
07	Negative DR1 & DR2 response
09	Positive (DR1) response **
0D	Negative (DR1) response **
10	Data
11	Data and control (that is, FM header present)
12	Data (SSCP-LU session only)
15	Network Services Command (NS Command)
16	Session Initiate Response Parameters
20	* Chase
21	* Cancel
22	* Quiesce Complete (QC)
23	* Logical Unit Status (LUSTAT)
24	* Ready To Receive (RTR)
25	* Bracket Initiation Stopped (BIS)
29	* Logical Unit Status (SSCP-LU session only)
40	* Quiesce at End of Chain (QEC)
41	* Request Shutdown (RSHUTD)
42	* Shutdown Complete (SHUTC)
43	* Release Quiesce (RELQ)
44	* Signal (SIG)
45	* Stop Bracket Initiation (SBI)
82	* Request Recovery (RQR)
A1	* Unbind session

* -- Response generated by ALA.

** -- Set for those write requests where DR1 is automatically requested (see "Write Type Field (AMSAWCT) Codes" on page 3-37).

Read Flags Field (AMSARCF) Codes

Response Protocol Requested (3 bits)

nnnn n000	No response
nnnn n001	Definite response protocol (DR1)
nnnn n010	Definite response protocol (DR2)
nnnn n011	Definite response protocol (DR1 & DR2)
nnnn n100	Not allowed
nnnn n101	Exception response protocol (DR1)
nnnn n110	Exception response protocol (DR2)
nnnn n111	Exception response protocol (DR1 & DR2)

Chaining Indicator (1 Bit)

nnnn 0nnn	Normal read or middle of chain
nnnn 1nnn	First or last in chain

Bracket, Direction, and Code Indicators (4 Bits)

nn00 nnnn	Middle of bracket
nn10 nnnn	Beginning of bracket
nn01 nnnn	End of bracket
nn11 nnnn	Only in bracket
n0nn nnnn	No change of direction
n1nn nnnn	Change direction indicator
0nnn nnnn	Not alternate code
1nnn nnnn	Alternate code

The following read control flags are in extension field AMSCARCE:

nnnn nnn0	LREAD AL search indicator (stop search)
nnnn nnn1	LREAD AL search indicator (continue search)
nn0n nnnn	Unpadded data
nn1n nnnn	Padded data
n0nn nnnn	Unexpedited message
n1nn nnnn	Expedited message
0nnn nnnn	Unenciphered data
1nnn nnnn	Enciphered data

| *Write Type Field (AMSAWCT) Codes*

Hex Value:	Meaning:
03	Positive response
07	Negative response
10	Data
11	Data and control (FM header present)
12	* Data (SSCP-LU session only)
15	* Network Services Command (NS Command)
20	* Chase
21	* Cancel
22	* Quiesce Complete (QC)
23	* Logical Unit Status (LUSTAT)
24	* Bid(BID)
25	* Bracket Initiation Stopped (BIS)
40	* Quiesce at End of Chain (QEC)
41	* Shutdown (SHUTD)
43	* Release Quiesce (RELQ)
44	* Signal (SIG)
45	* Stop Bracket Initiation (SBI)
80	* Clear
81	* Set and Test Sequence Numbers (STSN)
82	* Start Data Traffic (SDT)
A0	* Bind session
A1	* Unbind session

* -- DR1 response automatically requested with only-in-chain request.

Write Flags Field (AMSAWCF) Codes

Response Protocol Requested (3 bits)

nnnn n000	No response protocol
nnnn n001	Definite response protocol (DR1)
nnnn n010	Definite response protocol (DR2)
nnnn n011	Definite response protocol (DR1 & DR2)
nnnn n100	Not allowed
nnnn n101	Exception response protocol (DR1)
nnnn n110	Exception response protocol (DR2)
nnnn n111	Exception response protocol (DR1 & DR2)

Chaining Indicator (1 Bit)

nnnn 0nnn	Normal write or middle of chain
nnnn 1nnn	First or last in chain

Bracket, Direction, and Code Indicators (4 Bits)

nnn1 nnnn	End of bracket
nn1n nnnn	Beginning of bracket
n1nn nnnn	Change direction
0nnn nnnn	Not alternate code
1nnn nnnn	Alternate code

The following write control flags are in extension field AMSAWCE:

nn0n nnnn	Unpadded data
nn1n nnnn	Padded data
0nnn nnnn	Unenciphered data
1nnn nnnn	Enciphered data

Message Headers

The following abbreviations are used in the subsequent header definitions:

WR RQ: Request Written
WR RSP: Response Written
RD RQ: Request Read
RD RSP: Response Read

Physical Unit-Type 2 (PU-T2) Transmission Header

Format Identification Field (FID)

WR RQ: Obtained from CPGEN ALACU PUTYPE.
WR RSP: Obtained from TH in last message read.
RD RQ: Must equal CPGEN ALACU PUTYPE.
RD RSP: Must equal CPGEN ALACU PUTYPE.

Mapping Field

WR RQ: (Set based on the ALATERM SEGMENT during CPGEN).
WR RSP: Binary 'XXXX11XX' ("X" is either 0 or 1).
RD RQ: Used to reassemble segmented messages.
RD RSP: Must be binary 'XXXX11XX' ("X" is either 0 or 1).

Expedited Flow Indicator (EFI)

WR RQ: Set based on AMSAWCT.
WR RSP: Obtained from the transmission header (TH) in last message read.
RD RQ: Used to arrange the input queue.
RD RSP: Used to arrange the input queue.

Destination Address Field (DAF)

WR RQ: Obtained from the ALATERM SEL parameter during CPGEN for SSCP-LU or LU-LU flow or set to zero for SSCP-PU flow as determined from AMSAWCT .
WR RSP: Obtained from the TH OAF in the last message read.
RD RQ: When used, identifies the session.
RD RSP: When used, identifies the session.

Origin Address Field (OAF)

WR RQ: Set to ID of station sending message or to zero if SSCP-PU/LU is as determined from AMSAWCT.
WR RSP: Obtained from the TH DAF in the last message read.
RD RQ: If nonzero, routes the request to the correct work station.
RD RSP: If nonzero, routes the response to the correct work station.

Sequence Number Field (SNF)

WR RQ: Assigned by the controller from internal counters.
WR RSP: Obtained from the TH in the last message read.
RD RQ: Checked on LU-LU normal-flow and always placed in AMSASQ.
RD RSP: Placed in AMSASQ.

Request/Response Header (RH)

Request/Response Indicator (RRI)

WR RQ: Set to 0 as determined from AMSAWCT.
WR RSP: Set to 1 as determined from AMSAWCT.
RD RQ: Indicator mapped into AMSARCT value.
RD RSP: Indicator mapped into AMSARCT value.

Format Indicator (FI)

WR RQ: Set based on AMSAWCT.
WR RSP: Obtained from RH in last message read.
RD RQ: Used to identify NS commands on the SSCP-PU/LU flow and is mapped into AMSARCT value.
RD RSP: Used to identify NS command responses.

Sense Data Included Indicator (SDI)

WR RQ: Not used.
WR RSP: Set based on X'07' value in AMSAWCT.
RD RQ: Not used.
RD RSP: Not used.

Chaining Control

WR RQ: Set based on AMSAWCF for X'10' or X'11' value in AMSAWCT; for other AMSAWCT response values, set to binary 'XXXXXX11', where "X" is either 1 or 0.
WR RSP: Set to binary 'XXXXXX11', where "X" is either 1 or 0.
RD RQ: Mapped to AMSARCF with SMSDST indicating first or last in chain.
RD RSP: Must be set to binary 'XXXXXX11', where "X" is either 1 or 0.

Form of Response Requested

WR RQ: Set based on AMSAWCF for X'10' or X'11' value in AMSAWCT; for other AMSAWCT request values, set to DR1.
WR RSP: Set based on RH in last message read and AMSAWCT.
RD RQ: Placed in AMSARCF.
RD RSP: Mapped to AMSARCT.

Queued Response Indicator (QRI)

WR RQ: Not used.
WR RSP: Obtained from RH in last request read.
RD RQ: Not used.
RD RSP: Should not be on.

Response Type

WR RQ: Set based on AMSAWCF for X'10' or X'11' value in AMSAWCT; for other AMSAWCT request values, set to zero.
WR RSP: Set based on RH in last message read and AMSAWCT.
RD RQ: Placed In AMSARCF.
RD RSP: Mapped to AMSARCT.

Pacing

WR RQ: Set based on the ALATERM PACING or Bind pacing parameter during CPGEN.
WR RSP: Set to zero.
RD RQ: Used to generate an IPR.
RD RSP: Resets outbound pacing count.

Bracket Control, Change Direction, and Code Selection

WR RQ: Set based on AMSAWCF.
WR RSP: Set to zero.
RD RQ: Placed in AMSARCF.
RD RSP: Must be zero.

Request/Response Unit

WR RQ: For commands, AMSAWCT value translated to first byte; for data, RU taken from user segment.
WR RSP: No RU required.
RD RQ: If command, first byte translated and placed in AMSARCT; data placed in user segment.
RD RSP: Data placed in user segment.

Chapter 4. The 4700 Assembler Communication Instructions

These are the 4700 assembler instructions you must use in your application program to transmit data to, and receive data from, the host or ALA device. They start or stop the link, read data from and write data to link devices, control the alternate SNA/SDLC link (ALA), and check read and write operation status. The instructions are:

- ASSIGN—Defines work station and LU addresses.
- LCHECK AL and LCHECK CP—Checks the ALA host link status.
- LCNTRL—Controls the ALA SNA/SDLC link.
- LREAD AL and LREAD CP—Reads data from the ALA or host link.
- LWRITE AL and LWRITE CP—Sends data to the ALA device or host.
- STPLNK—Stops the SNA/SDLC host link.
- STRLNK—Starts or wrap test the SNA/SDLC host link.

Each instruction descriptions in this chapter begins with a general introduction to the instruction followed by the coding syntax, the operand descriptions, condition codes set by the instruction and their meanings, and any possible program checks.

For coding and syntax rules, refer to the *4700 Controller Programming Library, Volume 1.*

ASSIGN—Define Station/LU Addresses

By specifying the LUASSIGN option on the COMLINK CPGEN macro and setting the parameter list as described here and in Figure 4-1 on page 4-4, you may use the ASSIGN instruction to:

- Assign a specified logical unit (LU) address to a specified station or LU pool.
- Determine LU address-station assignments.

The communication link, station ID, and LU address are specified in a parameter list you first define, then select in the ASSIGN operand. Refer to Figure 4-1 on page 4-4 for a definition of the LU address parameter list format and settings.

For a description of how to include ASSIGN instruction LU address options, refer to the COMLINK macro description in Volume 6 of this programming library.

```
Name     Operation  Operand

                    ⎧ defld2     ⎫
[label]  ASSIGN     ⎨ seg2,disp2 ⎬
                    ⎪ (reg2)     ⎪
                    ⎩ (defrf2)   ⎭
```

operand 2
Defines the start of the parameter list. A DEFRF instruction label must always be in parentheses. The length of this operand is ignored and the first 5 bytes are assumed to be the parameter list. The parameter list is illustrated in Figure 4-1 on page 4-4. The segment number cannot be 14.

Condition Codes: When ASSIGN completes, the controller sets one of the following:

Hex Code:	**Explanation:**
01	The assignment was successful.
02	Unsuccessful assignment: the work station has an LU address assigned to it or the specified LU address is currently busy.
04	One of the following has happened: 1. The COMLINK macro option LUASSIGN was not specified. 2. The optional ASSIGN LU module was not loaded. 3. NO CPU asynchronous program entry point exists. 4. This station is not configured to use the communication link. 5. The communication link is non-SNA/SDLC link.
08	Parameter list error (station ID or LU address, A/B field, or divide specification). The station ID was specified as 0 or X‘FF’, and the LU address was also X‘FF’.

Program Checks: 1, 2, or 27 can be set.

Programming Notes: When using ASSIGN for LU addresses, the following points apply:

1. Specifying a station ID of zero (0) in the parameter list reassigns the LU address in the LU/LDA field to the LU pool. The station ID to which the LU address was previously assigned is set into the station ID parameter field. This ID will be zero (0) if the LU address is already in the LU pool.
2. Specifying a station ID of X'FF' in the parameter list returns the station ID to which the LU address in the LU/LDS field is now assigned, to the station ID field. This ID will be zero (0) if the LU address is in the LU pool.
3. If the LU address in the LU/LDA field is X'FF', the ASSIGN instruction returns the LU address currently assigned to the station ID specified. If ASSIGN returns X'FF' in the LU/LDA field, the specified station ID is not assigned an LU address.

LU ADDRESS

X'9'	X'0'	X'1'	X'0'	Shared Indicator	Station ID	LU

Byte: 0 1 2 3 4

Shared Indicator

Is the shared device indicator (C'A' or C'B'); if the device is not a shared device, this byte must contain a C'A'. This field is not used for SNA LU address functions.

Station ID

Specify the station ID to receive the desired LU address.

LU

Is the 8-bit binary number specifying the LU address requested by the related station ID.

Figure 4-1. Parameter List Used by ASSIGN

LCHECK AL—Check ALA Link Status

This form of LCHECK determines the status of an SLU and synchronizes data transmission between the SLU and the application program. You use it to determine the completion (and associated) status of all outstanding write operations or only those write operations to a specific device.

To check the status of all writes or to determine the current state of a specific device, place the device ID in AMSNID before issuing LCHECK AL. If you set AMSNID to '0', this checks the status of all outstanding writes issued by the station to owned devices.

Status indicating a change of device state has priority over the LWRITE AL status. For example, if Loss of Contact for a control unit occurs before the LWRITE returns status, the X'0201' loss of contact status occurs before the prior LWRITE status.

If a station issues a specific LCHECK without the TIO option, and LWRITE AL operations to the selected ALA device are not complete, the work station enters the wait state until either all outstanding writes to that device complete or one of the outstanding writes returns ending status.

A general LCHECK AL operates identically with a specific LCHECK AL except that all outstanding writes to *all* devices must complete or any one of the outstanding writes must return ending status before the controller releases the station from wait state.

If you specify the test I/O (TIO) option, outstanding data does not place the station in wait state.

The syntax for LCHECK AL is:

Name:	Operation:	Operand:
[label]	LCHECK	AL [,TIO]

AL

Checks the status of one or more write operations to the SLU.

TIO

Performs a test I/O operation to find if any SLU I/O operations are still pending. The application program remains in control whether the I/O operation being checked has been completed or not.

Condition Codes: One of the following is set:

Hex Code:	**Explanation:**
01	All I/O operations to the specified network ID have been completed and no status is pending. (Zero status is returned.)
02	I/O has completed and nonzero status was returned. The status code is in SMSDST, the write event number is in AMSAFN (if the status pertains to a write operation), and the network identification is in AMSNID. If LCHECK AL returns this condition code and multiple LWRITE operations are outstanding, issue another LCHECK to ensure that you have received all outstanding status.
04	I/O is not complete (applies only for the TIO option).

Program Checks: 9 can be set.

Possible Status: See Appendix D, "Link Status Codes."

Programming Notes

1. If LCHECK AL produces device status, it may return control to the program when "exceptional condition" status occurs, and before all the outstanding write operations are completed. In this case, issue another LCHECK AL to receive any outstanding status.

2. If LCHECK AL specifies the test I/O (TIO) option, the controller sets a condition code indicating whether or not any outstanding I/O operations are still pending.

3. If the status presented is for a prior LWRITE operation, LCHECK AL sets the condition code to X'02', turns on the "prior operation" indicator (bit 03) in SMSDST, and puts the Write Event Number of the failing LWRITE operation in AMSAFN.

4. Status caused by a change of device state does not set the prior operation indicator in SMSDST or change AMSAFN.

LCHECK CP—Check Status of Host LWRITE Operations

LCHECK CP synchronizes data transmission and determines status by checking all outstanding write operations to the host system. If all operations are not completed, LCHECK causes the work station to wait for their completion. When the operations are completed, the condition code is set, the logical station is taken out of the wait state, and a status code is stored in SMSDST. If there were no outstanding write operations, SMSDST is set to zero status.

Name	Operation	Operand
`[label]`	`LCHECK`	`CP[,TIO]`

CP

Specifies that write operations to the host system are to be checked.

TIO

Specifies that the appropriate status is to be returned and the condition code set to reflect that status of the last LWRITE CP issued. The application program retains control whether the LWRITE CP has been completed or not.

Condition Codes: When LCHECK completes, the controller sets one of the following:

Hex Codes:	**Explanation:**
01	Zero status is in SMSDST.
02	Nonzero status is set in SMSDST. (See Appendix D, "Link Status Codes" for an explanation of the status codes.)
04	LWRITE CP has not been completed (applies only to the TIO option).

Program Checks: None are set.

LCNTRL—Alternate (ALA) Link Control

The LCNTRL instruction performs nine distinct ALA link operations, depending on the LCNTRL operand you specify. This instruction description, therefore, differs from others in this chapter by beginning with a general introduction to LCNTRL and each operation it performs. Following the introduction are the individual LCNTRL instruction descriptions, each in the normal format used in describing other instructions in this chapter. Each description also contains status codes because the descriptions Appendix D, "Link Status Codes," do not distinguish between status codes for each LCNTRL operation.

Introduction

This LCNTRL description introduces each LCNTRL operand, and the function that each performs.

LCNTRL controls the 4700 system access to a device connected to the controller on the ALA link. The specific operation depends on the LCNTRL operand you specify. The following are the available LCNTRL operations:

ASSIGN: Establishes, transfers, or relinquishes ownership of an ALA link device.

START LINE: Activates the ALA link, or starts diagnostic tests.

STOP LINE: Deactivates the ALA link immediately.

VARY ONLINE: Establishes a logical connection between the device and the work station and, in general, begins polling.

VARY OFFLINE: Removes the logical connection between the device and the work station and can end polling.

STOP SOLICITING DATA: Causes incoming data from the ALA device on a line to be stopped temporarily. The specified ALA line is set to quiesce state.

RESUME SOLICITING DATA: Allows data transmission to resume on the ALA line. The specified ALA line is removed from the quiesce state.

ALTER PHYSICAL ADDRESS: Allows the changing of the physical select address of an SLU.

SENSE: Retrieves data relating to the current state of an ALA device.

General LCNTRL Programming Notes

1. Certain LCNTRL operations cause action to occur on a total line basis. The operations: Start Line, Stop Line, Stop Soliciting Data, Resume Soliciting Data, and Alter Physical Address may be issued by any work station configured for the ALA link, even though that station may not own all of the ALA devices on that line. (A station is not permitted to own a line.) These operations are defined in the following descriptions.

2. The LCNTRL operations, with the exception of Start Line, Stop Line, and Sense, are part of a separate optional module. It is possible to create a system that uses the link and does not contain these optional LCNTRL operations. If the application attempts to execute any of the optional LCNTRL functions without the required optional module in the system, a status of X'0404' is returned in SMSDST.

LCNTRL ALP (Alter Physical Address)

LCNTRL ALP changes the logical destination address field (DAF) of the SLU. The replacement DAF is contained in a variable length parameter list. LCNTRL ALP points to this list. The AMSNID field is not used. The parameter list is in the following format:

```
 ------------------------------------------/    /-----------------
|     ENTRY 1     |      ENTRY 2      |        ...       | ENTRY n |
 ------------------------------------------/    /-----------------
 Bytes 0 - 5         Bytes 6 - 11       Bytes 12 - 71      72
```

```
Each entry has the following format:
 -------------------------------------------------------------------
|     NETWORK ID     |          PPA          |     NEW PSA (DAF)     |
 -------------------------------------------------------------------
      Bytes 0 - 1          Bytes 2 - 3              Bytes 4 - 5
```

where:

DAF Specifies the new logical ID of the SLU (left-justified in byte 0).

The coding syntax for LCNTRL ALP is:

Name	Operation	Operand
[label]	LCNTRL	ALP, { defld1 seg1 seg1,disp1,len1 (reg1) (defrf1) }

ALP
Issues an "alter physical address" to the controller or device defined by bytes 0 and 1 of the parameter list.

operand 1
Is the location of the parameter list. When you specify *seg1*, the controller creates a two-byte machine instruction. All other operand 1 forms create a machine instruction that is six bytes long.

Condition Codes: One of the following is set:

Hex Code:	**Explanation**:
01	LCNTRL ALP executed successfully.
02	LCNTRL ALP returned status in SMSDST.

Note: If status is returned, SMSIML indicates the displacement from the start of the parameter list of the entry causing the returned status. All preceding entries in the list were updated.

Program Checks: 1, 2 or 9 might be set

Possible status: 0401, 0402, 0403 or 040B might result.

LCNTRL ALP Programming Notes

1. The SLU specified must be in the Offline state.
2. The SLU need not be owned by the work station issuing this instruction.
3. The change remains in effect until another LCNTRL/ALP instruction is issued or the system is restarted.
4. Unpredictable results occur if any of the specified address duplicates an existing address in the system.

LCNTRL ASN (ALA Device Assignment)

LCNTRL ASN is used to assign an ALA link device to a logical work station from the free device pool, to the ALA link free device pool from the owning station, or to another station from the owning station. The instruction may also be used, if the force option is specified, to assign device from one work station to another work station. The device ID and the work station ID are specified in a parameter list. LCNTRL ASN points to this list.

If the operation is completed successfully, LCNTRL ASN changes the station ID field in the parameter list to show the previous assignment of the ALA device. To assign the device to its previous state (that is, work-station ownership or the free device pool), issue a second LCNTRL ASN instruction using the same parameter list. (The list updated by the first LCNTRL ASN.) The parameter list is in the following format:

	NETWORK ID (NID)	STATION ID
BYTE	0 1	2

where:

NETWORK ID:	Specifies the logical NID of the ALA device.
STATION ID:	Specifies the receiving station ID, expressed as an 8-bit binary number.

The coding syntax for LCNTRL ASN is:

Name	Operation	Operand
[label]	LCNTRL	ASN,seg1 [,FORCE]

ASN

Specifies an assignment request

seg1

Is the segment containing the parameter list. The PFP must point to the start of the parameter list. The length associated with this segment specification must be at least 3 bytes.

The length of the parameter list is determined by either of the following:

1. The FLI if it is nonzero and less than or equal to the length between the primary field pointer and the end of the segment.
2. The length between the PFP and the end of the segment, if the FLI is 0 or greater than the length between the PFP and the end of the segment.

FORCE

Specifies that the assign function is to be performed if the specified ALA is currently owned by another work station or the assignment is to another work station from the free device pool.

Condition Codes: One of the following is set:

Hex Code:	**Explanation**:
01	The instruction was executed successfully.
02	Status is returned; the status code is contained in SMSDST.

Program Checks: 1, 2 or 9 might be set.

Program Status: 0401, 0402, or 0403 may result. Refer to Appendix B for status description and corrective action.

LCNTRL ASN Programming Notes

1. Specifying a station ID of '0' causes the ALA device to return to the free device pool.
2. Specifying a station ID of X'FF' causes LCNTRL ASN to return the ID of the owning station in the station ID field if the specified ALA device is currently owned. If it is not owned, '0' is returned. No actual assignment results from this instruction specification.
3. The device being assigned to the free device pool must be owned by the station issuing the LCNTRL ASN instruction.
4. A device may be assigned only from the free device pool to the work station issuing the LCNTRL ASN instruction, unless the FORCE option is specified. The FORCE option also permits assignment of a device from any work station to any other work station except to or from work station 1.
5. The parameter list cannot be in Segment 14.
6. AMSNID is not used, but is updated when the LCNTRL ASN instruction ends to show the Network ID from the parameter list.

LCNTRL RSD (Resume Soliciting Data)

LCNTRL RSD re-enables data input from the ALA line specified in AMSNID. If there has not been an earlier Stop Soliciting Data instruction active, this instruction has no effect on the line.

The coding syntax for LCNTRL RSD is:

Name	Operation	Operand
`[label]`	`LCNTRL`	`RSD`

RSD
Issues a "resume soliciting data" request to the controller or device identified by AMSNID.

Condition Codes: One of the following is set:

Hex Code:	**Explanation**:
01	The instruction was executed successfully.
02	Status is returned; the status code is contained in SMSDST.

Program Checks: 9 might be set.

Possible Status: 0402 or 0403 might result.

LCNTRL SENS (Sense)

LCNTRL SENS places 10 bytes of sense data plus '1' to 'n' bytes of additional statistical data in a user-specified segment. The network identification of the SDLC ALA device being queried (or '0' for the first device in the system) must be placed in AMSNID prior to issuing this instruction.

The first 10 bytes of sense data appear as follows:

	DEVICE STATE BYTE	AA	PPA	PSA	NETWORK IDENTIFICATION	X'03'	STAT ID
Byte	0	1	2-3	4-5	6-7	8	9

where:

DEVICE STATE BYTE Is the device state information. ALA support returns a byte of device state data with the following format:

1...	(X'80')	device is online
.1..	(X'40')	control unit is active
..1.	(X'20')	device is in poll list
...1	(X'10')	device is in select list
.... 1...	(X'08')	device is owned
.... .1..	(X'04')	line is started
.... ..1.	(X'02')	online status is pending
....1	(X'01')	loss of contact status is pending

The following are the definitions of each condition reported in the state byte:

ONLINE: The state of the device is either Online or Online Pending. The logical connection between the ALA device and the application program has been established. Offline signifies that there is no logical connection between this device and the application program if the device is currently owned.

CONTROL UNIT ACTIVE: This flag indicates that the control unit associated with the specified network ID is being polled.

DEVICE IN POLL LIST: The device associated with the network identifier specified is in the current poll list.

DEVICE IN SELECT LIST: The device associated with the network identifier specified is in the current select list.

OWNED: The device associated with the specified Network ID has been assigned to some station. (It is currently not in the free device pool.)

LINE STARTED: The line attaching the device associated with the Network ID has been started successfully.

ONLINE STATUS IS PENDING: Contact has been established with the control unit and, as soon as Online status is reported to the AP, the state of the ALA device will be changed from Online Pending to Online.

LOSS OF CONTACT STATUS PENDING: The ALA device was Online and a loss of contact was experienced with the control unit. This condition has not yet been reported to the application program.

Note: The Online Status Pending and Loss of Contact Status Pending bits may be set in combination. If this occurs, the establishment of contact and the loss of contact have occurred and both conditions are waiting to be reported to the program. You cannot determine the occurrence order from the device state byte but the order is reported to the program correctly.

AA	Is the Adapter Address of the ALA line associated with this device.
PPA	Is the physical poll address used by this device.
PSA	Is the physical select address used by this device.
NETWORK IDENTIFICATION	Is the logical network identifier assigned to this device.
X'03'	Indicates that this is an ALA device.
STAT ID	Is the station identifier of the owning station. If the device is in the free device pool this will be X'00'.

Status Represented by the Device State Byte: The "status" represented by Device State Byte 0 is not the same status as is in the SMSDST field, discussed in Appendix B. However, changes in the device state byte can also cause SMSDST status to be returned. Some of these states are discussed here.

State Change of Owned Device: Status indicating a change in the state of an owned device may be presented on an LREAD, LWRITE, or an LCHECK. The application program may avoid presentation of this status on an LCHECK or LWRITE operation by testing SMSAAP before issuing an LWRITE or LCHECK and then by issuing an LREAD, if appropriate.

Loss of Contact: This status indicates that ALA was unable to communicate with one or with all entities on a line. Loss of contact can be reported for the line, a control unit on a line, or for a terminal on a control unit. (A loss of contact on a write always refers to the control unit or line.) Loss of contact state is reported to the application program only if the program owns the entity to which the state applies and the NID previously entered the Online Pending or Online state.

When loss of contact has been reported, the application program can still perform read operations because data might have been pending when the loss of contact occurred. The application program should continue to issue reads until SMSDST status (X'400n' is presented. This signifies that no more data is available.

ALA reports loss of contact on every outstanding write operation that was affected by this failure. As a result, your AP should issue checks until a condition code of X'01' is received. Loss of contact occurs for every NID owned by the station that is affected by this condition. Loss of contact with a line results in the posting of loss of contact for each control unit on the line, and each terminal associated with each control unit on the line. Loss of contact for an individual control unit will result in the posting of loss of contact for each terminal associated with that control unit.

Vary Online and Online Pending Status: Online status is reported on an LCHECK, LWRITE, or LREAD instruction after a program issues Vary Online to a network entity that either cannot accept Vary Online, or was specified as active at CPGEN time. Online status occurring because of a Vary Online order to a terminal refers to the status of the terminal's control unit.

After a successful Vary Online, your program must determine the state of the terminal itself by testing the SMSDST status returned after either an LWRITE or an LREAD to the terminal. You can issue Vary Online even if the associated line has not yet been started. If the NID was specified as active at CPGEN time, the command is unnecessary unless you are testing for a line failure. When issued to an unstarted line, Vary Online may return either of the following status is SMSDST:

X'4002' when a Start Line has not yet been issued,

or,

X'4004' when a Start Line was issued but failed.

Issuing a Vary Online to a device that is not yet ready on a started line returns a status of either X'4001', indicating the Vary Online is pending; X'0800', indicating polling was previously started and the device is now online; or X'0200', indicating polling was previously started but loss of contact was experienced. When the Vary Online completes successfully, Online status is reported back to your program.

You may issue Vary Online either directly from your program, or indirectly by specifying the appropriate terminal as active during CPGEN. The status returned after an LREAD, LWRITE, or LCHECK of an entity on an unstarted line to which a Vary Online was already issued (either directly or indirectly) is either:

X'4003' when a Start Line failed,

or,

X'4002' when a Start Line has not yet been requested.

The coding syntax for LCNTRL SENS is:

Name	Operation	Operand
`[label]`	`LCNTRL`	`SENS,seg1 (,CLEAR )`

SENS
Specifies a sense request.

seg1
Is the number of the segment to receive the sense information. The primary field pointer (PFP) must point to the start of the field.

CLEAR
Resets the statistical counters associated with the secondary device being sensed to '0' unless X'0104' status is set.

The Sense data is stored in the specified segment starting at the PFP. The Sense operation terminates when any of the following conditions are satisfied:

1. When the end of the Sense and statistical data is reached.
2. When the end of the input field is reached (if the FLI is non-0 and less than or equal to the length between the PFP and the end of the segment).
3. When the end of the segment is passed (if the FLI is '0', or greater than the length between the PFP and the end of the segment).

At the end of the operation the PFP is unchanged and the length of the sense data is placed in SMSIML.

Condition Codes: One of the following is set:

Hex Code:	**Explanation:**
01	The instruction was executed successfully.
02	Status is returned; the status code is contained in SMSDST.

Program Checks: 1, 2, or 9 can be set.

Possible Status: 0104, 0105, 0402, or 0403 can occur.

LCNTRL SENS Programming Notes

1. The area specified must be at least large enough to contain the sense data (10 bytes). If the area specified is less than 10 bytes, X'0104' status is returned. If the area specified is greater than 10 bytes but less than is required to contain both the sense data and all the statistical counters X'0105' status is returned.

2. If the area specified to receive the data is not large enough to contain all the statistical data but is large enough to hold the sense data, the instruction returns only enough data to fill the specified area.

3. Repeated Sense instructions can be issued to obtain sense and statistical data for multiple devices without the application changing AMSNID because sense updates AMSNID with the next device ID on completion. If AMSNID is set to zeros initially, repeated execution of the sense instruction cycles through all device IDs. At the completion of the sense instruction, a value of '0' in AMSNID indicates presentation of data associated with the last device.

LCNTRL SSD (Stop Soliciting Data)

LCNTRL SSD stops data input from the ALA line specified in AMSNID. The instruction is not immediate. While input normally ceases after the next message is received, the maximum number of messages that may be received after the instruction is issued can be as great as the number of read buffers specified in CPGEN for that line.

> **Note:** Although no more messages are solicited from the SLU on the line, the control unit and terminals remain online.

The coding syntax for LCNTRL SSD is:

Name	Operation	Operand
`[label]`	`LCNTRL`	`SSD`

SSD

Issues a "stop soliciting data" request to the controller or device identified by AMSNID.

Condition Codes: One of the following is set:

Hex Code:	**Explanation:**
01	The instruction was executed successfully.
02	Status is returned; the status code is contained in SMSDST.

Program Checks: 9 might be set.

Possible Status: 0402 or 0403 might result.

LCNTRL STPL (Stop Line)

LCNTRL STPL deactivates an ALA line. STPL uses a 3-byte parameter list to specify the function to be performed and the characteristics to be associated with the line. The parameter list is in the following format:

```
 ---------------------------------------
|   FUNCTION    |         LINE          |
|  DEFINITION   |      DEFINITION       |
 ---------------------------------------
Byte    0               1          2
```

where:

FUNCTION DEFINITION:

X'00'	Stop all active lines using the existing line definition values. (Note: AMSNID is not used.)
X'10'	Stop the line designated by line definition byte 2. (Note: AMSNID is not used.)
X'20'	Stop all active lines specified in the parameter list. (Note: AMSNID is not used.)
X'40'	Stop the line indicated in AMSNID using the existing line definition values.
X'60'	Stop the line indicated in AMSNID.
X'80'	Stop all the lines using the existing line definition values. (Note: AMSNID is not used.)
X'A0'	Stop all the lines specified in the parameter list. (Note: AMSNID is not used.)

Note: Function bytes X'20', X'60', and X'A0' should not be used unless required by the ALA support in use.

Line Definition Byte 1 is not used by the ALA link.

Byte 2 (used only with Function definition X'10') designates the line number; for example, X'01' for line 1, X'02' for line 2, and so forth.

The coding syntax for LCNTRL STPL is:

```
Name     Operation  Operand

                          ⎧defld1              ⎫
                          ⎪seg1                ⎪
[label]  LCNTRL    STPL,  ⎨seg1,disp1,len1     ⎬
                          ⎪(reg1)              ⎪
                          ⎩(defrf1)            ⎭
```

STPL

Specifies a Stop Line request.

operand 1

Addresses the three-byte parameter list. If this address specifies a larger field, the first three bytes are considered the parameter list and all remaining data is ignored. Coding *seg1*, creates a two-byte machine instruction. Specifying any of the other addresses creates a machine instruction six bytes long.

Condition Codes: One of the following is set:

Hex Code:	**Explanation:**
01	The instruction was executed successfully.
02	Status is returned; the status code is contained in SMSDST.

Program Checks: 1, 2 or 9 might be set.

Possible Status: 0401 or 0402 might result.

LCNTRL STPL Programming Notes

1. When the AP issues this instruction, each associated line adapter is disabled (the line is stopped) immediately.
2. Any stations with devices in the Online state are posted with Loss of Contact status and any write operations in progress end with Loss of Contact status.
3. When using a multiple-line function byte, the first line that fails because of an invalid parameter list or the absence of an optional module causes the presentation of status. The remaining lines must then stop individually by reissuing STPL and specifying each line NID in AMSNID.

LCNTRL STRL (Start Line)

LCNTRL STRL activates ALA lines. STRL starts polling to any entities on the specified lines that are in an online pending state as a result of a VONL command or CPGEN specification of "active." LCNTRL STRL uses a 3-byte parameter list to specify the function to be performed and the characteristics to be associated with the line or lines. The parameter list is addressed by LCNTRL operands, with entries in the following format:

	FUNCTION DEFINITION	LINE DEFINITION	LINE NUMBER
BYTE	0	1	2

where:

FUNCTION DEFINITION:

X'01'	Perform Adapter Wrap test on the line specified in the AMSNID field.
X'02'	Perform Modem Wrap test on the line specified in the AMSNID field.
X'03'	Perform Adapter Wrap test and Modem Wrap test on the line specified in the AMSNID field.
X'04'	Begin SDLC Link Test. Line definition bytes 1 and 2 are ignored, and data beginning with byte 3 of the parameter list is transferred as test data to the SDLC station specified by AMSNID.
X'40'	Start the line indicated in AMSNID using the line definition byte specified in the CPGEN or by a previous LCNTRL STRL instruction.
X'50'	Start the line indicated in the line number byte of the parameter list using the line definition byte specified in the CPGEN or by a previous LCNTRL STRL instruction. (Note: AMSNID is not used.)
X'60'	Start the line indicated in AMSNID using the line definition byte supplied in the parameter list.
X'70'	Start the line indicated in the line number byte of the parameter list using the line definition byte supplied in the parameter list. (Note: AMSNID is not used.)
X'80'	Start all the lines using the existing line definition values established during the CPGEN specification or by a previous LCNTRL STRL instruction. (Note: AMSNID is not used.)
X'A0'	Start all the lines, using the line definition bytes specified in the parameter list. (Note: AMSNID is not used.)

For the SDLC Link Test (function byte equal to X'04'), line definition bytes 1 and 2 are not used. For all other LCNTRL STRL operations, bits 2, 5, and 6 of byte 1 are defined as follows:

Bytel, Bit:	***Function:***
2 = 0	NRZ encoding
2 = 1	NRZI encoding
5 = 0	Full-speed baud rate
5 = 1	Half-speed baud rate
6 = 0	Leased line with controlled request to send.
6 = 1	Leased line with permanent request to send.

Byte 2: See "LCNTRL STRL Programming Notes," below.

The coding syntax for LCNTRL STRL is:

```
Name      Operation  Operand

                           ( defld1             )
                           | seg1               |
[label] LCNTRL     STRL,  <  seg1,disp1,len1     >
                           | (reg1)             |
                           ( (defrf1)           )
```

STRL
Specifies a Start Line request.

operand 1
Addresses the three-byte parameter list. If this address specifies a larger field, the first three bytes are considered the parameter list and all remaining data is ignored. For an SDLC link test, the data in all bytes but 1 and 2 are considered test data. Coding *seg1*, creates a two-byte machine instruction. Specifying any of the other addresses creates a machine instruction six bytes long.

Condition Codes: One of the following is set:

Hex Code:	**Explanation:**
01	The instruction was executed successfully.
02	Status is returned; the status code is contained in SMSDST.

Program Checks: 1, 2 or 9 might be set.

Possible Status: 0104, 0203, 0204, 0401, 0402, 0403 or 0405 may result. Refer to Appendix B for status description and corrective action.

LCNTRL STRL Programming Notes

1. With some of the STRL functions, the AP must place the network identifier of the line in AMSNID before issuing the LCNTRL STRL instruction. The value placed in AMSNID is the network identifier assigned to the line at CPGEN time.

2. LCNTRL STRL must always be issued for an ALA line not CPGENed as active before data transfer can begin on the line. A Start Line function is automatically performed during system startup using the values specified in the CPGEN if a line is specified as active.

3. LCNTRL STRL must be issued for a line before ownership of any device on the line is established or a Vary Online is issued for any device on the line.

4. If the line fails to start, status indicating a Start Line failure is presented on the first Vary Online, LREAD, or LWRITE operation to an ALA device on the line. Because the station issuing the Start Line can either be the System Monitor or a user station, the asynchronous entry point is not activated for this failure. A specific action, as indicated, is required. The system log and statistics counters should be examined to determine the cause of the Start Line failure.

5. When using a multiple-line function byte, the first line that fails to start because of an invalid parameter list or the absence of an optional module causes the presentation of status. The remaining lines must then be started individually, specifying each line NID in AMSNID.

6. Byte 2 of the parameter list (used only with function definitions X'50' and X'70') designates the line number; for example, X'01' for line 1, X'02' for line 2, and so forth.

LCNTRL VONL (Vary Online)

LCNTRL VONL establishes a logical connection between an ALA device and the work station, starts polling, and sets the operating mode of an ALA control unit. The NID of the device to be varied online must be specified in AMSNID before you issue this instruction.

A work station can vary a terminal online. When a terminal is successfully varied online, the control unit is said to be in message routing mode. Messages received from the terminals on the control unit are associated in the 4700 with the work station owning the specific terminals.

The coding syntax for LCNTRL VONL is:

Name	Operation	Operand
`[label]`	`LCNTRL`	`VONL`

VONL
Issues "Vary Online" order to the ALA device or controller selected by the AMSNID field.

One of the following is set:

Hex Code:	**Explanation:**
01	The device indicated in AMSNID is currently in the Online state. (X'0800' status was previously reported.)
02	Status returned; the status code is contained in SMSDST. The status returned can indicate that the device is in the Online state (X'0800').

Program Checks: 9 might be set.

Possible Status: 0200, 0800, 0402, 0403, 0406, 0407, 4001, 4002, 4003, or 4004 might result.

LCNTRL VONL Programming Notes

1. A work station must own a device before a Vary Online (VONL) can be performed for that device.
2. When a terminal is successfully varied online, its associated control unit operates in message routing mode and cannot be varied online even though it may be owned. For a VONL to be successful in message routing mode, an Activate Physical Unit (Type 2) and an Activate Logical Unit must be sent in addition to an SNRM command, and the appropriate responses must be received.

3. If execution of VONL results in status of X'4001' (Online Pending) the device is in a state identical to that produced by specifying the initial status as active during the CPGEN process. Online status is reported on a subsequent ALA LREAD, LWRITE, or LCHECK. The application should exit and wait for the status to be presented at the ALA asynchronous entry point of the application program.

LCNTRL VOFF (Vary Offline)

LCNTRL VOFF removes the logical connection between the work station application program and an ALA device, stops polling, and resets the operating mode of the control unit if the last terminal on a control unit is varied offline. You must place the ID of the ALA device in AMSNID before you issue this instruction.

The coding syntax for LCNTRL VOFF is:

Condition Codes: One of the following is set:

Name	Operation	Operand
`[label]`	`LCNTRL`	`VOFF`

VOFF

Issues a "vary offline" request to the controller or device identified by AMSNID.

Hex Code:	**Explanation**:
01	The instruction was executed successfully.
02	Status is returned; the status code is contained in SMSDST.

Program Checks: 9 might be set.

Possible Status: 0402, 0403, or 0406 might result.

Notes on using LCNTRL VOFF:

1. The device must be owned by the work station issuing the instruction.
2. In message routing mode, a Deactivate Logical Unit is sent. If the SLU fails to return the required response, it is still placed in the offline state. If this is the last SLU to be varied offline, a Deactivate Logical Unit (PU Type 2 only) followed by an SDRM is transmitted. The PLU can force the entire PU offline by issuing a Vary Offline to the PU. In this case, an SDRM is sent to force the PU and all SLUs offline.
3. The first 12 bytes of messages received from terminals that are in the offline state, or of messages that are pending when the Vary Offline is executed, are logged and then discarded.

LREAD AL—Read Data from the Alternate (ALA) Link

LREAD AL passes data or status received in the 4700 controller input buffer from the SLU to the issuing station's application program. To read data from a specific ALA device, the device must be owned, not offline, and you must set the network ID of the device in AMSNID before you issue the LREAD.

If you set AMSNID to zero, the LREAD performs a general read. This passes data or status for any device owned by the issuing station to the application program.

Setting AMSNID to X'FFFF' causes LREAD AL to perform a "read search" for the AMSNID of the dispatching ALA device without reading the pending data or status. Whether the read search is for the first device found with pending data or status or for a specific device depends on the AMSARCF read flag's search indicator setting.

An LREAD AL with AMSNID of X'FFFF' and the search continue indicator off (stop search) returns the NID of the first device assigned to the station found to have data or status pending. Setting the search continue indicator on (search continue) with AMSNID equal to X'FFFF' causes the read search operation to restart with the NID following the one returned by any previous read search. By issuing successive read search LREAD AL instructions, you can query a particular device's outstanding data or status.

An LREAD AL instruction performing a read search operation, but that finds no pending status or data, sets AMSNID to zero and SMSDST status of X'4000'. An LREAD AL read search operation that finds a device with pending data or status returns the device's NID in AMSNID and zero status in SMSDST. To read the data or clear the pending status, you must set the device's NID in AMSNID and issue a normal LREAD AL.

An LREAD AL operation ends when one of the following conditions occurs:

1. All data is transferred.
2. Reading passes the end of the input field (if the FLI is nonzero and less than or equal to the length between the PFP and the end of the segment).
3. Reading passes the end of the segment (if the FLI is zero or greater than the length between the PFP and the end of the segment).

If no data or status is pending, LREAD AL returns a status of X'4000', indicating that no data is available. The read operation does not force the station to wait.

LREAD AL does not change the primary field pointer (PFP) but does store the data length in SMSIML, the network ID of the device that was read in AMSNID (if AMSNID was zero), the status in SMSDST, the message type and status in the read control field (AMSARC), and the sequence number and ID of the message read in AMSASQ. Refer to "Controller Fields and Indicators" in Chapter 2 for a description of AMSARC.

You should issue another LREAD AL after the first LREAD AL, or test SMSAAP to verify that no further status or input is pending.

If the status read is from a preceding LWRITE AL operation, the prior operation indication (Bit 3) in SMSDST is set and the write event number associated with the failing write operation is presented in AMSAFN.

When the station is idle and data or status is pending from an ALA device, the station is dispatched at the ALA= entry point.

The syntax for LREAD AL is:

```
Name     Operation  Operand

                      ⎧defld1             ⎫
                      ⎪seg1               ⎪
[label]  LREAD     AL,⎨seg1,disp1,len1    ⎬
                      ⎪(reg1)             ⎪
                      ⎩(defrf1)           ⎭
```

AL
Specifies that data is read from the ALA link.

operand 1
Defines the location into which data is read.

Condition Codes: One of the following is set:

Hex Code:	**Explanation:**
01	Data has been received successfully.
02	Status is returned; data may, or may not, have been transferred successfully; the status is contained in SMSDST.

Program Checks: 1, 2 or 9 might be issued.

Possible Status: Refer to Appendix D, "Link Status Codes."

LREAD CP—Read Data from the Host

This LREAD instruction reads data sent over the host link, and stores the data in the location specified by operand 2.

Reading ends when one of the following conditions occurs:

- The end of the message is reached.
- The end of the input field is reached (if the FLI is nonzero and less than, or equal to, the length between the PFP and the end of the segment).
- The end of the segment is passed (if the FLI is 0 or greater than the length between the PFP and the end of the segment).
- The operator signals attention.
- A loss-of-contact condition is encountered.

At the end of the operation, the PFP is unchanged. If the operation ends in any way other than the operator's signaling attention or loss-of-contact condition, the controller sets the following SMS fields:

- SMSIML is set to the binary record length.
- SMSCRC type (SMSCRT) and flags (SMSCRE/F) are set.
- The message sequence number in SMSCRS (for synchronous data).
- The response sequence number in SMSCSR (for a response only).
- Status in SMSDST.

The read types and control flags are described in Chapter 2, "SNA/SDLC Host Link Programming."

```
Name      Operation  Operand

                      ⎧ seg2        ⎫
                      ⎪ defld2      ⎪
[label]   LREAD    CP,⎨ seg2,disp2  ⎬
                      ⎪ (reg2)      ⎪
                      ⎩ (defrf2)    ⎭
```

CP

Specifies a read from the host system.

operand 2

Is the location to contain the data read. A length of zero causes data to be read into operand 2 beginning at the specified location and continuing to the end of the segment. Do not read data into Segment 14.

Condition Codes: When LREAD completes, the controller sets one of the following:

Hex Code:	**Explanation:**
01	LREAD executed successfully.
02	Status is returned in SMSDST. (See Appendix D, "Link Status Codes" for an explanation of the status codes.)

Program Checks: 1, 2, or 9 can be set.

Programming Note: LREAD forces the station to wait until all data is read and status is stored before allowing processing to continue. However, an LREAD that is issued between an LWRITE and an LCHECK completes independently of the LWRITE if the LREAD uses a different data area.

LWRITE AL—Write Data to the Alternate (ALA) Link

LWRITE AL sends data to an SLU device connected to the 4701 controller on the ALA link. In message routing mode, the write control field (AMSAWC) must be initialized (see "ALA/AMS Control Fields and Indicators" on page 3-34 for the AMSAWC values) and you must place the device Network ID (NID) in AMSNID before issuing the LWRITE AL command. After the LWRITE AL completes successfully, AMSASQ is set to the sequence number (normal flow) or the message ID (expedited flow).

The syntax for LWRITE is:

```
Name     Operation  Operand

                     ⎧ defld1           ⎫
                     ⎪ seg1             ⎪
[label]  LWRITE   AL,⎨ seg1,disp1,len1  ⎬
                     ⎪ (reg1)           ⎪
                     ⎩ (defrf1)         ⎭
```

AL
Specifies a write operation to an ALA controller or device.

operand 1
The location from which data is written. Specifying *seg1* generates a two-byte machine instruction. All other addressing forms create a machine instruction six bytes long.

Condition Codes: One of the following is set:

Hex Code:	Explanation:
01	The write operation was successfully initiated.
02	Status is returned; the status code is contained in SMSDST. Data has not been transferred.

Program Checks: 1, 2 or 9 might set.

Possible Status: See Appendix D, "Link Status Codes."

Programming Notes

1. If the LWRITE instruction returns a condition code of X'01', the Write Event Number (AMSAEN) is increased by 1. The previous number in AMSAEN is assigned to the write operation. Any error found during actual writing of this data will be reported on a subsequent LWRITE, LREAD, or LCHECK with the previous number being presented in AMSAFN. SMSIML is set to zero.

2. If the LWRITE instruction returns a condition code of X'02', the associated status code is stored in SMSDST, and AMSAEN is not increased. SMSIML is set to the amount of data not transmitted. If the status presented relates to a preceding LWRITE operation, the prior operation indicator (Bit 3) in SMSDST is set and AMSAFN contains the identifier associated with the failing LWRITE operation. If the prior operation bit is not set in conjunction with the condition code X'02', AMSAFN is set equal to AMSAEN.

3. The segment containing the write data should not be reused until the write operation completes successfully.

4. When the maximum number of outstanding writes (as defined in the CPGEN macros) is attained without an intervening LCHECK, an implied LCHECK occurs.

LWRITE CP—Write Data to the Host

This LWRITE instruction writes data to the host over the host link. The data is written from the location specified by operand 2.

Before LWRITE CP is issued, you must specify the data type (data, command, or response) in the SMSCWC write control type and flags fields (SMSCWT and SMSCWE/F). If the length of the data to be written is defined as 0, a data length of zero is transmitted.

After the LWRITE completes, processing continues with the next sequential instruction, and the following SMS fields are set:

- The write sequence number is stored in SMSCWS (for synchronous data flow only).
- The status code is stored in SMSDST.
- The message length is stored in SMSIML.
- If no program check occurs and no status other than unit exception is returned, SMSIML is set to 0. Otherwise, SMSIML is left at its initial value.

Refer to the write types and control flags descriptions in Chapter 2, "SNA/SDLC Host Link Programming."

Name	Operation	Operand
[label]	LWRITE	CP, { seg2 defld2 seg2,disp2,len2 (reg2) (defrf2) defcon2 }

CP

Specifies a write to the host system.

operand 2

Defines the data to be written. The length of the data to be written is from 0 to 4095 bytes. If *seg2* is specified, the SFP must point to the start of the field, and the PFP must point one byte past the end of the field.

Condition Codes: When LWRITE completes, the controller sets one of the following:

Hex Code:	Explanation:
01	The write operation was successful.
02	Status is returned; the status code is in SMSDST. (See Appendix D, "Link Status Codes" for an explanation of the status codes.)

Program Checks: 1, 2, 3, 9, 0F, 10, or 27 can be set.

Programming Notes:

1. During configuration, you must specify for either or both the controller and work station that *n* (1-7) LWRITE instructions can be issued to the host system before there is an automatic check of any previous write operations. When the *n+1* LWRITE instruction is issued, a check of the first LWRITE in the sequence occurs. (Similarly, the *n+2* LWRITE checks the second in the sequence, the *n+3* LWRITE checks the third, and so forth.)

2. Specifying *seg2* creates a 2-byte machine instruction; otherwise, the machine instruction is 6 bytes long.

STPLNK—Stop the SNA/SDLC Host Link

STPLNK immediately stops the host link. Operand 2 points to a 7-byte parameter list. This list may be the same parameter list used by STRLNK; however, STPLNK neither uses nor changes the parameters.

Name	Operation	Operand
[label]	STPLNK	defld2 seg2,disp2 (reg2) (defrf2) defcon2

operand 2

Defines the start of the parameter list. The length associated with the operand is ignored. The formats that the parameter list can have are shown in Figure 4-2 on page 4-46 . Any length specification is ignored, and the first 7 bytes are assumed to be the parameter list.

Condition Codes: The code is not changed.

Program Checks: 1, 2, 9, or 27 can be set.

STRLNK—Start the SNA/SDLC Host Link

STRLNK activates the communicating link to the host system if the link is stopped and performs a single external wrap test without starting the link. After the link is stopped, the program must issue STRLNK to restart the link. STRLNK points to a 7-byte parameter list that describes the link options, and can also define a selection sequence for an X.21 link (see Figure 4-2 on page 4-46).

Name	Operation	Operand
[label]	STRLNK	defld2 seg2,disp2 (reg2) (defrf2) defcon2

operand 2

Selects the parameter list. Any length associated with this operand is ignored. The parameter list for an X.21 link can vary from 4 to 32 bytes. If you specify 0 for operand 2, STRLNK uses the parameter list specified by the last STRLNK instruction. If none exists, the configuration parameters are used (refer to the COMLINK macro description in Volume 6 of this programming library).

Condition Codes: The code is not changed.

Program Checks: 1, 2, 9, or 27 can be set.

extension

Flag 1	Flag 2	CUA	Flag 3	XID	SSFL	Selection Sequence Field
Byte 0	1	2	3	4-6	7	Byte n

Flag 1 — a 1-byte binary number that describes the link options as follows:

Bit	*Explanation*
0	Reserved
1	Set to 1 to perform a single external wrap test.
2	Set to 1 for low speed.
3	Set to 1 for high speed.
4	Set to 1 for a wrappable modem.
5	Set to 1 for nonwrappable modem.
6	Set to 1 for non-NRZI (not for multiuse loops).
7	Set to 1 for NRZI (not for multiuse loops).

Flag 2 — a 1-byte binary number that describes the link options as follows:

Bit	*Explanation*
0	Set to 1 if CUA is present.*
1	Set to 1 if extension is present.*
2	Set to 1 for switched line.
3	Set to 1 for nonswitched line.
4	Set to 1 for connect data set to line.
5	Set to 1 for data set ready.
6	Set to 1 for permanent request to send.
7	Set to 1 for controlled request to send.

If bit 2 is 1, bit 6 is ignored.

CUA — a 1-byte binary address of the communication link.

Flag 3 — a 1-byte number that described the link options as follows:

Bit	*Explanation*
0	Set to 1 if XID is present.*
1	Set to 1 to perform wrap test**
2	Set to 1 to update selection sequence**
3	Set to 1 for X.21 autoanswer operation**
4	Set to 1 for X.21 autocall operation**
5	Set to 1 for X.21 direct call operation**
6-7	Reserved

If bits 2-5 are zero (0000), use the X.21 selection sequence specified by a previous STRLNK instruction; otherwise, use the selection sequence specified by the COMLINK macro.

*used by both X.21 and non-X.21 links
**used by X.21 links only.

Figure 4-2 (Part 1 of 2). Parameter List Used by STPLNK and STRLNK

XID — a 3-byte binary number assigned to this particular controller on the switched network.

The XID is represented in hexadecimal as a 20-bit binary number (3 bytes with the first 4 bits ignored). This ID must correspond with the ID specified for this controller during VTAM system definition with the IDNUM operand of the PU macro definition statement. The 4700 BLKID is X'57'.

The default value for XID is X'000000'.

SSFL — Length of the selection sequence field portion of the parameter list. This one-byte value, which determines the overall length of the parameter list, can be as large as 24 decimal.

Selection Sequence Field — up to 48 hexadecimal or 24 EBCDIC characters that control how the connection is made to the network. The 24-character limit includes data and delimiter characters.

The delimiter characters are:

- comma (,)
- minus (-)
- plus (+) — ending delimiter
- slash (/)
- point (.)

All decimal digits are valid data in the selection sequence field.

Figure 4-2 (Part 2 of 2). Parameter List Used by STPLNK and STRLNK

Chapter 5. 4700 SNA/SDLC Communication Macros

The 4700 communication macro instructions provide a way to communicate with the host system that ensures proper protocol. The two types of macro instructions, or "macros," are:

Definition macros: DEFLINK defines constants, work areas, tables, and registers; DEFASEP defines the entry point in your program where operation begins when the host link becomes active. DEFCODE generates the control macro subroutines that perform communication link message processing, including starting and ending a session. DEFCODE and DEFLINK also provide ways to branch and pass information to debugging routines that you supply. Note, however, that these facilities are for debugging only, and must be removed from the final version of the application program.

Control macros: OPENSESS begins a session; LSEND transmits a message to the host system; LRECEIVE sends a response to a message from the host system; CLOSSESS ends a session. CONFIRM sends a response (if required) to the last message received from the host. TMEXIT defines the communication exit from the controller application program.

Programming Considerations

When coding the controller application program that uses the SNA/SDLC communication macros, you must follow these conventions:

- Do not use the LREAD CP, LWRITE CP, LCHECK CP, or LEXIT assembler instructions in the same application program. If you do, the communication macros will not operate correctly.

- Your program must initialize the work area defined by DEFLINK to zero, but cannot use the work area during execution.

- Do not include any labels in your program that start with 'BUB' or '&BUB'.

- The 'COPY DEFSMS' and 'COPY DEFAPB' instructions must be issued in the application program.

- Set the user flags to zero before all macro calls.

- The communication macros should not use the linkage stack.

- Calling a macro destroys the PFP, SFP, and FLI for both the work segment and segment 1.

- Your program can use the work and linkage registers defined by DEFLINK, but calling a macro erases these registers.

- The message length field and the communication status field are correct after a macro call.

- Receive pacing and receiving BIND parameters from the host, as defined by the COMLINK macro's OPTION parameter, are not supported.

The contents of the input message length field (SMSIML) and the communication status byte (SMSCST) correctly reflect the result of communication with the host system when control is returned to the controller application program.

The Definition Macros

The definition macro instructions, DEFLINK and DEFASEP, must be coded once for each assembly of a controller application program. Figure 5-1 shows how you use these macros.

The DEFLINK Macro

The DEFLINK macro defines constants, fields, tables, work areas, and the equated values used by the controller application program. The DEFLINK macro must be the first communication macro issued, and must be within the first 4,000 bytes of the program.

```
         BEGIN          APBNM=TEST,DATA=01075,DEL=DELTAB,
                        ACP=CPU,PC=PC,ATD=TBD,STP=IPL
         COPY           DEFxxx         copy field definitions
           .
           .
           .
         DEFLINK        OUT=3,IN=4,WKSEG=(5,0),LKREG=14,WKREG=15
         DEFCODE        LOCE=LOST
           .
           .
           .
HOST     DEFASEP                       asynchronous entry point
         SETFPL         4,0,0          ready the input segment
         LRECEIVE       CANCEL=NO      read the message
         BRAN           CHKRET         check the condition code
           .
           .
           .
         TMEXIT
IPLRT    EQUATE         *              startup entry point
IPL        .
KBDRT    EQUATE         *              controller startup
KBD        .                           entry point
PCKRT    EQUATE         *              program check entry point
PC         .
           .
           .
LOST     EQUATE         *              loss of contact entry point
           .
           .
           .
         FINISH
         END
```

Figure 5-1. Contents of the Host Asynchronous Entry Point

The DEFLINK macro generates all fields needed by the application program. One of the fields that is built is the work area. Of particular interest to the controller application programmer are the first nine bytes of this area. The first 2 bytes are the user flags (BUBUSE). These flags are used to post events to the controller application program as follows:

If the condition code equals 1 after an LRECEIVE:

Label	**Hex Code**	**Description**
BUBUSEDA	8000	Normal data received
BUBUSEDC	4000	Data and control received
BUBUSERV	2000	Recoverable unit received
BUBUSEVS	1000	Host application program requires data
BUBUSEFC	0800	First-in-chain received
BUBUSELC	0400	Last-in-chain received
BUBUSEIL	0200	User buffer too small for message.

If the condition code equals 2 after an LRECEIVE:

Label	**Hex Code**	**Description**
BUBUSESE	8000	Session established
BUBUSESN	4000	Session ended
BUBUSESS	2000	Session suspended
BUBUSESR	1000	Session resumed
BUBUSECN	0800	Cancel received
BUBUSESH	0400	Shutdown requested
BUBUSENR	0200	Negative response received
BUBUSEMO	0100	Last message sent before failure of loss of contact to be received at the host
BUBUSEDL	0080	Data loss in previous session
BUBUSEFL	0040	Attempt to open session failed
BUBUSEAT	0020	LRECEIVE broken by the Attention key.

If the condition code equals 2 after an LSEND (RESP=DEF):

Label:	**Hex Code:**	**Description:**
BUBUSENR	0200	Negative response received.

Also, see the discussion on the LSEND and LRECEIVE macro instructions. The user flags can be tested in one of two ways:

```
(1)  TSTMSKI    BUBUSE,BUBUSESS                        SESSION SUSPENDED?
(2)  TSTMSKI    BUBUSE,AL2(BUBUSEFC+BUBUSELC)  FIC OR LIC READ?
```

The first example shows a single bit test, the second a multiple bit test. Following the user flags field is 3-byte field containing the address of the application program's next sequential instruction (BUBRTRN). This field is valid, and can be used as a return address if processing is interrupted after, for example, a program check or loss of contact. Because no instruction follows a TMEXIT macro, BUBRTRN contains the TMEXIT macro's address. If the program returns control to the TMEXIT macro after the first was interrupted, execution of an

LEXIT instruction is guaranteed. Following BUBRTRN is a 4-byte field, BUBSENSE, that holds the sense information received whenever the negative response flag (BUBUSENR) is on.

The control macros use the remaining fields defined by DEFLINK for various purposes.

The DEFASEP Macro

The DEFASEP macro defines the asynchronous communication entry point. When issued, DEFASEP sets a flag to indicate that the next LRECEIVE macro issued is a result of an asynchronous entry. The LRECEIVE macro must be executed before a TMEXIT macro is issued or an idle state (an LEXIT instruction issued) is entered (for example, a non-data message is received, which is processed totally within the communication macros' code).

After execution begins at the entry point and LRECEIVE is executed, control returns to the next sequential instruction (for example, data received). The DEFASEP macro must be the first statement at the CPU asynchronous entry point, and the 'ACP=' parameter of the BEGIN instruction must define the label of the DEFASEP macro.

The DEFCODE Macro

The DEFCODE macro instruction expands to create the subroutines necessary to process the control macros. There is one subroutine for each control macro. Each subroutine resets the user flags (BUBUSE) to zero and saves the linkage register.

When processing is complete, each subroutine except TMEXIT enters a common exit routine that tests the flags. If any flags are set, the correct condition code is also set, and the common exit routine passes control to the next sequential instruction after the macro call. If no user flags are set, a test is made to determine if the LRECEIVE subroutine is relinquishing control. If not, the condition code is set to 1 and control is given to the next sequential instruction, after the macro call. If the LRECEIVE subroutine is giving up control, the LRECEIVE is reissued or the TMEXIT subroutine is entered. The TMEXIT subroutine is entered if the asynchronous entry switch is set.

DEFCODE also creates some internal routines that the control macro subroutines use to perform LREADs and LWRITEs for message transmission and to process the various messages received, such as *Bid* or *Start Data Traffic*.

When coded in a program to be assembled with the SPLIT option, DEFCODE must be preceded by the SECTION INSTR instruction.

The Control Macros

The control macro instructions perform the communicating process for the 4700 system. They perform all communication between the host and the work station. The code that is generated by these macros is usually just one or two instructions. Basically, all they do is branch and link to the DEFCODE subroutine for the specified function. The control macros are OPENSESS, LSEND, LRECEIVE, CONFIRM, TMEXIT, and CLOSSESS.

The OPENSESS Macro

Either the host or controller must issue OPENSESS to establish a session with the host. The OPENSESS macro expands to an instruction sequence that saves the specified parameters in a work segment save area and then branches to the OPENSESS macro subroutine.

The subroutine determines whether the host or the controller application program is performing the request and if the request is valid, writes the initiate message, reads the response, and checks the ending status of each read and write operation. After OPENSESS completes, execution returns to the next sequential instruction in the program. An example of the programming to begin a session is shown in Figure 5-2 on page 5-6.

```
         BEGIN          APBNM=TEST,DATE=010175,DEL=DELTAB,
                        ACP=CPU,PC=PC,ATD=KBD,STP=IPL
         COPY           DEFxxx         copy field definitions
           .
           .
           .
         DEFLINK        OUT=3,IN=4,WKSEG=(5,0),LDREG=14,WKREG=15
         DEFCODE        LOCE=LOST
           .
           .
           .
CPU      DEFASEP                       asynchronous entry point
           .
           .
           .
IPLRT    EQUATE         *              startup entry point
IPL
           .
           .
           .
OPENHOST OPESESS        SESSID=MYTEST,TYPE=HOST,RESP=FME
         BRAN           MO,CHKRET      are you in communication
BIND     SETFPL         4,0,0          yes, prepare the in segment
         LRECEIVE       CANCEL=NO      and get the Bind message
         BRAN           CHKRET         check the statu returned
SDT      SETFPL         4,0,0          if everthing is all right,
         LRECEIVE       CANCEL=NO      get start data traffic and
         BRAN           CHKRET         session is established
         TSTMSKI        BUBPAR1,AL1(BUBFLG1) branch back if
         BRAN           MO,BACK        controller-initiated session
           .
           .
           .
KBDRT    EQUATE         *              operator startup entry
KBD
           .
           .
           .
OPENLU   OPENSESS       SESSID=MYTEST,TYPE=LU,RESP=RRN
         BRAN           MO,CHKRET      initiate accepted?
         BRANL          BIND           yes, get Bind message
BACK     *                  return point after session established
           .
           .
           .
PCKRT    EQUATE         *              program check routine
PC
           .
           .
           .
LOST     EQUATE         *              loss of contact routine
           .
           .
           .
CHKRET   EQUATE         *              check for proper status after
                                       return from macro call
           .
           .
           .
         FINISH
         END
```

Figure 5-2. Initiating a Session Using the 4700 Communication Macros

Note: OPENSESS does not support the OPTIONS=BIND specification on the COMLINK configuration macro.

If the host requests permission to start a session before the OPENSESS macro is issued, the request will be denied. Therefore, if TYPE-HOST is coded, it is recommended that the OPENSESS macro be issued in the routine at the startup entry point (STP parameter on the BEGIN instruction). Also, if a loss-of-contact condition occurs after a session is established, and the session was host-initiated, the host must reinitiate the session. The OPENSESS remains in effect until a CLOSSESS macro is issued or until the session failure flag is turned on (BUBUSEFL).

The LSEND Macro

The LSEND macro instruction transmits one data message to the host. The message must be placed in the output segment (OUT parameter of the DEFLINK macro) before LSEND is issued, the output segment's secondary field pointer (SFP) must point to the first character of the message, and the primary field pointer (PFP) must point to the byte after the message.

LSEND expands to two instructions: one to store the specified parameters in the save area and the other to branch to the subroutine for the LSEND macro. If the work station is not in communication (SMSCST) with the host or if "session established" was not previously reported to the controller application program, the LSEND subroutine enters the Common Exit Routine. The subroutine sets the write control field (SMSCWC) according to the parameters specified by LSEND and OPENSESS, and the current bracket state.

Figure 2-28 on page 2-55 and Figure 2-29 on page 2-56 show all the possible bit settings for building the write control field. After setting the write control field, the subroutine sets the "in bracket" state and updates the write sequence number in SMSCWS (this is because the communication macros have taken responsibility for the message, and to ensure that the number is updated before a possible loss-of-contact occurs.) Next, the LSEND routine writes the message to the host and issues a response, if required. When writing is completed, the routine resets the bracket state if an end bracket condition was set in the write control field. When the LSEND routine returns control to next sequential instruction through the common exit routine, it sets the condition code to 1 if no BUBUSE user flags are set. A condition code of 2 is possible only if a definite response is required. In this case, BUBUSENR may also contain sense information indicating that there was a negative response to the LSEND message.

Note: If RESP=DEF is specified, the communications macros consider the message to be a recoverable unit. Control does not return to the next instruction until a response is received. If a loss-of-contact condition occurs, a request to resend the message will be indicated when the session is reestablished.

The use of the BRCKT parameter is important only during a bracket protocol as defined for CICS and IMS. Misuse of this parameter will cause unpredictable results. For example, if the controller application program begins a transaction and a host reply ends the transaction, specify BRCKT=MARK parameter on the beginning LSEND macro. If you expect no host reply, specify BRCKT=NORM.

Note: LSEND does not support inbound pacing and change-direction protocols.

An example of the coding that may be used to perform message transfer using bracket protocol is shown in Figure 5-3.

```
         BEGIN       APBNM=TEST,DATE=010175,DEL=DELTAB,
                     ACP=CPU,PC=PC,ATD=KBD,STP=IPL
         COPY        DEFxxx         copy field definitions

         BEGIN       APBNM=TEST,DATA=010175,DEL=DELTAB,
                     ACP=CPU,PC=PC,ATD=KBD,STP=IPL
         COPY        DEFxxx         copy field definitions
           .
           .
           .
         DEFLINK     OUT=3,IN=4,WKSEG=(5,0),LKREG=14,WKREG=15
         DEFCODE     LOCE=LOST
           .
           .
           .
HOST     DEFASEP                    asynchronous entry point
           .
           .
           .
           .
           .
           .
           .
         SETSFP      3,0            prepare to send data
         MVFLD       3,DATAF        move data to output segment
FLMARK   LSEND       RESP=DEF,DATA=CTL,BRCKT=MARK  send the data
         BRANL       CHKRET         will check for first-in-chain
RDRESP   SETFPL      4,0,0          or last-in-chain and status
         LRECEIVE    CANCEL=YES     read the response
         BRANL       CHKRET         check for proper status
NMARK    SETSFP      3,0            prepare to continue sending
         MVFLD       3,DATANXT
         TSTMSK      3,2DELIM       test to see if more data to be
         BRAN        MZ,FLMARK      sent, branch if not to set LIC
         LSEND       RESP=DEF,DATA=CTL,BRCKT=NORM
         BRANL       CHKRET         check for proper status
         BRAN        RDRESP         continue processing
           .
           .
           .
CHKRET   EQUATE      *              check for proper status
           .
           .
           .
DELTAB   DEFDEL      (C' ',X'FE')            delimiter table
           .
           .
           .
         FINISH
         END
```

Figure 5-3. Message Transmission Using Bracket Protocol

The LRECEIVE Macro

The LRECEIVE macro reads one data message from the host or receives status concerning the session through the user flag (BUBUSE). If a messager is received, it will be placed in the input segment specified by the DEFLINK macro. Before issuing LRECEIVE, the program must set the primary field pointer (PFP) for the input segment to specify the location where the message is to be placed. You can also specify the message length, or it can be implied by the end of the buffer. If a message overruns the buffer, LRECEIVE branches to a subroutine.

The overrun subroutine contains two entry points. The one chosen depends on how you specified the CANCEL operand of the LRECEIVE macro. If the keyboard/display Reset key can end the host read, the program enters the message-processing routine normally. If the Reset key cannot end the read operation, or the program executed LRECEIVE by beginning execution at the CPU asynchronous point, the routine sets a prevention flag and tests for a previous message that might have been delayed.

Delayed message processing is necessary because a host read operation can be initiated by either the LSEND or the LRECEIVE macro subroutine. The controller program can process any message read by an LRECEIVE macro, but not any message read by LSEND. Normally, LSEND reads a response from the host. But if LSEND reads a message that also requires a response, processing is delayed. This is accomplished by setting a flag indicating the type of message received. When the program later issues LRECEIVE, it reads the delayed message and performs *delayed message processing*.

To perform delayed message processing, the program must provide a special entry into the message processing subroutine to test for delayed messages and any user status returned by the host. If neither occurred, the program can then process the message normally.

After processing the message, the program enters a common exit routine, and processing continues. When control returns to the next sequential instruction, LRECEIVE sets the condition code to 1 or 2, depending on the status received. A condition code of 1 indicates a message was read; 2 indicates no message was read. The following flags may also be set if LRECEIVE sets a condition code of 1:

BUBUSEIL

The user's buffer cannot hold the complete message. If set, either BUBUSEDA or BUBUSEDC are also set.

BUBUSERU

The message received is a recoverable unit. A definite response is required; the application program must issue a CONFIRM, TMEXIT, or another LRECEIVE macro (RECEIVE and TMEXIT cause a positive response to be sent). If set, either BUBUSEDA or BUBUSEDC are also set.

BUBUSEFC/BUBUSELC

Either, but not both, can be set. BUBUSEFC indicates that this message is the first-in-chain; BUBUSELC indicates that this message is the last-in-chain. If the message is only on middle-in-chain, no indication is set. These flags are always set along with either BUBUSEDA or BUBUSEDC.

BUBUSEVS

The host program requested data. This flag is set if CICS uses the Converse function. This flag is always set with either BUBUSEDA or BUBUSEDC.

BUBUSEDA/BUBUSEDC

Either one, but not both, must be set. BUBUSEDA indicates a normal data message was received; BUBUSEDC indicates that control information is included in the text of the message. The sequence number received on this message can be found in SMSCRS. BUBUSEDC will be set for all CICS and IMS data transmissions. Refer to the CICS and IMS manuals for the formats of the control information.

A condition code of 2 is set if no message was read, but status concerning the session is indicated in the user flags. Only one of the following combinations is set in the user flag if the condition code equals 2:

BUBUSESE

The session requested by the OPENSESS macro has been established, and data can now be transferred. Either the BUBUSEMO or BUBUSEDL flag can also be set. The BUBUSEMO flag indicates that the last recoverable message sent in the previous session was lost. The message can be retransmitted if desired. The BUBUSEDL flag indicates the possibility that a message transmitted from the station was lost and/or a message to the station will be duplicated. Recovery actions are left to the controller application program.

BUBUSESN

The session is ended as a result of the CLOSSESS macro being issued. The controller application program may now issue another OPENSESS macro.

BUBUSESS

The session has been suspended. Data transmission cannot be resumed until an LRECEIVE is completed with BUBUSESR set in the user flags.

BUBUSESR

The session has been resumed. Data transmission should continue where it left off.

BUBUSECH

A Cancel message was received. All elements of the chain received should be discarded. The next message received will be first-in-chain or only-in-chain.

BUBUSESH

The host application program requested the end of the session. A CLOSSESS should be issued as soon as possible.

BUBUSENR

A negative response was received for a message previously sent requesting exception response (... LSEND ...RESP=EX...). The response sequence number field (SMSCSR) contains the sequence number of the message associated with the negative response.

BUBUSEFL

The attempt to open a session (OPENSESS) failed. Either the application program was not started or, unknown to VTAM and the session, was initiated by the controller, or the name of the host application program requesting a session did not match the name on the OPENSESS macro.

BUBUSEAT

The receive macro was broken by the attention key on the 4704 or 3604. This flag can be set only if the 'CANCEL=YES' parameter is coded on the LRECEIVE macro, and the LRECEIVE macro is issued for other than an asynchronous entry.

The CONFIRM Macro

The CONFIRM macro instruction sends a positive or a negative response to the last message received from the host. This macro expands to one instruction; this instruction branches to the subroutine for the CONFIRM macro.

There are two entries to this subroutine, depending on whether a positive or negative response is to be written. The correct response value is placed in the write control field, and the host Write routine is entered. The CONFIRM macro becomes a "no-operation" if any of the following is true:

1. No response was required for the received message.
2. The response was already sent.
3. The station is not communicating with the host.
4. 'RESP=OK' is coded and the BUBUSERU flag was not set when the message was received.

Upon return from the host write subroutine, the common exit routine is entered, and processing continues.

Note: If the BUBUSERU flag is set after issuing LRECEIVE, a response is required. If the BUBUSERU flag was not set, the message was received in either exception or no response mode. Because CONFIRM is processed as no-operation if a particular response is not required, the controller application programmer should always assume exception response mode if the BUBUSERU flag is not set.

The TMEXIT Macro

The TMEXIT macro instruction is the only valid exit macro for use with the communication macros; if you use LEXIT, the communication macros may not operate correctly.

The TMEXIT macro enters the idle state or branches to the CPU asynchronous entry point. This macro expands to one instruction that branches to the TMEXIT subroutine.

The subroutine first tests the delayed processing flags to see if such processing is needed. If delayed processing is needed, the TMEXIT routine enters the program at the CPU asynchronous entry point where LRECEIVE processes the delayed messages.

If no delayed processing flags are set, the TMEXIT routine determines if the Ready to Receive (RTR) message should be sent. This message is needed if the station issued a negative response to a previous Bid or message containing Begin Bracket. If RTR is required, but the station is not in bracket state, the station writes RTR and reads the response. A negative response to the RTR message (indicating that the data is no longer available) causes the station to issue LEXIT. A positive response again dispatches the station at the CPU entry point, allowing the first-in-bracket flow from the host to the station. If RTR is not required, the station issues LEXIT.

The CLOSSESS Macro

The CLOSSESS macro enables the controller application program to end a session. The session may be ended by the controller or the host. If the host ends the session without this macro being issued, control is passed to the loss-of-contact routine (specified in the DEFCODE macro). The loss-of-contact routine should no longer issue a CLOSSESS macro; the controller attempts to reestablish the session.

If a CLOSSESS macro is issued before a session is established, the session will be ended, regardless of the parameter specified. If the CLOSSESS macro is issued before an OPENSESS macro is issued, or if a second CLOSSESS is issued before the first completes, the result is a no-operation. This macro expands into two instructions: one saves the parameter; the other branches to the subroutine for the CLOSSESS macro.

The CLOSSESS subroutine performs several actions, depending on the parameter list and the current session state:

1. If a session is not established (SMSCST), the CLOSSESS executor functions independently of any parameters specified on the CLOSSESS macro. If an Initiate message was sent, CLOSSESS generates a negative response when the Bind message is received. If an Initiate message was not sent, a delayed processing flag is set, including a session failure or end. Control is then passed to the Common Exit routine.

2. If in session and TERM is specified on the TYPE parameter, a Terminal message is issued, the response is read, and the common exit routine is entered.

3. If in session and if a Shutdown message was previously received, independently of whether LU or HOST was specified on the TYPE parameter, a Shutdown Complete message is written, the response is read, and the common exit routine is entered.

4. If in session and if a Shutdown message was not previously received, a Request Shutdown message is written and the response is read if LU was specified on the TYPE parameter. If HOST was specified, the common exit routine is entered to await a Shutdown message.

The session is ended when the BUBUSESN flag is set on return from an LRECEIVE instruction.

Program Checks

A program check may occur during execution of the controller application program. To identify errors caused by the communication macros, traps are included in the code expanded by the DEFCODE macro. They can be identified by a program check code of 9 (invalid operation code) at a displacement of 6 into segment 1 (SMSPCC).

When a program check occurs, the program check subroutine ("PC=" option of the BEGIN instruction) is entered, if 1 had been coded. This subroutine should first check BUBRTRN. If it equals zero, the error did not occur within the code generated by the communication macros. If BUBRTRN is non-zero, it contains the return address of the invalid code within the communication macros.

To reestablish the session after a program check, a CLOSSESS TYPE=TERM should be coded to clear all delayed messages. The session may now be reinitiated via an OPENSESS. If the OPENSESS is issued prior to the CLOSSESS, it will be treated as a no-operation (because the session-established flag is still on) and a second program check may occur.

The following program check codes identify errors in the communication macros' programming:

FFFF	An impossible condition was encountered. This error occurs when processing delayed messages.
FFF0	Unexpected status was found after an LWRITE instruction was issued. The device status is stored in BUBUSE. Figure 2-28 on page 2-55 shows all possible device status settings.
FFF1	Unexpected status was found after an LREAD instruction was issued. The device status is stored in BUBUSE.
FFF2	A data check was reported on an LREAD instruction. This is a network error.

There are also traps that may be encountered in the code to identify invalid protocol usage:

FFF3	An unexpected response was received on a non-data command. The error is in the host (VTAM) application code. This may occur when a message expects a positive response and the host responds otherwise.
FFF4	An unexpected message was received. The VTAM application program protocol is not compatible with the communication macros.

FFF5	An unexpected form of the set and test sequence number (STSN) was received, or STSN was received at an invalid time.
FFF6	Two negative responses were received for a message, and RESP=DEF was not coded on the LSEND macro.

A program check may also occur if the primary and secondary field pointers are not set correctly prior to entry of the communication macros' code. The program check code will be 2 or 3. When the code is entered, BUBRTRN will contain the address of the LREAD and LWRITE instruction that program checked in the communication macros' code. The program check codes are:

300x	The field pointers were set incorrectly for an LRECEIVE macro (x is the input segment number).
310x	The field pointers were set incorrectly for an LSEND macro (x is the output segment number).

Message Flow Using the Communication Macros

The following examples show some of the ways communications can be achieved by using the communication macros. By making a comparison with the previous communicating sequences, you can see that the same steps are taken. The obvious advantage in using these macros, as opposed to the 4700 assembler instructions, is greatly reduced coding. This is accomplished by the internal code that handles and checks for the proper sequences, responses, and protocols that are used within the controller application program.

Initiating a Session (Figure 5-4 on page 5-15): A session may be initiated by the host application program or by the controller application program. For a session to be established using the communication macros, an OPENSESS macro must be issued. When the OPENSESS macro is issued, a test is made to see if a session had already been established. If it has, entry is made into the Common Exit routine. Next, OPENSESS decides who is going to start the session — the host or the controller. On host initiation, return is made through the common exit routine, and processing is similar to that shown in Figure 2-13 on page 2-39. If the session is to be initiated by the controller application program, an initiate message is written to the host. If the communications status field (SMSCST) indicates that a command is in progress, a positive response is read; otherwise, the Ready Message was not received and the common exit routine is entered to await a Ready Message.

After a positive response, the host issues an OPNDST macro, sending a Bind message to the controller. After receiving the Start Data Traffic (SDT) message, the communication macros set the BUBUSESN flag (session established), completing session initiation.

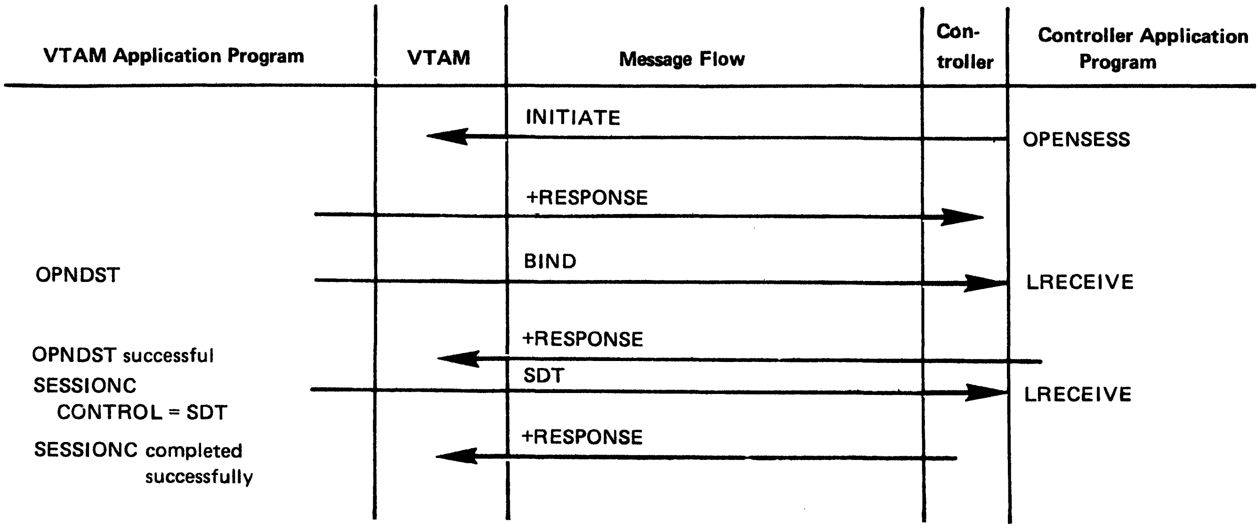


Figure 5-4. Session Initiation by the Communication Macros

Temporary Suspension of a Session (Figure 5-5): This figure shows how a session can be suspended using the Quiesce at End-of-Chain message. Upon receiving the Release Quiesce message, the session is reestablished, and data flow can begin again.

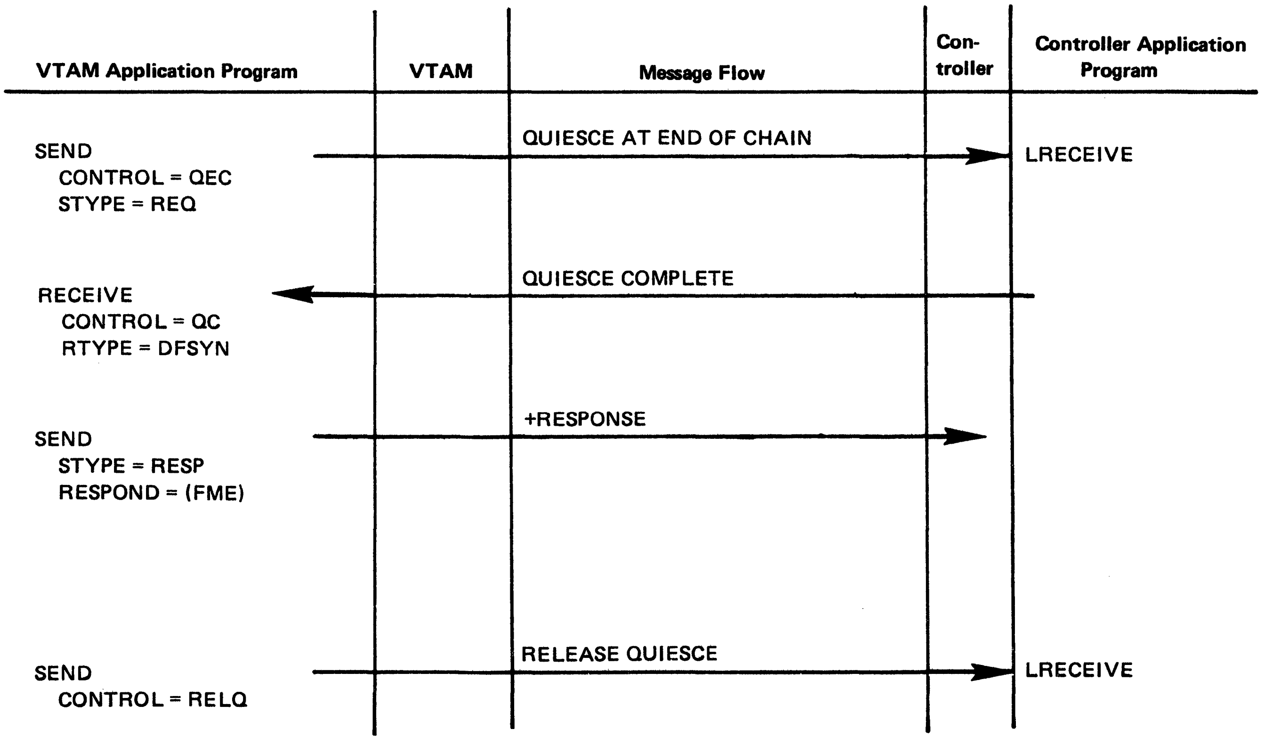


Figure 5-5. Temporary Suspension of a Session

Bidding for Control (Figure 5-6 on page 5-16): This figure shows how the host can gain control of the work station by Bidding. Control of the data flow is achieved by using bracket protocol. If the work station is already in the bracket state, the delayed processing flag is set to handle the Bid message as soon as the end bracket is recognized and the work station has relinquished control. If the work station is not in the bracket state, asynchronous entry is made, and the bracket state is entered.

Using the Converse Function (Figure 5-7 on page 5-17): This diagram shows how the change direction protocol can be used. This protocol can be started by the host or the controller application program. The number of messages transmitted between the work station and the host depends on the application program.

Immediate Termination of a Session (Figure 5-8 on page 5-18): This diagram shows session termination. As was discussed earlier, this type of termination should be used when a program check occurs during a session. The session ends upon receiving the unbind message, and the BUBUSESN flag is set. All messages in the network are lost.

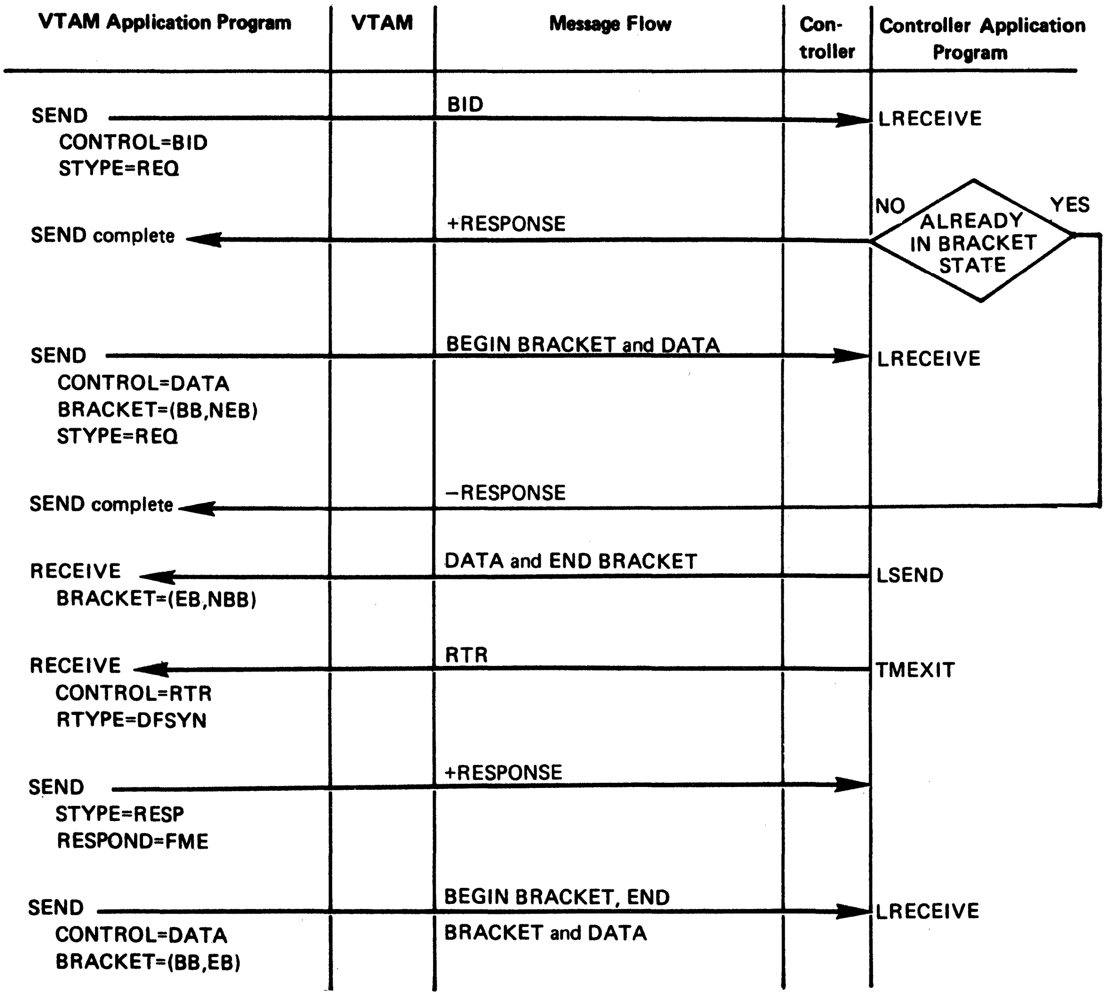


Figure 5-6. Bidding for Control

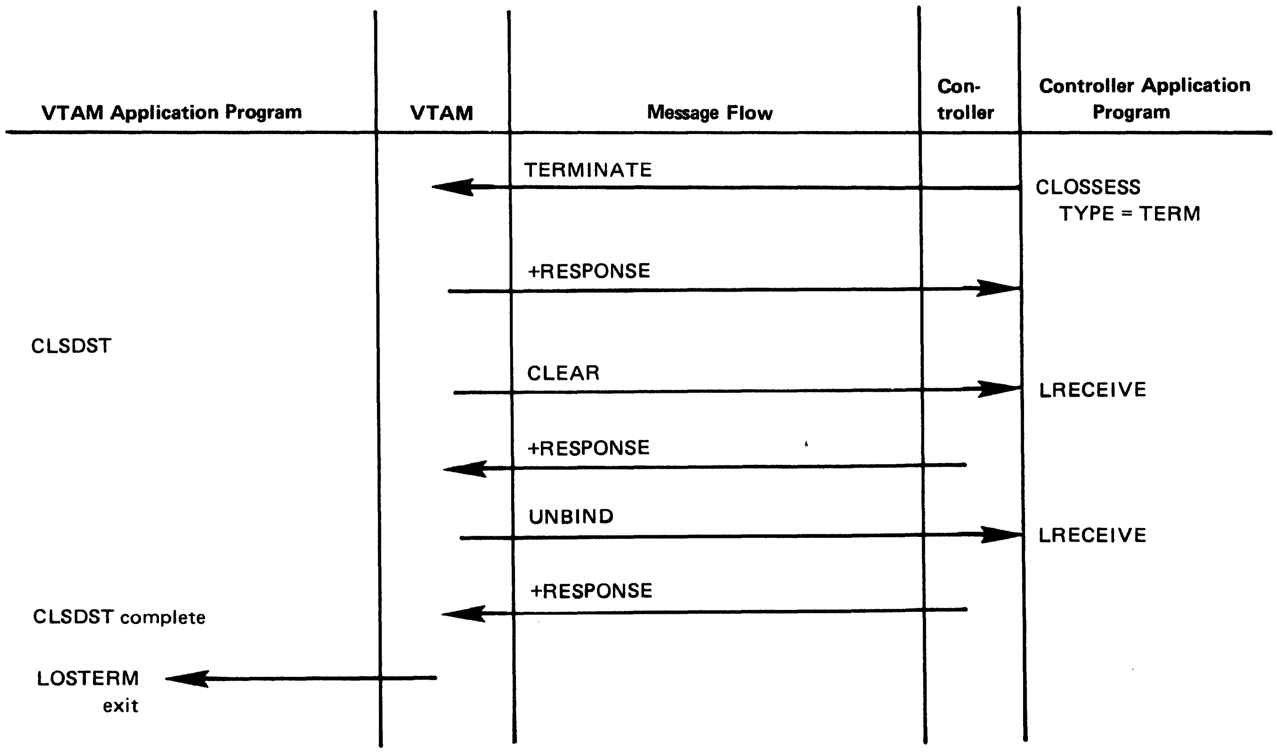


Figure 5-7. Using the Converse Function

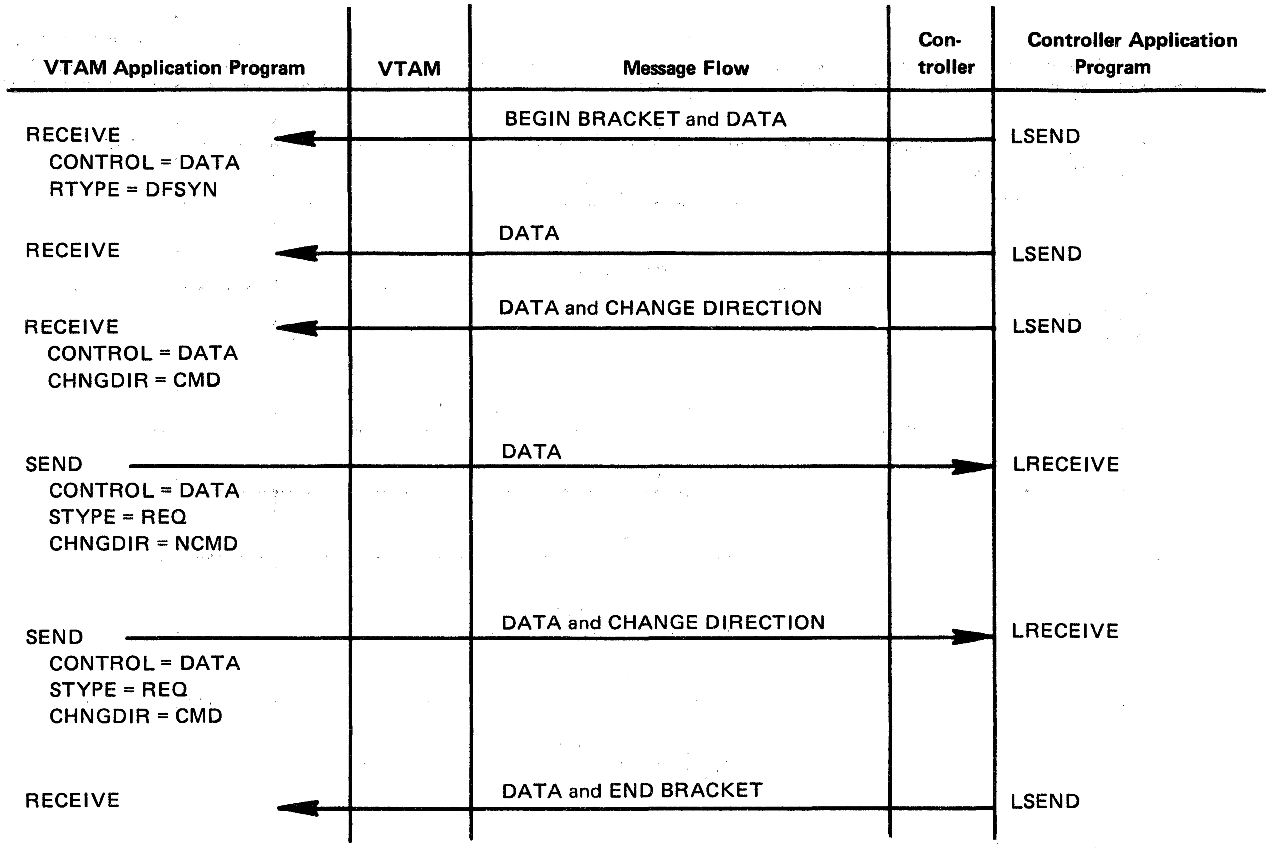


Figure 5-8. Immediate Termination of a Work Station

Orderly Termination of a Session (Figure 5-9 on page 5-19): Orderly termination of a session is achieved when the work station no longer has to communicate with the host. The controller application program must then issue a Request Shutdown message by coding the CLOSSESS macro. Orderly shutdown is achieved when the session-ended flag (BUBUSESN) is set in the Unbind routine. All messages within the network are processed before this flag is set so that no messages are lost.

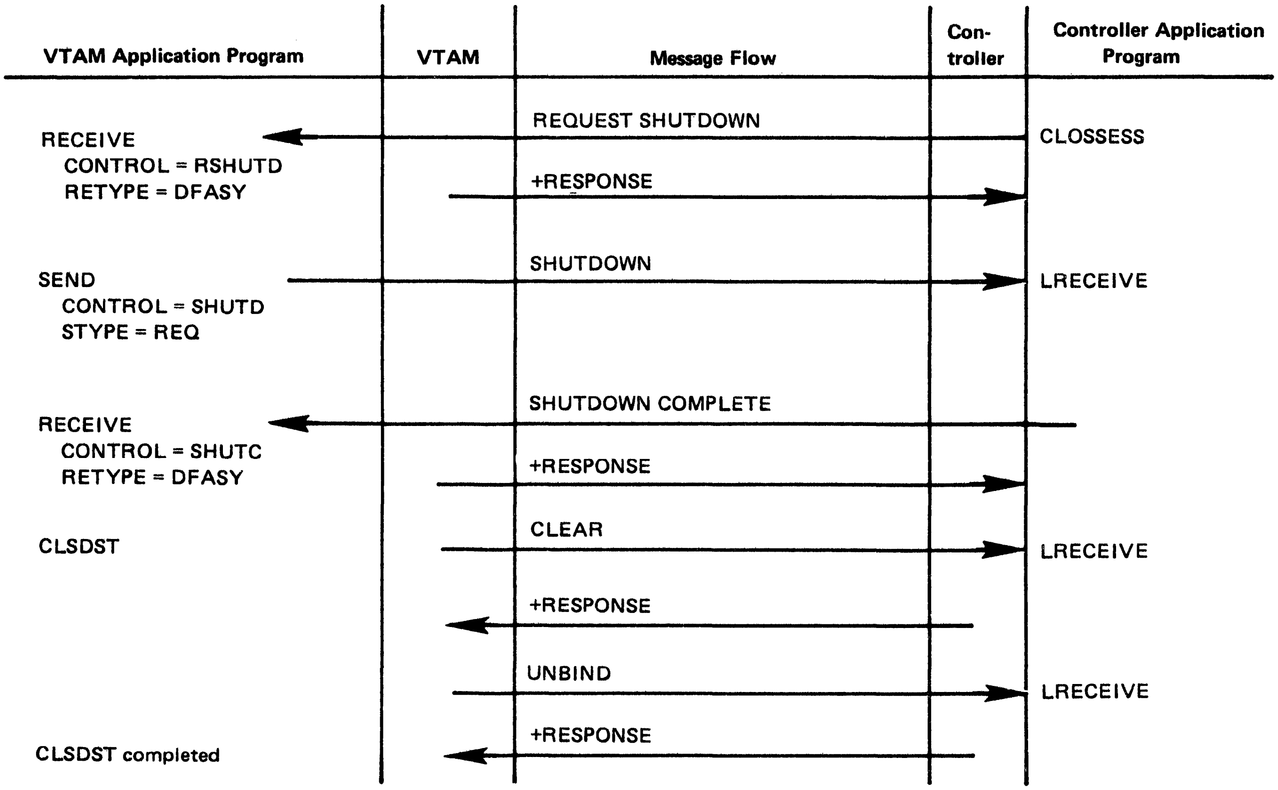
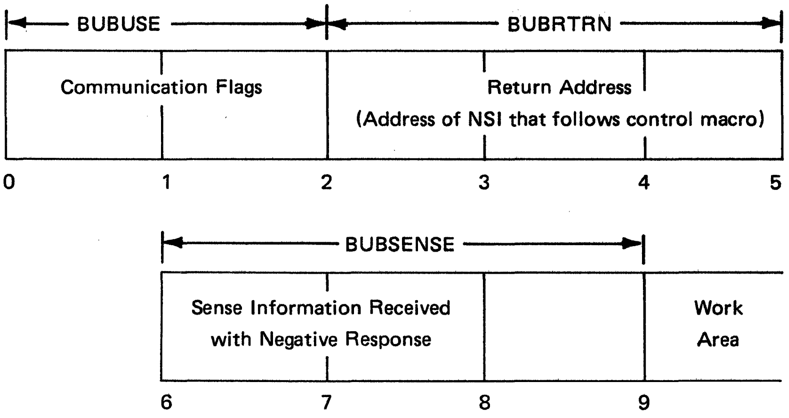


Figure 5-9. Orderly Termination of a Session

Flags and Sense Fields

The first 9 bytes of the work area defined by the DEFLINK instruction contain fields that reflect the result of communication with the host processor. These fields have the following format (the labels shown are defined by the DEFLINK instruction and may be used by the controller application program):



BUBUSE contains flags indicating the result of executing an LRECEIVE or LSEND. The following table shows the flags set, the label of the EQUATE instruction generated by DEFLINK, and an explanation of the flags:

Flag (hex)	Label	Explanation	
8000	BUBUSEDA	Data message received	
4000	BUBUSEDC	Data and Control message received	
2000	BUBUSERU	Recoverable unit received	Condition Code set to 1 after LRECEIVE
1000	BUBUSEVS	VTAM application program requires data	
0800	BUBUSEFC	First-in-chain received	
0400	BUBUSELC	Last-in-chain received	
0200	BUBUSEIL	Input segment too small for message	
8000	BUBUSESE	Session established (The session is not established until this bit is set.)	
4000	BUBUSESN	Session terminated (The session is not terminated until this bit is set.)	
2000	BUBUSESS	Data transfer suspended	
1000	BUBUSESR	Data transfer resumed	
0800	BUBUSECN	Cancel received	Condition Code set to 2 after LRECEIVE
0400	BUBUSESH	Shutdown received	
0200	BUBUSENR	Negative response received	
0100	BUBUSEMO	Last message sent prior to loss of contact was not received by VTAM application program	

Flag (hex)	Label	Explanation	
0080	BUBUSEDL	Data loss in previous session	
0040	BUBUSEFL	Session initiation failed	
0020	BUBUSEAT	LRECEIVE broken by keyboard attention	
0200	BUBUSENR	Negative response received	Condition Code set to 2 after LSEND

If it is necessary for the user to determine if a test negative condition was returned in response to a 'Set and Test Sequence Number', the user may test the one-byte field labeled BUBUTNEG in the work segment after the session has been established. A value of X'70' in BUBUTNEG indicates that a test negative condition occurred. At all other times, BUBUTNEG is set to X'50'; test positive.

Macro Descriptions

The following descriptions define the functions of the communication macros, how they are coded, their operand descriptions and options, the expansion storage that each uses, and the program checks ("traps") that each macro may set.

Each description begins on a new page for ease of reference. The coding syntax notation is the same as for other 4700 assembler instructions, and is described in Volume 1 of the *4700 Controller Programming Library.* Descriptions of the macro program traps are in the section following the macro descriptions.

CLOSSESS—End a Session

CLOSSESS enables the communication code to end (terminate) a session. The termination request can be made by either the work station or VTAM application program. Session termination is completed when the BUBUSESN flag is set in BUBUSE.

If the VTAM application program terminates a session before CLOSSESS is issued, control is passed to the loss-of-contact routine in the controller application program (as specified on the LOCE parameter of the DEFCODE macro). If CLOSSESS is issued before the BUBUSESE flag is set during session initiation, the session is ended. If CLOSSESS is issued before OPENSESS or when the work station is not in session, no operation is performed.

```
Name    Operation  Operand

                        ( TERM       )
[label] CLOSSESS TYPE= { LU          }
                        ( HOST       )
```

TERM

The communication code terminates the session and causes the VTAM application program's lost-terminal exit to be activated (the loss of contact routine in the controller application program will not be activated). This parameter should be coded if CLOSSESS is issued because of a program check.

LU

The communication code sends Request Shutdown to begin session termination.

HOST

The VTAM application program sends Shutdown to start session termination.

Program Storage Used: 10 bytes.

Program Check: Program check 9 may be set. Refer to "Communication Macro Instruction Traps" on page 5-41 for an explanation of the program check.

CONFIRM—Respond to a Host Message

CONFIRM sends either a positive or a negative response to the last message received from the host system. No operation is performed if any of the following is true:

1. No response was required.
2. A response has already been sent.
3. RESP=OK is coded, but the BUBUSERU flag was not set when the message was received.
4. The station has not established communication.

```
Name     Operation  Operand

                          { ERR  }
[label]  CONFIRM   RESP=  {      }
                          { OK   }
```

ERR

A negative response is sent. Before CONFIRM is issued, 4 bytes of sense information must be placed in the output segment (specified by the OUT parameter of DEFLINK), the SFP is set to the first byte of the sense information, and the PFP is set one byte past the end of the sense information.

OK

A positive response is sent.

Program Storage Used: 4 bytes

Program Checks: Program check 9 may be set. Refer to "Communication Macro Instruction Traps" on page 5-41 for an explanation of the program check.

DEFASEP—Define Communication Entry Point

DEFASEP must be the first statement at the ACP (host) entry point specified in the BEGIN statement; it may be coded only once for each application program.

Name	Operation
`[label]`	`DEFASEP`

label

The label of the DEFASEP macro. This label must be used as the label specified in the ACP operand of the BEGIN instruction.

Program Storage Used: 8 bytes

DEFCODE—Generate 4700 Communication Code

DEFCODE generates the 4700 assembler instructions used to communicate with the host system; it may be coded only once for each controller application program. In a split controller application program, DEFCODE must be preceded by SECTION INSTR.

Name	Operation	Operand
[label]	DEFCODE	LOCE=label[,LOCD=label] $\left[\text{,SEFPASMB=}\begin{Bmatrix}\text{Y}\\\text{N}\end{Bmatrix}\right]$

LOCE

The label of the first instruction in the routine for processing loss of contact with the host system.

LOCD

The label of the first instruction in the routine to which control will be passed after each execution of an LREAD, LWRITE, and LCHECK instruction in the communication macro code. This is for debugging purposes only, and should be coded after the application program runs successfully. When control passes to the debugging routine, DEFCODE stores the address of the next sequential location in the trace register defined by the DEFLINK macro. This location holds a two-byte code indicating the type of input/output operation being traced. The codes are:

W1 = Write negative response — bracket denied

W2 = Write negative response — permission rejected

W3 = Write negative response — logical unit reserved

W4 = Write positive response to Set and Test Sequence Number

W5 = Write data from user segment

W6 = Write data from work segment

C1 = LCHECK

R1 = Read data into user segment

R2 = Read data into work segment

To return from the debugging routine, you must branch to the address that is two bytes beyond the address in the trace register.

Note: If this operand is used, the user application program must save the status and condition code fields and restore them before returning control to the next sequential instruction in the communication macro.

SEPASMB

Specifies whether DEFLINK is assembled in a separate assembly from DEFCODE (Y) or is assembled with DEFCODE. If SEPASM=Y is specified, then DEFCODE must not appear in the same assembly with DEFLINK.

Controller Application Program Storage Used: 2048 bytes.

DEFLINK—Define Constants and Work Areas

DEFLINK defines the equated values, constants, tables, work areas, registers, and segments used by the communication macros. DEFLINK must be the first communication macro assembled in the application program, and must begin within the first 4000 bytes. Any symbols used in the operands must be defined prior to the macro.

```
Name     Operation  Operand

                                         { (seg, disp) }
[label]  DEFLINK    OUT-seg,IN=seg,WKSEG= {             },
                                         { seg         }

                    ,LKREG=reg

                    WKREG=reg [,DREG=reg] [,SEN1=xxxxyyyy]

                    [,SEN2=xxxxyyyy] [,SEN3=xxxxyyyy]

                    [          { N }]
                    [,SEPASMB= {   }]
                    [          { Y }]
```

OUT

A decimal number or symbol indicating the segment containing the data to be sent to the host system. This segment can be the same as the input segment.

IN

A decimal number or symbol indicating the segment where data will be placed when data is being received from the host system. This segment can be the same as the output segment.

WKSEG

seg

A decimal number (2-12) or symbol indicating the number of the segment used for the 78-byte work area. If you do not specify the displacement, 0 is assumed. This segment cannot be the same as the OUT or IN segment.

disp

A decimal number (0-178) or symbol indicating the displacement into the work segment of the start of the work area.

LKREG

A decimal number (1-15) or symbol indicating the register used by the communication macros as a linkage register. This register cannot be the same as WKREG or DREG.

WKREG

A decimal number (1-15) or symbol indicating the register used by the communication macros as a work register. This register cannot be the same as LKREG or DREG.

DREG

A decimal number (1-15) or symbol indicating the register used by the communication macros as the trace register. This register cannot be the same as LKREG or WKREG. Code this parameter only if you are using the DEFCODE debugging facility.

SEN1

Identifies the sense bytes for 'bracket denied' sense* returned with a negative response sent by the communication macros; where *xxxx* are the Systems Network Architecture sense bytes, and *yyyy* are the the user sense bytes. If not specified, the default is 08140000.

SEN2

Identifies the sense bytes for 'logical unit reserved' sense* returned with a negative response sent by the communication macros; where, *xxxx* are the Systems Network Architecture (SNA) sense bytes, and *yyyy* are the user sense bytes. If not specified, the default is 08080000.

SEN3

Identifies the sense bytes for 'permission rejected' sense status. This status is returned when the communication macros send a negative response. *xxxx* are the Systems Network Architecture sense bytes, and *yyyy* are the user sense bytes. If not specified, the default is 080A0000.

The sense bytes specified should be compatible with the host system and with Systems Network Architecture. The sense information is specified in hexadecimal numerals. The user should not put the hexadecimal notation (X) to identify it as hexadecimal.

SEPASMB

Specifies whether DEFCODE is assembled in a separate assembly from DEFLINK (Y) or is assembled with DEFLINK. If you specified SEPASMB=Y, then you must include DEFLINK in a DUMMY section with the DEFCODE assembly.

Controller Application Program Storage Used: 260 bytes.

LRECEIVE—Receive a Message from the Host

LRECEIVE is used to receive all messages sent from the host system. If the message is data or data-and-control, the data is placed in the segment specified by the IN parameter of the DEFLINK instruction, starting at the PFP, for a length equal to the message length, the amount of space remaining in the segment, or the value of the FLI (whichever is shorter). If the message is other than data or data-and-control, any information concerning session status is returned in the flags at BUBUSE.

Name	Operation	Operand
[label]	LRECEIVE	[CANCEL= { NO YES }]

NO

The LRECEIVE cannot be terminated by pressing a 4704 or 3604 attention key. This mode of operation is enforced if the LRECEIVE is issued after the work station is dispatched at the host system entry point.

YES

The LRECEIVE can be terminated by pressing the keyboard/display Attention key.

Controller Application Program Storage Used: 4 bytes.

Condition Codes: One of the following is set when control is returned to the instruction following LRECEIVE:

Hex Code:	Explanation:
01	Data or data-and-control was read: BUBUSE flags may be set.
02	A message other than data or data-and-control was read: BUBUSE flags may be set.

Program Checks: Program check 9 can be set. Refer to "Communication Macro Instruction Traps" on page 5-41 for an explanation of the program check.

LSEND—Send a Message to the Host

LSEND sends a data or data-and-control message to the host processor. The message must be placed in the output segment (specified in the DEFLINK macro), the SFP set to point to the first byte of the message, and the PFP set to point one byte beyond the end of the message before LSEND is executed. When LSEND completes, the send operation is verified, using an LCHECK instruction, to ensure that the message was received by the 3704 or 3705 without error and that the work station buffer may be reused. The message sequence number is at SMSCWS. If LSEND is issued when the logical work station is not in session, no operation occurs.

Name	Operation	Operand
[label]	LSEND	$\left[\text{RESP=}\begin{Bmatrix}\text{NO}\\ \text{DEF}\\ \underline{\text{EX}}\end{Bmatrix}\right]\left[\text{,DATA=}\begin{Bmatrix}\text{CTL}\\ \underline{\text{NORM}}\end{Bmatrix}\right]$ $\left[\text{,BRCKT=}\begin{Bmatrix}\text{NONE}\\ \text{MARK}\\ \underline{\text{NORM}}\end{Bmatrix}\right]\left[\text{,CHNGDIR=}\begin{Bmatrix}\text{YES}\\ \underline{\text{NO}}\end{Bmatrix}\right]$

RESP=NO
No response is requested.

DEF
A definite response is requested (definite response protocol).

EX
An exception response is requested (exception response protocol).

CTL
The message type is set for data-and-control.

NORM
The message type is set for data.

NONE
There is no bracket protocol.

MARK
A begin bracket or end bracket indication accompanies the message.

NORM
The message does not require a begin or end bracket.

YES
Send a change-direction indiator with the message.

NO
No change-direction indicator is required.

Program Storage Used: 10 bytes.

Condition Codes: One of the following is set when control is returned to the instruction following LSEND:

Hex Code	**Explanation**
01	No flags are set in BUBUSE.
02	Flags are set in BUBUSE.

Program Checks: Program check 9 can be set. Refer to "Communication Macro Instruction Traps" on page 5-41 for an explanation of the program check.

OPENSESS—Initiate a Session

OPENSESS enables the communication code to establish a session. The session can be established by the controller work station or the VTAM application program: the OPENSESS remains in effect until a CLOSSESS macro is issued and status is set in BUBUSE. If OPENSESS is issued when an OPENSESS is already in effect, no operation occurs.

When the work station initiates a session and contact is lost, the session is automatically reinitiated when contact is restored. When the VTAM application program initiates a session and contact is lost, the VTAM application program must reinitiate the session when contact is restored. If the initiation is done by the VTAM application program, the OPENSESS must be issued before the BIND message is received or else the communication code returns a negative response.

Name	Operation	Operand
[label]	OPENSESS	$\texttt{SESSID= (iden}\left[\texttt{,}\left\{\begin{array}{l}\texttt{len}\\ \underline{\texttt{8}}\end{array}\right\}\right]$ $\left[\texttt{,TYPE=}\left\{\begin{array}{l}\texttt{LU}\\ \underline{\texttt{HOST}}\end{array}\right\}\right]\left[\texttt{,RESP=}\left\{\begin{array}{l}\texttt{RRN}\\ \underline{\texttt{FME}}\end{array}\right\}\right]$

iden
The label of the field that contains the resource (VTAM application program) name.

len
A decimal number indicating the length of the resource name. If not specified, 8 is assumed.

LU
The session is initiated by the work station.

HOST
The session is initiated by the VTAM application program.

RRN
An RRN response will be requested when RESP=EX or RESP=DEF is coded for an LSEND macro.

FME
An FME response will be requested when RESP=EX or RESP=DEF is coded for an LSEND macro.

Program Storage Used: 20 bytes.

Program Checks: Program check 9 can be set. Refer to "Communication Macro Instruction Traps" on page 5-41 for an explanation of the program check.

TMEXIT—Communication Macro Exit

TMEXIT is used rather than an LEXIT instruction in controller application programs that use communication macros. If an LEXIT instruction is used, the communication macros do not function correctly. When TMEXIT is issued and a message is pending, the work station either enters the idle state or branches to the host system entry point.

Name	Operation
`[label]`	`TMEXIT`

Program Storage Used: 4 bytes.

Program Checks: Program check 9 can be set. Refer to "Communication Macro Instruction Traps" on page 5-41 for an explanation of the program check.

Communication Macro Instruction Traps

Conditions that cause a communication macro program check are indicated by the execution, within the communication code, of an invalid instruction when control is passed to the program check routine (the last message written to the log contains the invalid instruction). BUBRTRN may be tested in the program check routine to determine whether the program check was caused by the communication code (BUBRTRN contains a return address) or by some other part of the controller application program (BUBRTRN is set to zeros).

To initialize the communication code after any program check occurs, issue CLOSSESS TYPE=TERM and reestablish the session by issuing another OPENSESS after session-terminate bit has been set.

The following instruction traps may be encountered:

FFF0	(Communication Macro Error): Unexpected status resulted from an LWRITE: the device status is stored in BUBUSE.
FFF1	(Communication Macro Error): Unexpected status resulted from an LREAD: the device status is stored in BUBUSE.
FFF2	(Network Error): A data check was reported after an LREAD instruction.
FFF3	(User Protocol Error): An unexpected response was received for a non-data message: there is an error in the VTAM application program.
FFF4	(User Protocol Error): An unexpected message was received: the VTAM application program protocol and the communication macros are incompatible.
FFF5	(User Protocol Error): An unsupported form of Set and Test Sequence Numbers (STSN) was received or STSN was received at an unexpected time.
FFF6	(User Protocol Error): Two negative responses were received for a message, and RESP=DEF was not coded on the LSEND macro.
FFFF	(Communication Macro Error): An impossible condition was detected in the communication code.
300x	(Controller Application Program Error): The field pointers were set incorrectly for an LRECEIVE macro (x is the input segment number).
310x	(Controller Application Program Error): The field pointers were set incorrectly for an LSEND macro (x is the output segment number).

Part 2. Programming the BSC3 Host Link

This part of the manual presents the Binary Synchronous Communication (BSC) protocol for operating the 4700 host link. The next chapter presents a guide to 4700 BSC programming, followed by a chapter describing the 4700 Assembler macros for BSC operation.

Chapter 6. Guide to BSC3 Programming

BSC3 communication for the IBM 4700 Finance Communication System permits dedicated, multipoint connection of finance communication controllers to host systems using the Binary Synchronous Communication—Type 3 (BSC3) line discipline.

The operating system in the host system may be DOS/VS, OS/VS1, or OS/VS2. The required versions of the operating systems are those associated with the 4700 Independent Release being used. The teleprocessing method is BTAM. The data base access method used by the 4700 Host Support and the BSC3 service program is VSAM. The required versions of the access methods are those released with the operating system being used.

The host end of the teleprocessing link may be either an IBM 3704 or 3705 Communications Controller operating in emulator mode or a System/370 Model 115, 125, or 135 with the Integrated Communication Adapter (ICA). (The ICA requires the transparency feature for diskette creation.)

Controller Application Programming

BTAM, the ICA or host communication controller, and the 4700 communication controller regulate data traffic on the BSC3 link.

The controller application programs for BSC host communication are basically the same as those for SNA/SDLC. The differences are in the flags and fields used and the communication routine logic. BSC3 flag, field, and instruction use is described later in this chapter; the next chapter describes how to code the BSC3 commands. Communication options available to the controller application program are also described later in this chapter.

Controller Configuration

The procedure for controller configuration defines the 4700 subsystem including the communication protocol used by the controller. BSC3 support in the 4700 controller allows either single message or batch transmission from the controller to the host.

The only required operand is the TYPE parameter of the COMLINK configuration macro. TYPE must equal 1422 for BSC3 support. If only this operand is specified, the following defaults occur:

Explanation:	**Default:**	
General poll character	X'F0'	
System monitor unit select character	X'F1'	(Same unit select as for starter diskette)
Logical work station unit/select/ specific poll characters:		
Stations 2 through 9	X'F2' through X'F9'	
Stations 10 through 18	X'C1' through X'C9'	

Stations 19 through 27	X'D1' through X'D9'
Stations 28 through 35	X'E2' through X'E9'
Stations 36 through 44	X'81' through X'89'
Stations 45 through 53	X'91' through X'99'
Stations 54 through 60	X'A2' through X'A8'
Maximum length of write for a batch mode transmission	256 bytes
Maximum number of messages in a batch mode transmission	5
Station to receive data from unrecognized unit selection	Highest numbered station

Any of these defaults can be overridden by specifying the optional parameters shown on the COMLINK, STARTGEN, and STATION macros. Defaults not overridden are generated as shown above. Note that all default addresses are one byte; if any address is specified in the configuration as two bytes, all defaults and all specified one-byte addresses are padded to the right with one byte of X'40' to produce two-byte addresses.

Programming a BSC3 Host Link

The controller operates as a multipoint tributary (BSC type three) station. The host access method used is BTAM. Host application programs designed to communicate with other types of terminals such as the IBM 3271 Control Unit should be examined to determine the requirements for adding 4700 controllers to the network. Detailed information about the link-level commands and protocols supported by 4700 including the effect on BTAM control blocks, is included later.

For the work station to communicate with the host, you must first perform the following:

- Allow the CPU operand of the STATION configuration macro to default to Y (yes) or to specify CPU=Y, giving the work station the ability to use the host link.

- Specify 1422 in the TYPE= operand of the COMLINK configuration macro. This adds the BSC3 support to the controller load image and generates standard general polling and unit select characters for the controller and work stations.

- Code the BSC operands of the COMLINK, STARTGEN, and STATION configuration macros. These macros allow the default poll and select characters, batch message transmission parameters, and default unit select work station to be modified.

- When coding the controller application program that communicates with the host, specify the ACP operand on the BEGIN instruction. This allows the station using this program to begin executing at the ACP entry point when the controller receives Select from the host.

Addressing

The poll and select addresses for the controller and work stations are in the form:

cucuscENQ

where:

cucu

Is the controller address. This address is specified as a one-byte value from X'C0' to X'FF' either by the control operator, in response to the 00002 message or after entering the 041 command, or in the STRLNK instruction. This one-byte value is automatically expanded by the controller to a two-byte address used for general and specific polls. The controller also generates a two-byte select address by setting bit 1 of the poll address to 0. For example, if the operator enters X'C1', the controller recognizes X'C1C1' for the poll address and X'8181' for the select address.

After the address is entered by the control operator, the controller stores the address on the diskette where it remains until changed by the control operator.

sc

Is the one- or two-byte general poll address or specific poll and select address of the controller and work stations that communicate with the host system. The general poll address is specified on the COMLINK configuration macro instruction. The specific poll and select addresses for work stations are specified on the appropriate STATION configuration macros. The specific poll and select address of the starter diskette is fixed at X'F1'. The specific poll and select address of the system monitor on a diagnostic or operating diskette is also X'F1' for one-byte addressing or X'F140' for two-byte addressing, unless it is changed by specifying a different value on the STARTGEN configuration macro. Any byte is valid except those used as line control characters.

ENQ

Is the ENQ control character (X'2D') used to request a response from the controller.

Communication Characters and Protocols

In all cases, the controller does the actual communication with host. The controller may specify that:

- The message be transmitted as either transparent or nontransparent data.
- The message begins either with a header and an SOH (Start of Header) character, or without a header but with an STX (Start of Text) character.
- The controller expects a conversational response.
- The message should end with an ETB (End of Block) or an ETX (End of Text) character.
- The unit selection character or characters be inserted at the start of the message following the STX.

After these parameters are specified in a write control field (SMSBWC), and the message header length, if any, is specified in a header length field (SMSBOH), and the message data is pointed to, the controller assembles the actual message transmitted to the host. Data in the segment is unchanged after transmission.

When a message is read by the controller application program, the controller sets the appropriate bits in a read control field (SMSBRL), sets the message header length, if any, in a header length field (SMSBIH), removes all data link control characters, and passes only the message data to the application program.

The controller also handles all operations such as responses (ACK and NAK), EOT, RVI (Reverse Interrupt), forward abort, and ENQ.

The work station is informed of successful or unsuccessful operations through device status (SMSDST). Unsuccessful operations indicate that the controller has retried the operation and has been unable to complete it. Possible status codes are shown in Appendix A.

If contact is lost, all outstanding LREAD CP and LCHECK CP instructions end unsuccessfully, and the controller returns loss-of-contact status. The controller dispatches all other work stations at the asynchronous CPU entry point and returns loss-of-contact status in response to LREAD CP.

When contact is reestablished, the work station receives contact-established status if an LREAD CP is already outstanding; otherwise, the controller dispatches the program at the asynchronous CPU entry point and contact-established status is returned in response to the next LREAD CP. If contact is established and then lost again before the work station issues the LREAD, the work station might receive loss-of-contact status without receiving ready status.

Loss of contact may also be detected by testing the contact and adapter enabled flags in GMSIND. These flags allow the work station to determine whether the controller communication adapter failed or was disabled or whether the communication link failed.

Note that BSC3 conversational mode is not supported by the 4700.

Issuing Host Link Instructions

The instructions for programming a BSC3 link are the same as for an SNA link—LREAD, LWRITE, LCHECK, and the link control commands STPLNK and STRLNK. LREAD receives host messages, LWRITE sends messages to the host, and LCHECK tests for completion of an LWRITE instruction. These instructions all use the CP operand when communicating with the host.

The 4700 supports BSC3 conversational mode when established by the controller. The logical work station can issue an LWRITE data message requesting a converstional response. The host can then send a data message as a positive acknowledgment. However, the host cannot request a conversational response from the controller.

Your program can issue LREAD at any time; normally, it is in the ACP entry point routine. This is where the program begins execution when an unsolicited host message arrives, dispatching the host communication work station. If the station already has control, it can test for a host message by reading the asynchronous host interrupt flag in SMSACP.

When the program issues LREAD, the work station gives up control until one of the following happens:

1. SMSBRL and SMSBIH are set and the message data is in the work station's storage.

2. The read operation fails and status is returned in SMSDST.

3. The display operator breaks the read by indicating an attention.

Your program can issue LWRITE whenever the controller is in contact with the host. When there are no outstanding LWRITE CP instructions, the current LWRITE is issued immediately and program control returns to the next sequential instruction. If not in contact with the host, the controller returns the appropriate device status. If it is in contact with the host, the controller queues the current LWRITE and waits for a general or specific poll for that work station. When the controller detects a poll matching the issuing station, it transmits the LWRITE message.

Issuing an LWRITE does not protect the message data in the work station's storage; you should not let the program change any message data until the write operation completes.

The two ways to check for completion of a write operation are:

1. Issue the LCHECK instruction. If the LCHECK instruction has the TIO (test I/O) operand, the program sets a condition code indicating whether the LWRITE is complete or still executing. The program then continues executing immediately at the next sequential instruction.

   If you do not specify TIO, the work station gives up control until the LWRITE is complete; when the program resumes operation, it sets a condition code indicating whether or not the LWRITE was successful. An unsuccessful LWRITE also returns status in SMSDST.

2. Issue another LWRITE instruction. The work station waits until the first LWRITE completes. If it was successful, the controller queues the second LWRITE and the program continues with the next instruction. If the first LWRITE was unsuccessful, the second LWRITE does not complete. Instead, it returns an unsuccessful condition code and SMSDST status indicating "prior operation" and the reason why the first LWRITE failed.

BSC Use of Flags and Fields

A BSC3 host link uses the SMS and GMS fields described in this section. You should use the COPY instruction to label these flags and fields. *Do not* use absolute addressing for fields in segment 1, segment 14, or the fixed portion of segment 15.

Segment 1

Each work station has a segment 1 that contains numerous fields defined by the COPY DEFSMS instruction; refer to Volume 2 of the *4700 Controller Programming Library* for details on using COPY. Detailed information about the contents of the COPY-generated fields is in Appendix B, "COPY Fields."

Four of the fields for SNA/SDLC communication have been redefined for BSC3 communication. They are as follows (the SNA/SDLC name is shown in parentheses):

SMSBRL (SMSCRF)

This is the BSC3 read control field. When a message is received from the host, the controller sets the bits in this field to indicate the control characters that accompanied the data. SMSBRL is a one-byte field.

Bit:	**Value and Meaning:**	
0	0	Nontransparent data was received.
	1	Transparent data was received.
1	0	The message started with STX.
	1	The message started with SOH; the header length is in SMSBIH.
2	0	The message ended with ETX.
	1	The message ended with ETB.
3-7		Reserved

SMSBIH (SMSCRS)

This is the message header length. The value in this field is valid only if bit 1 of SMSBRL = 1. The message header is the first portion of the data in the input segment and is immediately followed by the message text. SMSBIH is a two-byte binary value.

SMSBWC (SMSCWF)

This is the BSC3 write control field. It is set by the work station prior to issuing an LWRITE to indicate the attributes of the control information to accompany the data. SMSBWC is a one-byte field.

Bit:	**Value and Meaning:**	
0	0	The data is not transparent.
	1	The data is transparent. This setting is mutually exclusive with SOH.

1	0	The message should begin with STX.
	1	The message should begin with SOH; the header length is in SMSBOH. This setting is mutually exclusive with transparent data.
2	0	The message should end with ETX.
	1	The message should end with ETB.
3	0	The unit selection characters should not be included with the message.
	1	The unit selection characters for this logical work station should be added to the message; these characters are added by the controller during transmission as the first one or two bytes following the STX.
4	0	No conversational write is expected.
	1	A conversational reply should result from this write. This is true only if the BSCOPT= parameter specified CONV on the COMLINK macro.
5-7	(Reserved)	

SMSBOH (SMSCWS)

This is the message header length. It is set by the work station before the work station issues LWRITE when SMSBWC specifies the SOH option. The message header is the first portion of the data in the output segment and is followed by the message text, if any. SMSBOH is a two-byte binary value.

The SMSACP field is common to both SNA/SDLC and BSC. Bit 0 is set to 1 if a message is pending for the work station.

The SMSDST field is used by both SNA/SDLC and BSC3 communication to return communication link status. Refer to Appendix A for a definition of the possible status codes.

The SMSIML field is used by both SNA/SDLC and BSC. When a read is completed, it contains the binary length of the data in the message (including the header, if any).

The SMSCST field is set to X'01' when the work station is informed that contact is established, and it is set to X'00' when the work station is informed of loss-of-contact.

The following fields in segment 1 are not used when communicating using BSC: SMSCRT, SMSCWT, and SMSCSR.

Segment 15

There is one segment 15 that is shared by all work stations in the controller. The global indicator byte (GMSIND) contains the status of the controller communication adapter and of the communication link. The following are the possible settings of GMSIND:

Bits:	Meaning:
01xx xxxx	Adapter enabled and contact established.
11xx xxxx	Adapter enabled and contact not established.
x0xx xxxx	Adapter disabled (contact flag ignored).

The global link ID byte (GMSLID) contains a number that indicates which link load module is loaded. If GMSLID contains X'05', the BSC3 link module is loaded; if GMSLID contains X'01' or X'02', an SNA/SDLC link module is loaded.

Message Buffering

The controller buffers any messages until the appropriate work station gains control and issues an LREAD instruction. The size and number of controller buffers are established during controller configuration using the CNL and CNB operands of the COMLINK macro. Unless the controller has been directed to chain read buffers for long messages (messages whose data portion is larger than the controller's input buffer size), the controller discards the excess data, passes the buffered data and incorrect length status to the work station, and sends an incorrect acknowledgment to the host system.

The two ways to direct the controller to chain read buffers are:

1. Specify BSCOPT=DRBC on the COMLINK configuration macro.
2. Design the host application program to insert ITB characters to segment the message into units that fit the controller buffers.

Refer to "Segmenting Long Messages" on page 6-11 for more information on buffer chaining and message segmenting. There are no buffers for messages sent to the host. The controller does maintain a buffer for each work station that communicates with the host. The controller uses this buffer to format the message control characters. However, the controller transmits the message itself directly from segment storage in the work station.

Message Headers

Messages can have message headers, or can comprise only a header. The header is indicated by the SOH character; the length of the header is in SMSBIH or SMSBOH. The message header is concatenated with the message data in the work station's storage. The message header starts in the leftmost portion of the input or output buffer and has a length equal to the value in SMSBIH or SMSBOH; the message begins in the next byte.

When the controller writes a message, it first transmits an SOH character, then the first part (a length equal to SMSBOH) of the output buffer data, followed by an STX character, the remaining data in the buffer, and then the ending character. The LWRITE length specifies only the header and message data; the controller adds all required control characters during transmission.

When the controller receives a message, it removes the SOH character, places the header data in the segment, removes the STX character, concatenates the message data with the header data in the segment, and also removes the ending character. When the operation ends, SMSIML contains the length of the header and message data, and SMSBIH contains the length of only the header data.

The total length of header and message data for a write operation may range from 1 to 4095 bytes. For a read operation, the total length is the buffer size, minus control characters.

Transmitting Single Messages

The work stations can send single messages to the host. If the message is in response to a specific poll, the host knows which work station sent the message.

However, if the message is in response to a general poll and you want the host to know which work station sent the message, you must specify the addressing characters option in SMSBWC. The controller then adds the unit selection character or characters of the applicable work station as the first one or two bytes following the STX. Your program does not need to allow space for these characters, because they are added during transmission.

Transmitting Batch Messages

Transmitting batch messages is an option that must be specified during configuration. If you specify this option, the link can be started in either batch or single message mode.

In batch mode, the controller sends messages in response to a general poll only; EOT sent in response to a specific poll. The controller sends an STX character followed by the unit selection address of a work station, the station's message followed by an IRS character, a unit selection address of a work station followed by that station's message, and so on. The controller continues to format messages until it either reaches the BSC operand limit or it receives a general poll; the controller then ends the batched message with an ETX.

Note that if a work station message includes a control character, the controller replaces it with an ENQ and ends transmission. In response to the next general poll, the controller resends all valid data messages and substitutes IRS characters for the failing address and data fields. For example, in a three-part message where the second data field contains a control character, the first message transmitted is as follows (excluding spaces):

STX addr1 text IRS addr2 text2 ENQ

The second message transmitted is:

STX addr1 text1 IRS IRS IRS IRS addr3 text3 ETX

Before using batch transmission, you must set SMSBWC to X'10'. Transparent data and headers are not allowed, and the trailer must be an ETX character. For example, three outstanding batch requests cause the batched message to have the following format:

STX addr1 text1 IRS addr2 text2 IRS addr3 text3 ETX

Request for Test

If the controller receives a message beginning with an SOH% (request for test), the controller acts only as a responder by returning an ACK1 to the first message and the appropriate ACK0 or ACK1 to all subsequent messages until an EOT is received. None of the messages are sent to any work station.

The controller does not validate the characters following SOH%. If an error occurs, the controller performs normal error recovery procedures unrelated to the request for test. A request for test or a subsequent test message does, however, add 1 to the normal communication link "request for test" statistical counter.

Starting and Stopping the Link

Either your application program (using STRLNK or STPLNK) or the 4700 system control operator (using the system monitor) can start or stop the host link. Use of STRLNK and STPLNK is described in this chapter; the following chapter describes the commands themselves.

BSC3 Line Protocol

The 4700 Finance Communication System runs as a tributary station on a BSC3 multipoint line. The following describes how the host system communicates with the 4700 subsystem using BSC3 data link control.

The host addresses the controller using a four- or five-byte sequence. The first two bytes are a repeated control unit address. For polling, this value ranges from X'C0' to X'FF'. For selection, bit 1 of the polling character is set to 0 (X'80' to X'BF'). For example, if the controller address is X'C1C1', the address used during unit selection is X'8181'. Following the poll/select control unit address is the general poll address or the specific poll and select address. The address ends with an ENQ character.

Polling

Polling is the way the host requests transmissions from the controller.

General Poll: Requests messages from the addressed controller. A general poll addresses the controller using a two-character polling address followed by a one- or two-character component address. The component address must have been defined during configuration as the general poll address. Normal replies to a general poll include:

- Data
- EOT (No data to send)

Following a general poll, the controller sends all queued data messages except those that the host cannot accept.

Specific Poll: Used to solicit messages from the work station addressed. A specific poll addresses the work station using the controller's polling sequence followed by the specific work station's unit selection characters specified during controller configuration.

Normal replies to a specific poll include:

- Data (From the polled work station only)
- EOT (When the polled work station has nothing to send)

Note: The controller sends no response unless it recognizes the work station address.

Selection

Selection is used when the host has data to transmit to the 4700. The selection sequence consists of the selection address followed by a one- or two-character work station identifier. The work station identifier is the unit selection characters specified during controller configuration, where you must also specify an alternate station to receive any messages for which no station is specified.

Replies to selection include:

- ACK0 (The 4700 is ready to receive the data)
- NAK (No buffers are available for the data)

Segmenting Long Messages

You can request the controller chain read buffers to accept long messages either by specifying BSCOPT=DRBC on the COMLINK macro, or by segmenting the messages.

The host application program segments messages by dividing the message into units smaller than a single controller read buffer and delimiting these units with ITB characters. The units are then transmitted as a single message. When the controller finds the ITB character, it automatically chains another available read buffer and continues to accept data. No data is lost unless the length of the message exceeds the total available buffer space; the controller then discards the excess data, sends the data in the read buffers along with an incorrect length indication to the work station, and returns an incorrect ACK to the host. If the message is received successfully, the total message, minus the ITB characters, is passed to the work station when it issues an LREAD.

For example, if the controller read buffers were each 100 bytes long and three read buffers were available at the controller, the host application program could transmit the following message:

STX	100 bytes of data	ITB	100 bytes of data	ITB	100 bytes of data	ETX

End of Message

Messages from the controller can end with either ETX or ETB, as defined by the controller application program protocol. This protocol may or may not correspond to the protocols used by other devices, depending on how the controller application program is coded.

Host application programs written for devices other than the controller may expect a series of messages from the terminal in which each message ends in ETB except the last, which ends in ETX. Because the controller serves multiple work stations, the host application program cannot assume that a message ending in ETX is the last message in a series. The host must either use a specific poll, or use a general poll and keep track of each work station's messages.

In response to a general poll, the controller transmits all queued messages from all work stations. The work stations have the option to add the unit selection address as the first one or two bytes following the STX. If this option is used, the host application can use a general poll and identify the originator of the message and the message series.

Batch Message Transmission

Messages from the controller may be transmitted in either single-message or batch-transmission mode. In single-message mode, one data message is transmitted in response to each LREAD Initial or LREAD Continue macro; either specific or general poll may be used when the controller is in this mode. In batch-message mode, the controller concatenates a series of logical work station data messages into a single transmission message and sends one transmission message in response to each Read Initial or Read Continue macro; only general poll may be used when the controller is in this mode. The batch-transmission message has the following format:

STX/addr1/text/IRS/addr2/text/.../ETX

where each data message is preceded by the unit selected address of the work station, and the address and text portions of the transmission message are separated by an IRS character. The maximum number of data messages in a batch-transmission message and the maximum size of a batch transmission message are specified during controller configuration.

Note that the maximum data length specified during controller configuration in the BSC operand of the COMLINK configuration macro does not include the link-control, addressing, and IRS characters. The buffer allocated in the host application program must be large enough to hold the data and these characters. The following formula can be used to determine the size of the host application program buffer:

$$m + 1 + n(a+1)$$

where:

m and n are the BSC operand values specified in the COMLINK macro and a is the length of the unit selection address (one or two characters).

Conversational Mode

The 4700 supports conversational bsc3 communication as described before under "Issuing Host Link Instructions" on page 6-4.

Message Formats

The host can receive BSC3 messages having the following host buffer formats:

STX/text/ETX
STX/text/ETB
STX/addr/text/ETX
STX/addr/text/ETB
DLE/STX/text/DLE/ETX
DLE/STX/text/DLE/ETB
DLE/STX/addr/text/DLE/ETX
DLE/STX/addr/text/DLE/ETB
SOH/hdr/STX/text/ETX
SOH/hdr/STX/text/ETB
SOH/hdr/STX/addr/text/ETX
SOH/hdr/STX/addr/text/ETB
STX/addr/text/IRS/addr2/text/.../ETX

The host can send messages having the following formats:

STX/text/ETX
STX/text/ETB
DLE/STX/text/DLE/ETX
DLE/STX/text/DLE/ETB
SOH/hdr/STX/text/ETX
SOH/hdr/STX/text/ETB
STX/text/ITB/text/ITB.../ETX

BTAM Terminal Definition Macros

Use the following DOS/VS terminal definition macros and operands when defining a controller. All operands not shown should be allowed to default.

```
BTMOD       SEPASMB,BUFFER,ERLOGIC,CANCEL,BSCS=YES,BSCMPT=YES,
            DECBEXT,RMSR,RESETPL,SS=NO

DTFBT       LINELST,CU=2703,DEVICE=BSC3,BUFCB,BUFNO,BUFL,
            SEPASMB,MODNAME,LERBADR,CTLCHAR=EBCDIC,RETRY=6,
            CONFIG=MPT,MODELST,LCBNUM

DFTRMLST    AUTOLST or AUTOWLST,3732,poll address
DFTRMLST    OPENLST,(select address)

DCB         DSORG=CX,MACRF=R,W),DDNAME,DEVD=BS,BUFNO,
            BUFL,BUFCB,EXLST,BFTEK,LERB
DFTRMLST    AUTOLST or AUTOWLST,(poll address,3737373737)
DFTRMLST    OPENLST,(select address)
```

BTAM Read and Write Macros

Use the following DOS/VS read and write macros when communicating with a 4700 controller:

Type of Read or Write	Op Type Code
Read Initial	TI
Read Continue	TT
Read Repeat	TP
Read Inquiry	TQ
Read Interrupt	TRV
Write Initial	TI
Write Continue	TT
Write Inquiry	TQ
Write Wait-Before-Transmit	TW
Write Reset	TR
Write Initial Transparent Text	TIX
Write Continue Transparent Text	TX
Write Initial Transparent Block	TIE
Write Continue Transparent Block	TE

Use the following OS/VS read and write macros when communicating with a 4700 controller:

Type of Read or Write	Op Type Code
Read Initial	TI
Read Continue	TT
Read Repeat	TP
Read Inquiry	TQ
Read Interrupt	TRV
Write Initial	TI
Write Initial and Reset	TIR
Write Continue	TT
Write Continue and Reset	TTR
Write Inquiry	TQ
Write Wait-Before-Transmit	TW
Write Reset	TR
Write Initial Transparent Text	TIX
Write Initial Transparent Text and Reset	TIXR
Write Continue Transparent Text	TTX
Write Continue Transparent Text and Reset	TTXR
Write Initial Transparent Block	TIE
Write Continue Transparent Block	TTE

Read and Write Operation Completion Codes

Figure 6-1 and Figure 6-2 show DECB bit settings at the completion of BTAM read and write operations for DOS/VS. Figure 6-3 and Figure 6-4 show DECB bit settings at the completion of BTAM read and write operations for OS/VS. Included with each figure is an explanation of the completion type and suggested procedures when each completion type is encountered.

The Effect of Link Traffic on BTAM

The following pages describe all possible combinations of BSC3 communication link traffic between BTAM and the controller.

All pages are based on one of two flow charts. The first chart in Figure 6-5 is for polling, the second in Figure 6-6 is for selection. Each description title contains a reference to a chart, and a number referring to an event on the chart (for example, the reference to number 1 on the polling chart indicates the receipt of the polling address).

Polling

The following are the BTAM macros that are used during polling of the controller:

Number on Chart 1	Transparent Mode		Nontransparent Mode	
	DOS/VS	OS/VS	DOS/VS	OS/VS
1	READ TI	READ TI	READ TI	READ TI
2				
3				
4	READ TI or READ TT	READ TI or READ TT	READ TI or READ TT	READ TI or READ TT
5	READ TT	READ TT	READ TT	READ TT
6				
7	WRITE TR	WRITE TR	WRITE TR	WRITE TR
8	READ TRV	READ TRV	READ TRV	READ TRV
9	WRITE TW	WRITE TW	WRITE TW	WRITE TW
10				
11				

	DECB Contents							
	Completion Code	Response Info	TCU Sense	Flag Byte	TP Code	Error Info		
Operation	Byte 0	8	16	24	28	29	Meaning	Action
Write Initial (TI)	7F						Normal completion — no error	2 or 3
	41		Bit 7		06		No response to addressing received	1
	41		Bit 7		20		No response to text received	1
	41			Bit 1	20		EOT response to text received when writing long messages	1
	41				20	Bit 7	NAK response to text received (ERP retries, then EOT)	1
	44				06		NAK response to selection received (controller out of buffers)	2
Write Continue (TT)	7F						Normal completion — no error	2 or 3
	41		Bit 7		20		No response to text received	3
	41			Bit 1	20		EOT response to text received	1
	41				20	Bit 7	NAK response to text received (ERP retries, then EOT)	1
	60				20		Incorrect ACK received	1

Action	1	Issue a Write Initial (TI) to retry the operation.
	2	Issue a Write EOT (TR) to reset the line.
	3	Issue a Write Continue (TT) to retry or to transmit more data.

Figure 6-1. Completion Codes and Suggested Actions for DOS/VS BTAM LREADs

	DECB Contents							
	Completion Code	Response Info	TCU Sense	Flag Byte	TP Code	Error Info		
Operation	Byte 0	8	16	24	28	29	Meaning	Action
Read Initial (TI)	7F						Normal completion – no error text received	1
	41		Bit 7		03		No response to polling received	2 or 3
	41			Bit 1	11	Bit 6	Error or exceptional condition – text ending with ENQ received	3
	42						Exceptional condition – RFT received	4
	54						Normal completion – no error, polling ended without positive response	2 or none
Read Continue (TT)	7F						Normal completion – no error, text received	1
	7F			Bit 1			Normal completion – no error, device operation ended, EOT received	None
	41		Bit 7				Text timeout	1
	42						Exceptional condition – RFT received	4
	62						Acknowledgment not recognized by remote device, ENQ received	1

Action	1	Issue a Read Continue (TT) to acknowledge text received without error.
	2	Issue a Read Initial (TI) to retry the operation.
	3	Issue a Write EOT (TR) to reset the line.
	4	Controller application accidently sent SOH %, ignore the RFT.

Figure 6-2. Completion Codes and Suggested Actions for DOS/VS BTAM LWRITEs

Operation and TP-Op Code (hex)	Completion Code (hex)	Other Indications (hex)	Meaning	Action
Read Response to Polling (0A)	41	Timeout	No index byte was received	1, 2, or 4
Poll (09)	7F	DECFLAGS: 04	Negative response to polling	1, 2, or 4
Poll (03)	41	Timeout	No terminal responded to polling	1, 2, or 4
	48		Initial read terminated by RESETPL macro	1, 2, or 4
Read Text (11)	41	Lost data, data check, or overrun	Text was received in error	2 or 3
	41		ENQ response to Read Continue	5
	41	DECFLAGS: 40	Text was received with an ENQ	1, 2, or 4

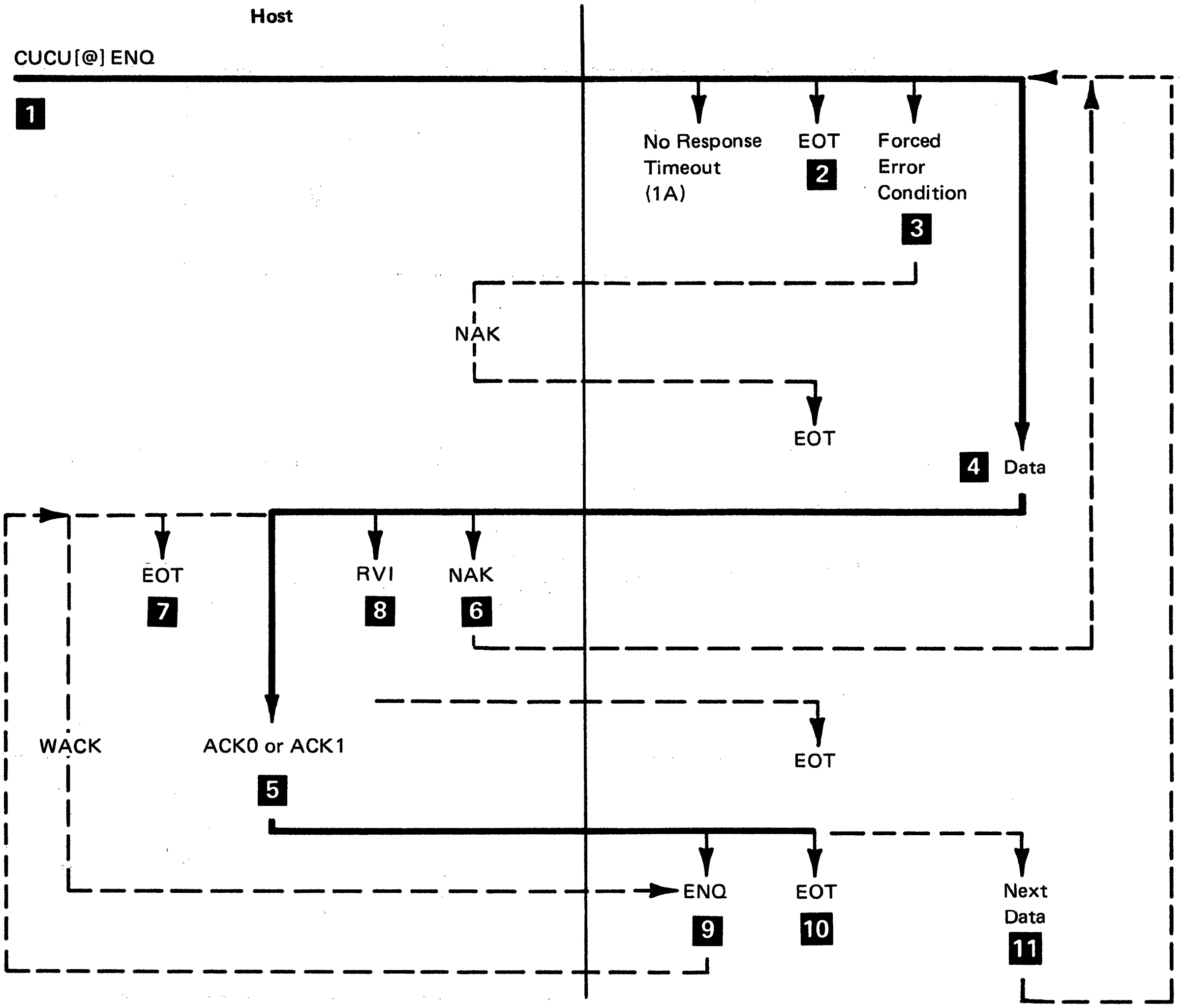
Action	1	Issue a Read Initial (TI) macro to poll the same or a different station.
	2	Issue a Write Reset (TR) macro.
	3	Issue a Read Repeat (TP) macro.
	4	Issue a Write Initial (TI) macro.
	5	ENQ requests retransmission of the last acknowledgment sent from the host (either the wrong acknowledgment, ACK0 or 1, or no acknowledgment was received at the controller).

Figure 6-3. Completion Codes and Suggested Actions for OS/VS BTAM LREADs

Operation and TP-Op Code (hex)	Completion Code (hex)	Other Indications (hex)	Meaning	Action
Read Response to Addressing (06)	7F	DECFLAGS: 04 DECRESPN: NAK	NAK received in response to addressing	1 or 2
	41	Timeout	No response received to addressing	1 or 2
Read Response to Text (BSC) (25)	7F	DECFLAGS: 20	Wrong acknowledgment received in response to text	1, 2, or 4
	41	DECFLAGS: 40 DECRESPN: NAK	NAK received in response to text	1, 2, or 3
	7F	DECFLAGS: 40 DECRESPN: EOT	EOT received in response to text	1 or 2
	41	Timeout	No response received to text	1, 2, or 4

Action	1	Issue a Write Initial (TI) macro to address the same or a different station.
	2	Issue a Write Reset (TR) macro to terminate selection.
	3	Issue a Write Continue (TT) macro.
	4	Issue a Write Inquiry (TQ) macro.

Figure 6-4. Completion Codes and Suggested Actions for OS/VS BTAM LWRITEs



Legend:

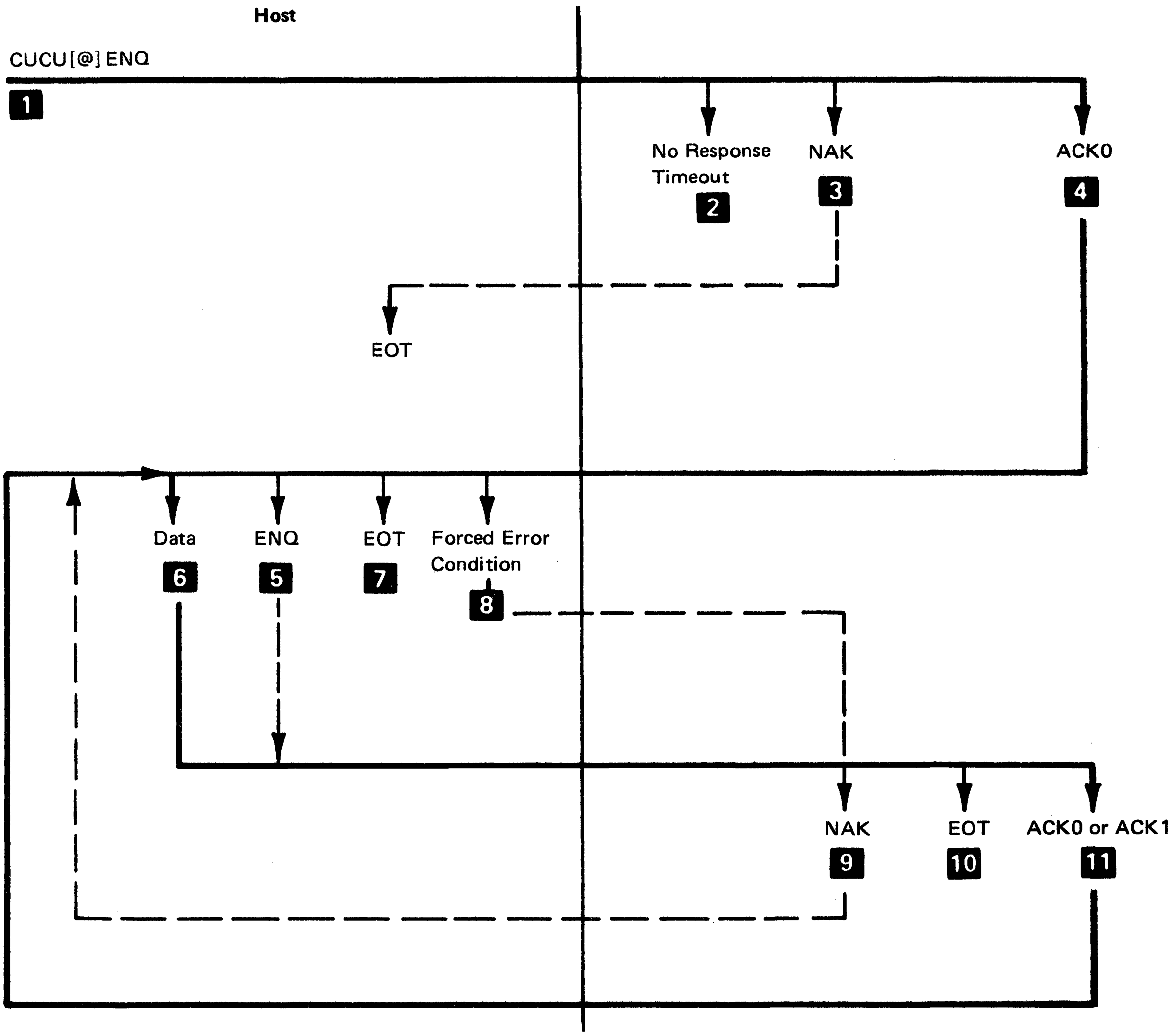
▬ Indicates normal data flow.

– – – Indicates abnormal data flow.

Notes:

1	CUCU is uppercase for poll. @ is one or two general or specific poll characters.
(1A)	The 4700 does not respond if it does not recognize the polling sequence.
2	EOT is the negative response to poll.
3	The forced error condition is a start character, data, and ENQ. It indicates a 4700 user data error.
4	Data is the positive response to a poll.
5	Alternating ACK0 and ACK1 is the normal response to data.
6	NAK indicates a transmission problem. The previous transmission is resent.
7	Forced ending from host.
8	Host termination of polling.
9	ENQ requests repeat of last acknowledgment. ENQ is sent when no acknowledgment is received in 3 seconds or the wrong ACK0 or ACK1 is received. ENQ is also sent in response to WACK (after seven WACK and ENQ sequences, EOT is sent). WACK is used to temporarily suspend data transmission to the host.
10	EOT indicates no more data to send.
11	·4700 sends all messages that it has queued in response to a general poll.

Figure 6-5. Chart 1: Polling Sequence Flow Chart



Legend:

Indicates normal data flow.

Indicates abnormal data flow.

Notes:

1 Lowercase is used for select CUCU. @ is one or two unit selection characters.

2 The controller does not respond if it does not recognize the control unit selection sequence. The BSCOPT=DSAD parameter (COMLINK) determines the response if the controller does not recognize the station address.

3 Negative response to selection (no buffers available in controller).

4 Positive response to selection (ready to receive data).

5 ENQ requests repeat of last acknowledgment. ENQ is sent when no acknowledgment is received in 3 seconds or the wrong ACK0 or ACK1 is received.

6 Data Transfer.

7 No more data to send.

8 Host application error (nontransparent data transmission ending with ENQ).

9 Transmission problem. The previous transmission is re-sent.

10 Forced ending. Indicates that no buffer was available for the previous message. The host would have to reselect the transmit message.

11 Alternating ACK0 and ACK1 is the normal response to data.

Figure 6-6. Chart 2: BSC3 Selection Sequence Flow Chart

Polling Sequence Sent from Host

(Figure 6-5, Number 1)

The polling sequence is used to request a block of data from the controller. The controller does not respond if it does not recognize the control unit address in the polling sequence. All controllers on the line receive the polling characters, but only the controller that recognizes its own polling characters responds to the poll (with a block of data, EOT, or a forced error condition). However, if the user has made an error in specifying the control unit address in the AUTOWLST operand of the DFTRMLST macro, the polling characters may not be recognized by any controller, and there may be no response to the polling sequence.

A wraparound polling operation (using an AUTOWLST) continues indefinitely if each terminal in the list responds with EOT to its polling sequence and has no data to send. A wraparound polling operation stops if the host receives a positive response from the controller (that is, an actual block of data is sent in response to the poll) or if no terminal at all responds to the polling sequence.

Negative Response to Polling

(Figure 6-5, Number 2)

EOT is the negative response to polling on a multipoint leased line. If an open polling list (AUTOLST) is used for polling, the polling operation stops when the end of the list is reached if all control units that were polled responded negatively. If a wraparound polling list (AUTOWLST) is used for polling and all control units that are polled respond negatively, the polling operation continues indefinitely until data is received in response to a poll or until no control unit responds to a poll.

Forced Error Condition Sent from a Controller

(Figure 6-5, Number 3)

The controller transmits a forced error condition (STX/data/ENQ) to the host when the controller application program's output data stream contains an invalid character (a BSC3 control character in a nontransparent message). This is a controller application program error.

Data Received at the Host

(Figure 6-5, Number 4)

A block of data is the positive response to polling on a multipoint leased line. For a Read Initial (READ TI) operation, the first byte of the input area in main storage contains the index byte of the polling list entry corresponding to the terminal that responded positively to the poll. This byte is placed into this position by the 3704/3705 in Emulator Mode; it is not transmitted by the remote device.

The formats of data placed into main storage as a result of a Read Initial (READ TI) operation are as follows:

Nontransparent Mode

Index byte/SOH/header/data/STX/text/ETB
Index byte/SOH/header/data/STX/text/ETX

Index byte/STX/text/ETB
Index byte/STX/text/ETX

Transparent Mode

Index byte/DLESTX/text/ETB
Index byte/DLESTX/text/ETX

The formats of data placed into main storage as a result of a Read Continue (READ TT) operation are the same as for a Read Initial operation, except that the TCU (3704/3705 in Emulator Mode) does not place the index byte in the first position of the input message area.

If general polling of the controller is being used, and the host application program wants to know which work station responded to the general poll, the controller application program should use the address-character-insertion option. The unit selection character for the work station (one or two bytes) is then placed immediately behind the STX in each block of text sent to the host.

For transparent mode, note that the TCU removes the DLE preceding the ETB or ETX before the message enters main storage.

If a block of data is sent from the controller to the host in response to a Read Initial operation by the host, the controller expects a response from the host for this block. If a block check character error occurs on the transmission, BTAM ERP sends NAK and retries the read operation until the block is correctly received or the BTAM ERP retry limit is reached. However, if no block check error occurs on the transmission, the Read Initial macro may be followed by a Read Continue (READ TT) macro, a Read Interrupt macro (READ TRV), a Write Reset macro (WRITE TR), or a Write Wait-Before-Transmitting (WRITE TW) macro.

If a Read Continue macro is issued after the Read Initial macro, the correct positive alternating acknowledgment (ACK0 or ACK1) is sent to the controller and a Read CCW is executed. The read operation either reads the next block of data or EOT from the controller. This Read Continue operation causes polling to continue within the polled controller but not on the line, because the line is placed into data transfer mode when a block of data is sent to the host in response to a Read Initial operation. When the READ TT macro is issued by the host application program, any work station on the polled controller can send data to the host, provided general poll is issued. For example, a host application program may issue the READ TT macro in a loop, checking for an EOT from the controller as the end-of-loop condition.

Alternating Positive Acknowledgment Sent from Host

(Figure 6-5,Number 5)

An alternating positive acknowledgment (ACK0 or ACK1) is the positive response to a block of data. A Read Continue (READ TT) macro transmits the proper alternating acknowledgment when a block of data is received correctly and then reads the next block of data or an EOT. If a Read Continue macro is issued after a Read Initial (READ TI) macro reads a block of data correctly, polling continues for any work station on the polled controller if a general poll was issued.

Negative Acknowledgment Sent to the Controller

(Figure 6-5, Number 6)

A NAK indicates that the block check character accumulated at the host is not equal to the block check character sent from the controller with this block of data.

Forced Ending by Host

(Figure 6-5, Number 7)

If the host has received a block of data from the controller (by a Read Initial or Read Continue operation), the controller waits for a response to that block. The host program, however, can end the session with this controller by transmitting an EOT, resetting the line to control mode.

Since the host sent no acknowledgment, the controller must resend the data. The host, however, must first issue either Read Initial or Write Initial to regain contact with a controller before more data can be transferred. The controller then resends the data after the next poll of either the controller itself, or the specific work station.

Reverse Interrupt Sent from Host

(Figure 6-5, Number 8)

If the host has received a block of data from the controller (by a Read Initial or Read Continue operation), the controller is waiting for a response to that block. However, the host application program can now inform the controller that it does not wish to receive any more blocks from the controller by issuing a Read Interrupt (READ TRV) macro. The RVI is sent to the controller, indicating that the host has correctly received the previous block but does not wish to receive any additional blocks. The Write RVI command is followed by a Read Text command in the CCW chain. The controller responds with EOT when it receives the RVI.

ENQ Transmitted to Host

(Figure 6-5, Number 9)

Three separate situations are represented by this portion of the Poll chart:

1. When a three-second time-out occurs because the controller received no link-level acknowledgment from the host for the last data block sent to the host, the controller transmits ENQ to the host. ENQ requests the host to retransmit its last link-level acknowledgment. The controller transmits an additional ENQ character for each additional 3-second time-out that it experiences.

   For example, if the host application issues a Read Initial macro and then waits 12 seconds before issuing a Read Continue macro, four 3-second time-outs occur, but none occur at the host communication controller. If seven consecutive three-second time-outs occur at the controller, it ends the operation and sends EOT to the host.

2. When the controller receives an incorrect alternating acknowledgment (ACK0 instead of ACK1 or ACK1 instead of ACK0) from the host for the last block of data sent to the host, the controller transmits ENQ to the host.

This ENQ character requests a retransmission of the last link-level acknowledgment. An incorrect alternating acknowledgment may indicate that an entire block of data was lost during transmission on the line. If ENQ is transmitted and the wrong alternating acknowledgment is received from the host, ENQ is retransmitted. If the error condition occurs seven consecutive times, the operation is terminated by sending EOT to the host.

3. The host application program may issue the Write Wait-Before-Transmitting (WRITE TW) macro to transmit WACK to the controller. The controller always responds with ENQ when it receives a WACK, causing BTAM to post normal completion (completion code X'7F' in the DECB). The host application program must check for a maximum of seven WACK=ENQ sequences, after which the controller transmits EOT to the host. The BTAM ERP is not involved in the transmission of this WACK, because the host application program issues the WRITE TW macro to transmit it.

EOT Transmitted to Host

(Figure 6-5, Number 10)

At normal end-of-data transmission from the controller to the host, the controller sends EOT in response to a Read Continue macro issued by the host application program.

Next Block of Data Sent to Host

(Figure 6-5, Number 11)

When the host application program issues a Read Continue macro, as described under "Data Received at the Host (see Number 4)," the controller transmits the next block of data to the host.

Selection

The following are the BTAM macros used during selection of a logical work station:

Number on Chart 2	**Transparent Mode**		**Nontransparent Mode**	
	DOS/VS	**OS/VS**	**DOS/VS**	**OS/VS**
1	WRITE TIE or WRITE TIX	WRITE TIE or WRITE TIX	WRITE TI	WRITE TI
2				
3	WRITE TR	WRITE TR	WRITE TR	WRITE TR
4				
5				
6	See Note.	See Note.	See Note.	See Note.
7	WRITE TR	WRITE TR	WRITE TR	WRITE TR
8				
9				
10				
11				

Note: Macros for 6 are listed before under "BTAM Write Macros for Data (Chart 2, Number 6)."

Selection Sequence Sent from Host

(Figure 6-6, Number 1)

The host issues a Write Initial macro to transmit the selection (addressing) characters and the first block of data to a specified controller. The controller does not respond to the selection sequence if it does not recognize the control unit address as its own. All controllers on the line receive the selection characters, but only the controller that recognizes its own control unit address responds to the addressing sequence (with ACK0 for a positive response or with NAK for an out-of-buffers response). However, if the user has made an error in specifying the control unit address of the controller in the OPENLST operand of the DFTRMLST macro, the selection characters may not be recognized by any controller, and there may be no response to the addressing sequence.

No Response to Selection

(Figure 6-6, Number 2)

All controllers on the line receive the selection characters, but only the controller that recognizes the repeated control unit address (CUCU in the selection sequence) responds to the selection. If no controller at all on the line responds to the selection characters, a time-out on Read Response to addressing occurs.

If BSCOPT=DSAD has not been specified, any station address in the selection sequence is accepted, because the user must specify a default station ID to receive all messages for which there is no station ID in the controller.

If BSCOPT=DSAD has been specified on the COMLINK configuration macro instruction, and if two-byte, duplicate station addresses are being used, the 4700 controller will not respond to unduplicated station addresses.

Negative Response to Selection

(Figure 6-6, Number 3)

When a 4700 controller recognizes the control unit address in the selection sequence but has no buffers available to receive the data from the host, the 4700 sends NAK in response to the selection sequence.

After receiving the negative response, the host issues the Write Reset macro (WRITE TR) to send EOT to the controller. EOT resets the line to control mode, forcing all controllers to monitor the line for polling and selection characters. The host must then issue either Read or Write Initial to start new activity on the line.

After issuing WRITE TR, the host program should recover the operation according to the type of host and controller programs in process. If the host and controller programs are not synchronized with each other for Read and Write operations, the host program should first poll (with a Read Initial operation) the controller and then issue Read Continue macros to receive all available host data (that is, until EOT is received). This allows other work stations that are waiting to complete their WRITE operations to send their data to the host and then receive data from the host, clearing the controller buffers for more data from the host. The host application can now issue the Write Initial macro that previously resulted in a negative response to selection.

If the host and controller programs are synchronized for Read and Write operations, the host program should reissue the Write Initial macro a specified number of times (seven, for example) and test for responses to selection in a loop that also issues WRITE TR if a negative response to selection is received. If the host program completes the specified number of Write Initial operations without getting a positive response to selection, it should consider the situation to be a permanent error condition, because the Read and Write operations are probably not synchronized properly.

When you first code and test a host application program whose read and write operations are synchronized with those of a controller application program, you should consider any negative response to selection as a permanent error condition. Do not attempt a specified number of retries of the Write Initial operation in a loop. This notifies the controller of all such out-of-buffer conditions that occur (on a Write Initial operation) in the controller.

Positive Response to Selection

(Figure 6-6, Number 4)

When a controller recognizes the control unit address in the selection sequence and has buffers available for the data to be sent by the host, the controller transmits ACK0 as the positive response to selection.

ENQ Transmitted to the Controller

(Figure 6-6, Number 5)

The ENQ character sent from the host asks the controller to transmit its last acknowledgment again. This ENQ is transmitted by the BTAM ERP in two separate situations:

1. The host experiences a three-second time-out waiting for the link-level acknowledgment for the block of data that was just transmitted from the host to the controller.

2. An incorrect alternating acknowledgment (ACK0 instead of ACK1 or ACK1 instead of ACK0) has been received at the host. An incorrect alternating ACK may indicate that an entire block of data was lost during transmission on the line.

BTAM Write Macros for Data

(Figure 6-6, Number 6)

	Transparent Mode		**Nontransparent Mode**	
Data	**DOS/VS**	**OS/VS**	**DOS/VS**	**OS/VS**
SOH/header/data/ STX/text/ETB			WRITE TI WRITE TT	WRITE TI WRITE TT
SOH/header/data/ STX/text/ETX			WRITE TI WRITE TT	WRITE TI WRITE TIR WRITE TT WRITE TTR
STX/text/ETB			WRITE TI WRITE TT	WRITE TI WRITE TT
STX/text/ETX			WRITE TI WRITE TT	WRITE TI WRITE TIR WRITE TT WRITE TTR
DLESTX/transparent text/DLEETB	WRITE TIE WRITE TE	WRITE TIE WRITE TTE		
DLESTX/transparent text/DLEETX	WRITE TIX WRITE TX	WRITE TIX WRITE TIXR WRITE TTX WRITE TTXR		

A WRITE macro with the Reset Option in OS/VS BTAM (WRITE with TIR, TIXR, TTR, or TTXR) transmits EOT along the line after reading the response to the block of data transmitted to the controller.

However, the reset function is *not* performed (that is, EOT is *not* transmitted by the execution of one of these WRITE macros) if a permanent error occurred during the operation.

When the host transmits an EOT character on a leased multipoint line, all controllers are reset to control mode on the line. This means that all controllers are now monitoring the line for polling or addressing characters. Therefore, the next operation issued by the host must be a Read Initial or Write Initial operation.

EOT Transmitted by Host

(Figure 6-6, Number 7)

When the host has transmitted all of the required data blocks to a particular controller, the host application issues the Write Reset macro (WRITE TR) to send EOT along the line. This EOT resets the line to Control mode to make all controllers on that line monitor the line for polling or selection characters. Therefore, the next macro that the host issues must be for a Read Initial or Write Initial operation.

Forced Error Condition Sent by Host

(Figure 6-6, Number 8)

A forced error condition occurs when the host application program attempts to send a nontransparent block of data that contains a BSC3 control character. The forced error condition is sent to the controller, which returns a NAK. No data is passed to the controller application program.

Negative Acknowledgment Sent from the Controller

(Figure 6-6, Number 9)

A NAK sent from the controller in response to a block of data indicates that the block check character accumulated at the controller is not equal to the block check character sent by the host with this block of data.

Forced Ending by the Controller

(Figure 6-6, Number 10)

The controller transmits EOT to the host in response to a block of data if no controller buffer was available for that block. (This is the same situation described by Figure 6-6, Number 3, except that in this case it occurs on a Write Continue operation rather than on a Write Initial operation from the host.)

The recommended recovery action to be taken by the host application program in this case depends upon the type of application programs that are executing in the host and in the controller:

1. If the host application program and the controller application programs are not synchronized with each other for Read and Write operations, the host application program should issue a Write Reset (WRITE TR) macro to

transmit EOT along the line, resetting it to control mode. The host program should then issue a general poll (a Read Initial operation) of this controller and should then issue Read Continue macros in a loop until no more data is received (that is, until EOT is received). This allows the work stations in the controller that are waiting for completion for their WRITE operations to send their data to the host and then get control at their asynchronous entry points to receive data. This clears the controller buffers so that more data can be received. The host application program should issue a Write Reset (WRITE TR) macro to transmit EOT along the line, resetting it to Control mode. The host can then issue a Write Initial operation to transmit the block of data that previously resulted in the host's receiving an EOT (forced ending).

2. If the read and write operations in interacting host and controller application programs must be synchronized, the host program should issue a Write Reset (WRITE TR) macro to transmit EOT to the line, resetting it to control mode. The host application program then issues a Write Initial operation for the same block of data that caused a forced ending. This Write Initial macro should be issued in a program loop that executes a limited number of times (seven, for example). This loop should also test for a negative response to selection and should issue a WRITE TR macro if the host receives a negative response. If the program executes this loop the specified number of times without getting a positive response to selection, consider this a permanent error condition; the read and write operations in the two application programs are most likely not synchronized properly.

To determine all possible out-of-buffer conditions when testing a new host application program that must synchronize with a controller program, you could allow the controller to force an end, rather than attempt retries of Write Reset/Write Initial operations.

Positive Acknowledgment Sent by the Controller

(Figure 6-6, Number 11)

The correct positive alternating acknowledgment (ACK0 or ACK1) sent by the controller in response to a block of data sent from the host means that the block check character accumulated at the controller for this block is equal to the one sent with this block by the host.

Chapter 7. The 4700 BSC3 Communication Instructions

These are the instructions you use in your application program to communicate with the host over a BSC3 link. They start and stop the communication link, perform read and write operations to and from the work stations, and check the status of the read and write operations. The assembler communication instructions are:

- LCHECK CP—Check the status of the host system.
- LREAD CP—Read data from the host.
- LWRITE CP—Send data to the host.
- STPLNK—Stop the BSC3 host link.
- STRLNK—Start or wrap test the BSC3 host link.

Each instruction description in this chapter begins with a general description of the instruction followed by the coding syntax, the operand descriptions, condition codes that the instruction can set and their meanings, and any program checks.

For a description of the coding syntax and rules, refer to Chapter 4, "The 4700 Assembler Communication Instructions."

LCHECK CP—Check the Host Status

LCHECK synchronizes data transmission and determines status by checking all outstanding write operations to the host system. If all operations are not completed, LCHECK causes the work station to wait for their completion. When the operations are completed, the condition code is set, the logical station is taken out of the wait state, and a status code is stored in SMSDST. If there were no outstanding write operations, a status code of 0 is returned.

Name	Operation	Operand
`[label]`	`LCHECK`	`CP [,TIO]`

CP

Specifies that the write operation to the host system is to be checked.

TIO

Indicates that a test I/O operation is to be performed. The application program retains control whether the I/O operation being checked has completed or not.

Condition Codes: When completed, LCHECK sets one of the following condition codes:

Hex Code:	Meaning:
01	LCHECK completed successfully.
02	A status code is returned in SMDST. See the status code descriptions in Appendix D, "Link Status Codes."
03	LCHECK is still in progress (valid only when TIO is specified).

Program Checks: None are set by LCHECK.

LREAD CP—Read Data from the Host

This LREAD instruction reads data that is transmitted over the BSC3 link from the host. LREAD CP reads data into the location specified by operand 2. Once started, LREAD causes the station to wait until the data transmission is completed. Status of the operation is stored before processing continues. LREAD operates independently of any LWRITE operations.

LREAD ends when one of the following conditions occurs:

- End of the message is reached.
- End of the input field is reached.
- End of the segment is reached.
- The operator signals attention.
- A loss-of-contact condition is encountered.

The message length is stored in SMSIML. If LREAD ends in any way other than an attention or loss-of-contact condition, status is stored in SMSDST. LREAD stores the message type and any transmitted message header in the read control fields SMSBRL and SMSBIH.

```
Name     Operation  Operand

                      ⎧ defld2           ⎫
                      ⎪ seg2             ⎪
[label]  LREAD    CP, ⎨ seg2,disp2,len2  ⎬
                      ⎪ (reg2)           ⎪
                      ⎩ (defrf2)         ⎭
```

CP
Specifies a read from the host system.

operand 2
Is the location to contain the read data. If you specify a length of zero, data is read into operand 2 beginning at the specified location and continuing to the end of the segment.

Condition Codes: When completed, LREAD sets one of the following:

Hex Code:	**Meaning:**
01	LREAD completed successfully.
02	A status code is returned in SMDST. See the status code descriptions in Appendix D, "Link Status Codes."

Program Checks: 1, 2, or 9 can be set.

LWRITE CP—Write Data to the Host

The LWRITE instruction writes the data in operand 2 to the host. Before issuing LWRITE, you must place the message type and message header length (if any) in the write control fields SMSBWC and SMSBOH. If you specify a length of zero, the LWRITE is ignored and the program processing continues with the next sequential instruction.

The following is the format and description of the LWRITE instruction:

```
Name     Operation  Operand

                      ⎧ defld2           ⎫
                      ⎪ seg2             ⎪
[label]  LWRITE   CP, ⎨ seg2,disp2,len2  ⎬
                      ⎪ (reg2)           ⎪
                      ⎩ (defrf2)         ⎭
```

CP

Specifies a write operation to the host system.

operand 2

Defines the data to be written. DEFRF must be specified in parentheses. The length of the data to be written is from 1 to 4095 bytes. If you specify a *seg2* address, set the secondary field pointer (SFP) to address the beginning of the field and the primary field pointer (PFP) to an address one byte beyond the end of the message.

A *seg2* address generates a two-byte LWRITE machine instruction; otherwise, the LWRITE machine instruction is 6 bytes long.

Condition Codes: When LWRITE completes, the controller sets one of the following condition codes:

Hex Code:	**Meaning:**
01	LWRITE completed successfully.
02	A status code is returned in SMDST. See the status code descriptions in Appendix D, "Link Status Codes."

Program Checks: 1, 2, 3, 9, 0F, 10, or 11 may be indicated.

STPLNK—Stop the BSC3 Host Link

STPLNK deactivates the communication link to the host system. STPLNK points to a 7-byte parameter list that you select with operand 2. This parameter list can be the same one used by STRLNK; however, the parameters are neither used nor changed by STPLNK.

```
Name     Operation  Operand

                    ( defld2      )
                    ( defcon2     )
[label]  STPLNK    {  seg2,disp2   }
                    ( (reg2)      )
                    ( (defrf2)    )
```

operand 2
Defines the start of the parameter list. Any length you specify is ignored, and the first 4 bytes are assumed as the parameter list. The parameter list format is shown in Figure 7-1 on page 7-12.

Condition Codes: The code is not changed.

Program Checks: 1, 2, 9, or 27 can be set.

Programming Note: The STPLNK parameter list can be the same as that used by a Start Link (STRLNK) instruction; STPLNK neither uses or changes the parameter list.

STRLNK—Start the BSC3 Host Link

STRLNK activates the communication link to the host system if the communication link had previously been stopped by a STPLNK instruction, or performs a single wrap test of the host link. STPLNK points to a 4-byte parameter list that you create to define the link and wrap-test options, and then select with operand 2.

Name	Operation	Operand

`[label]`	`STRLNK`	`defld2`
		`defcon2`
		`seg2,disp2`
		`(reg2)`
		`(defrf2)`

operand 2

The first 4 bytes of this location are assumed to be the parameter list. If you specify *0* for operand 2, STRLNK uses the parameter list defined by the last STRLNK instruction; otherwise, the COMLINK configuration macro parameters are used (refer to the COMLINK macro description in Volume 6 of the *4700 Controller Programming Library*). Figure 7-1 shows the required format for the parameter list.

Condition Codes: The code is not changed.

Program Checks: 1, 2, or 9 can be set.

Flag 1	**Flag 2**	**CUA**	**Flag 3**
Byte 0	1	2	3

Flag 1 — A 1-binary number that describes the link options as follows:

Bit	*Explanation*
0	Reserved
1	Set to one (1) to perform a single external wrap test.
2	Set to 1 for low speed.
3	Set to 1 for high speed.
4	Set to 1 for a wrappable modem.
5	Set to 1 for nonwrappable modem.
6	Set to 1 for non-NRZI.
7	Set to 1 for NRZI.

Flag 2 — a 1-byte binary number number that describes the link options as follows:

Bit	*Explanation*
0	Set to 1 if CUA is present.
1	Set to 1 if flag byte is present.
2	Set to 1 for switched line.
3	Set to 1 for nonswitched line.
4	Set to 1 for connect data set to line.
5	Set to 1 for data set ready.
6	Set to 1 for permanent request to send.
7	Set to 1 for controlled request to send.

If bit 2 is 1, bit 6 is ignored.

CUA — a 1-byte polling address.

Flag 3 — a 1-byte number that describes the link options as follows:

Bit	*Explanation*
0-5	Reserved
6	Set to 1 for batch mode.
7	Set to 1 for single-message mode.

If both bits are zero the transmission mode is unchanged.

Figure 7-1. Parameter List Used by STPLNK and STRLNK (BSC3)

Appendix A. Machine Instruction Formats

This appendix describes the machine instruction formats for the 4700 assembler instructions included in this volume. See *Volume 1 - General Controller Programming* for an explanation of the symbols used in this appendix.

ASSIGN

4C	0	S_2	D_2	
0	8	12	16	31

ASSIGN

79	00	00	R_2	0	
0	8	16	24	28	31

ASSIGN

79	10	00	R_2	0	D_2	
0	8	16	24	28	32	47

LCHECK (AL-WAIT)

66	30	
0	8	15

LCHECK (AL-NOWAIT)

66	B0	
0	8	15

LCHECK (CP)

32	W	0	
0	8	12	15

LCNTRL ASN (NO-force option)

67	5	S_2	
0	8	12	15

LCNTRL ASN (Force option)

67	6	S_2	
0	8	12	15

LCNTRL STRL (Segment addressing)

6A	D	S_2
0	8	12 15

LCNTRL STRL (Fixed-field addressing)

6B	D	S_2	L_2	D_2	00
0	8	12	16	28	40 47

LCNTRL STRL (Fixed-field addressing)

78	4B	0	D	R_2	0
0	8	16	20	24	28 31

LCNTRL STRL (Fixed-field addressing)

78	5B	0	D	R_2	0	D_2	L_2
0	8	16	20	24	28	32	48 63

LCNTRL STPL (Segment addressing)

6A	4	S_2
0	8	12 15

LCNTRL STPL (Fixed-field addressing)

6B	4	S_2	L_2	D_2	00
0	8	12	16	28	40 47

LCNTRL STPL (Fixed-field addressing)

78	4B	04	R_2	0
0	8	16	24	28 31

LCNTRL STPL (Fixed-field addressing)

78	5B	04	R_2	0	D_2	L_2
0	8	16	24	28	32	48 63

LCNTRL VONL

66	70
0	8 15

LCNTRL VOFF

66	90
0	8 15

LCNTRL SSD

66	20
0	8 15

LCNTRL RSD

66	C8
0	8 15

LCNTRL APL (Segment addressing)

6A	8	S_2
0	8	12 15

LCNTRL APL (Fixed-field addressing)

6B	8	S_2	L_2	D_2	00
0	8	12	16	28	40 47

LCNTRL APL (Fixed-field addressing)

78	4B	08	R_2	0
0	8	16	24	28 31

LCNTRL APL (Fixed-field addressing)

78	5B	08	R_2	0	D_2	L_2
0	8	16	24	28	32	48 63

LCNTRL SENS (Clear)

67	A	S_2
0	8	12 15

LCNTRL SENS (No Clear)

67	E	S_2
0	8	12 15

LREAD AL (Segment addressing)

67	1	S_2
0	8	12 15

LREAD AL (Segment addressing)

78	0B	01	S_2	0	D_2	L_2
0	8	16	24	28	32	48 63

LREAD AL (Segment addressing)

78	1B	01	R_2	0
0	8	16	24	28 31

LREAD AL (Segment addressing)

78	2B	01	R_2	0	D_2	L_2
0	8	16	24	28	32	48 63

LREAD CP

30	0	S_2
0	8	12 15

LREAD CP

78	63	00	S_2	0	D_2	L_2
0	8	16	24	28	32	48 63

LREAD CP

78	73	00	R_2	0
0	8	16	24	28 31

LREAD CP

78	83	00	R_2	0	D_2	L_2
0	8	16	24	28	32	48 63

LWRITE AL (Fixed-field addressing)

6B	0	S_2	L_2	D_2	00
0	8	12	16	28	40 47

LWRITE AL (Fixed-field addressing)

78	4B	0D	R_2	0
0	8	16	24	28 31

LWRITE AL (Fixed-field addressing)

78	5B	0D	R_2	0	D_2	L_2
0	8	16	24	28	32	48 63

LWRITE AL (Segment addressing)

6A	0	S_2
0	8	12 15

LWRITE CP

31	0	S_2
0	8	12 15

LWRITE CP

33	0	S_2	L_2	D_2	00
0	8	12	16	28	40 47

LREAD CP

78	95	00	S_2	0	D_2	L_2
0	8	16	24	28	32	48 63

LWRITE CP

78	A5	00	R_2	0
0	8	16	24	28 31

LWRITE CP

78	B5	00	R_2	0	D_2	L_2
0	8	16	24	28	32	48 63

STPLNK

50	9	S_2	D_2
0	8	12	16 31

STPLNK

7A	25	00	R_2	0
0	8	16	24	28 31

STPLNK

7A	35	00	R_2	0	D_2
0	8	16	24	28	32 47

STRLNK

50	1	S_2	D_2
0	8	12	16 31

STRLNK

7A	26	00	R_2	0
0	8	16	24	28 31

STRLNK

7A	36	00	R_2	0	D_2
0	8	16	24	28	32 47

Appendix B. COPY Fields

This appendix gives detailed listings of the communication field and COPY parameter lists created by the 4700 COPY assembler instruction. The six-character *DEFxxx* parameter list names shown in this appendix are the operands you must code in the COPY instruction's *copyfilename* operand to create the respective parameter lists, as follows:

```
          (label)   COPY    DEFAMS
```

This COPY DEFAMS statement creates the AMS parameter list needed for operating the alternate (ALA) SNA/SDLC link. Refer to the *Controller Programming Library, Volume 1*, for descriptions of parameter lists not shown in this appendix.

DEFALP

When specified as the *copyfilename* operand on the COPY instruction, DEFALP produces the following ALP parameter definition:

```
      LSPACE
********************************************************************
  ALTERNATIVE LINE ALTER PHYSICAL ADDRESS PARAMETER DEFINITION
********************************************************************
      LSPACE
ALPNID    DEFLD DEFALPS,,2       NETWORK IDENTIFIER OF THE DEVICE
ALPNID1   DEFLD DEFALPS,ALPNID,1 NETID BYTE ONE
ALPNID2   DEFLD DEFALPS,,1       NETID BYTE TWO
ALPPPA    DEFLD DEFALPS,,2       PHYSICAL POLL ADDRESS
ALPPPA1   DEFLD DEFALPS,ALPPPA,1 PHYSICAL POLL ADDRESS BYTE ONE
ALPPPA2   DEFLD DEFALPS,,1       PHYSICAL POLL ADDRESS BYTE TWO
ALPPSA    DEFLD DEFALPS,,2       PHYSICAL SELECT ADDRESS
ALPPSA1   DEFLD DEFALPS,ALPPSA,1 PHYSICAL SELECT ADDRESS BYTE ONE
ALPPSA2   DEFLD DEFALPS,,1       PHYSICAL SELECT ADDRESS BYTE TWO
*                    ADDITIONAL POLL/SELECT ENTRIES
                     FOLLOW
      LSPACE
```

The guidelines for defining the location of the ALP fields in the appropriate segment are the same as explained under the DEFAMS operand description.

DEFAMS

When specified as the *copyfilename* on the COPY instruction, DEFAMS produces the following definition for the alternate link machine segment:

```
      LSPACE
************************************************************
            ALA MACHINE SEGMENT
************************************************************
      LSPACE
AMSNID   DEFLD    DEFAMSS,,2         Network Identifier
AMSARC   DEFLD    DEFAMSS,,2         Read Control Fields
AMSARCF  DEFLD    DEFAMSS,AMSARC,1   Read Flags
AMSARCT  DEFLD    DEFAMSS,,1         Read Type
AMSAWC   DEFLD    DEFAMSS,,2         Write Control Fields
AMSAWCF  DEFLD    DEFAMSS,AMSAWC,1   Write Flags
AMSAWCT  DEFLD    DEFAMSS,,1         Write Type
AMSAEN   DEFLD    DEFAMSS,,1         Write Event Number
AMSAFN   DEFLD    DEFAMSS,,1         Failing Write Event
                  Number
AMSASQ   DEFLD    DEFAMSS,,1         Sequence Number
AMSARCE  DEFLD    DEFAMSS,,1         Read Flags Extension
AMSAWCE  DEFLD    DEFAMSS,,1         Write Flags Extension
AMSANRT  DEFLD    DEFAMSS,,1         Next Read Type
```

Because the segment number DEFAMSS is undefined, the application program must precede the COPY...DEFAMS statement with:

```
  DEFAMSS  EQUATE   seg   AMS Segment Number
```

where seg is the number of the segment that is to contain the parameter list. The segment must not be segment 14, but may be the number or the label of a previously-specified EQUATE.

The displacement to the beginning of the segment containing AMS is, by default, the sum of the location and length of the last DEFLD that referred to the specified segment. If no previous DEFLDs have referred to the segment, the AMS starts at a displacement of 0.

The application can explicitly control the location of AMS in the segment by coding:

```
  DEFAMSS   EQUATE   seg              AMS Segment Number
  DEFAMSD   DEFLD    DEFAMSS,disp,0   AMS Displacement
         COPY     DEFAMS
```

where disp is either the value of the desired displacement, the label of an EQUATE, or the label of a previously defined DEFLD that refers to seg. As described above, the starting location of AMS is the sum of the location and length of the last DEFLD that referred to the specified segment.

> **Note:** Regardless of what technique the application program uses to define the location of AMS, the definition must match the segment and displacement specified in the STATION configuration macro. This applies to AMS only.

DEFASN

When specified as the *copyfilename* operand on the COPY instruction, DEFASN produces the following ASN parameter definition:

```
         LSPACE
***********************************************************************
           ALTERNATIVE LINE ASSIGN PARAMETER DEFINITION
***********************************************************************
         LSPACE
ASNNID   DEFLD DEFASNS,,2        NETWORK IDENTIFICATION FIELD
ASNNID1  DEFLD DEFASNS,ASNNID,1 NETWORK ID - BYTE ONE
ASNNID2  DEFLD DEFASNS,,1        NETWORK ID - BYTE TWO
ASNSTID  DEFLD DEFASNS,,1        STATION IDENTIFIER
         LSPACE
```

The guidelines for defining the location of the ASN fields in the appropriate segment storage are the same as explained under the DEFAMS parameter list description.

DEFAST

When specified as the *copyfilename* operand on the COPY instruction, DEFAST produces the following Start/Stop Line parameter definition:

```
         LSPACE
*****************************************************************
     ALTERNATIVE LINE, START-STOP LINE, PARAMETER DEFINITION
*****************************************************************
         LSPACE
ASTFUN   DEFLD DEFASTS,,1    START/STOP FUNCTION TYPE
ASTNDF   EQUATE X'00'        START ALL LINES CPGENED AS ACTIVE
ASTSTOP  EQUATE X'00'        STOP ALL ACTIVE LINES
*  UTILIZE EXISTING LINE DEFINITION BYTES
         LSPACE
ASTDEF   EQUATE X'20'        START OR STOP ALL ACTIVE LINES
*  UTILIZE LINE DEFINITION BYTES SUPPLIED IN PARAMETER LIST
         LSPACE
ASTSIN   EQUATE X'40'        START/STOP THE LINE INDICATED IN AMSNID
*  UTILIZE EXISTING LINE DEFINITION BYTES
         LSPACE
ASTSINPA EQUATE X'60'        START/STOP THE LINE INDICATED IN AMSNID
*  UTILIZE LINE DEFINITION BYTES SUPPLIED IN PARAMETER LIST
         LSPACE
ASTACTND EQUATE X'80'        START ALL LINES
ASTSTOPA EQUATE X'80'        STOP ALL LINES
*  UTILIZE EXISTING LINE DEFINITION BYTES
         LSPACE
ASTACTDF EQUATE X'A0'        START OR STOP ALL LINES
*  UTILIZE LINE DEFINITION BYTES SUPPLIED IN PARAMETER LIST
         LSPACE
ASTAWRAP EQUATE X'01'        ADAPTER WRAP REQUEST, AMSNID REQUIRED
ASTMWRAP EQUATE X'02'        MODEM WRAP REQUEST, AMSNID REQUIRED
ASTAMWRA EQUATE X'03'        ADAPTER AND MODEM WRAP REQUEST
*                  AMSNID REQUIRED
         LSPACE
ASTDEF1  DEFLD DEFASTS,,1    LINE DEFINITION BYTE ONE
ASTDEF2  DEFLD DEFASTS,,1    LINE DEFINITION BYTE TWO
         LSPACE
```

The guidelines for defining the location of the AST fields in the appropriate segment are the same as explained under the DEFAMS operand description.

DEFSEN

When specified as the *copyfilename* operand on the COPY instruction, DEFSEN produces the following definition of the secondary link SENS parameter list:

```
      LSPACE
***************************************************************
            ALTERNATIVE LINE SENSE DEFINITION
***************************************************************
      LSPACE
SENSTATE  DEFLD DEFSENS,,1      DEVICE STATE BYTE
      LSPACE
SENONLI   EQUATE X'80'         DEVICE IS ONLINE
SENACTV   EQUATE X'40'         CONTROL UNIT IS ACTIVE
SENINPOL  EQUATE X'20'         DEVICE IS IN POLL LIST
SENINSEL  EQUATE X'10'         DEVICE IS IN SELECT LIST
SENOWNED  EQUATE X'08'         DEVICE IS OWNED
SENLSTAR  EQUATE X'04'         ALA LINE IS STARTED
SENONLPD  EQUATE X'02'         ONLINE STATUS IS PENDING
SENLCTPD  EQUATE X'01'         LOSS OF CONTACT IS PENDING
      LSPACE
SENRESV   DEFLD DEFSENS,,1      ADAPTER ADDRESS OF LINE
SENPOLL   DEFLD DEFSENS,,2      POLL ADDRESS
SENPOLL1  DEFLD DEFSENS,SENPOLL,1 POLL ADDRESS BYTE ONE
SENPOLL2  DEFLD DEFSENS,,1      POLL ADDRESS BYTE TWO
SENSEL    DEFLD DEFSENS,,2      SELECT ADDRESS
SENSEL1   DEFLD DEFSENS,SENSEL,1  SELECT ADDRESS BYTE ONE
SENSEL2   DEFLD DEFSENS,,1      SELECT ADDRESS BYTE TWO
SENNID    DEFLD DEFSENS,,2      NETWORK IDENTIFIER
SENNID1   DEFLD DEFSENS,SENNID,1  NETID BYTE ONE
SENNID2   DEFLD DEFSENS,,1      NETID BYTE TWO
SENTYPE   DEFLD DEFSENS,,1      DEVICE TYPE
SENSTID   DEFLD DEFSENS,,1      OWNING STATION ID
SENSTIDO  EQUATE X'00'         DEVICE IS ASSIGNED TO FREE POOL
```

The guidelines for defining the location of the SENS fields in the appropriate segment storage are the same as explained above under the DEFAMS operand description.

Appendix C. Program Check Codes

If the 4700 controller encounters an execution request that indicates a logic error, a program check results. The following are the hexadecimal codes and the explanations for possible program checks:

Code	**Explanation**
01	Invalid segment specification: An operand specifies a segment that was not defined during controller configuration procedure, or segment 14 was specified in an instruction that will cause data to be stored or changed in segment 14.
02	Segment overflow: Completion of the instruction requires more storage than the specified segment provides.
03	Field length error: An incorrect field was specified. The length is greater than 2 for an immediate operand; or a SETFPL instruction attempted to adjust the field length indicator to a negative value; or a value is specified which, when added to the PFP, would be greater than the segment length; or the field length was greater than 255 for a PAKSEG instruction.
04	Return-address stack error: An LRETURN instruction was issued, but the return-address stack was empty; or a branch instruction was issued, but the stack was full.
06	Instruction count threshold: The number of instruction executions allowed per transaction has been exceeded.
08	No overlay name: The overlay name is not in the resident overlay directory.
09	Invalid operation or segment code: The instruction operation or segment selection code specified is invalid. Make sure that any required OPTMOD coding for the instruction was entered and that any parameter fields are properly coded.
0A	No entry point: There is no startup entry point specified.
0B	Instruction address error: An addressing error has occurred. In the case of branch instructions, the program check address field of segment 1 will contain the address of the branch instruction.
0C	Instruction count exceeded: 65,535 instructions have been executed without a release of control.
0D	DEFDEL missing or incorrectly used: Either a delimiter request was made but no delimiter table was found or the table is not halfword aligned.
0E	EDIT mask error: The mask used with an EDIT instruction contains an error.
0F	Invalid link write control field: The link write control field or write options are invalid.
10	Communication link write length error: Data length exceeds 4095, data length during an LWRITE in batch mode was too long, command data length is incorrect; negative-response data length is incorrect, or there was a negative response to setting or testing sequence numbers.
11	Invalid parameter list, or parameter space is insufficient.
12	Indexing is not active.
20	Program check in called application program.

Code	Explanation
21	Called application program not found.
22	APCALL link stack full.
23	Recursive APCALL to an application program defined as USE=STATIC during configuration.
24	APCALL storage pool defined by MAXSTOR=was exceeded.
25	APCALL segment pool defined by MAXSEG=was exceeded.
26	APRETURN issued with no APCALL link stack entry - no calling application program.
27	Register address contains invalid segment space ID.
28	No transient pool: a transient pool was not defined for this station.
29	Transient application size error: the target transient application program will not fit in the largest transient area defined in the pool for this station.
FF	System error.

Appendix D. Link Status Codes

The list below and tables that follow contain information about the two bytes of status that are set in SMSDST when an exceptional condition occurs (condition code=X‘02’). The status bits in the first byte (SMSDS1) indicate the general condition:

Bits in SMSDS1:		**Condition:**
.......1	(X‘01’)	Incorrect length
......1.	(X‘02’)	Unit check
.....1..	(X‘04’)	Command reject
....1...	(X‘08’)	Attention
...1....	(X‘10’)	Prior operation
..1.....	(X‘20’)	Data check
.1......	(X‘40’)	Unit exception
1.......	(X‘80’)	Intervention required

The status bits in the second byte (together with those in the first) indicate the specific condition, as shown in the tables.

The first table (Figure D-1 on page D-1) applies to a SNA/SDLC communication link; the second (Figure D-2 on page D-11) to a BSC3 link. The list of SMSDS1 codes, above, is common to both tables. To use the list and tables, first find the status code in the appropriate table, and then the applicable description according to the instruction that caused the status. Remember that status can be posted for one instruction after a later instruction has been issued; refer to the appropriate chapter for more information on posting of status.

A status value not in the tables may be a combination of status codes. When such status values occur, review the appropriate table and search for the highest value first, then the next highest value. Remember that a status bit can be shared by more than one status code.

SNA/SDLC Status

The status codes in the following tables result when the 4700 assembler communication instructions described in Chapter 4, “The 4700 Assembler Communication Instructions,” the SNA/SDLC macros described in Chapter 5, “4700 SNA/SDLC Communication Macros,” or the BSC3 assembler instructions described in Chapter 7, “The 4700 BSC 3 Communication Instructions” set a non-zero condition code.

Status Bits/Code:	Condition and Action:	Instruction:
. . . . 1 . . . . . 1 . . (X‘0104’)	Incorrect length: The message did not fit into the buffer in the controller application program's segment. One of the following may have caused the condition: The message sent by the host application program may have been too large; the controller application program's buffer may have been too small. *Action:* Determine the cause of the condition and correct the host application program or the controller application program if necessary.	LREAD

Figure D-1 (Part 1 of 10). SNA/SDLC Link Status Codes

Status Bits/Code:	Condition and Action:	Instruction:
. . . . 1 . . . . . . 1 . . (X'0104') (CONTINUED)	Incorrect length: The message or SDLC link test data read was longer than the available space in the segment, the output data length is too large for the specified device, or the area specified to receive sense data was less than 10 bytes long. *Action:* Change the size of the area specified so that it is large enough, change the amount of data being written, or increase the size of the segment specified in the sense instruction.	LREADAL LWRITEAL LCNTRL
. . . . 1 . . . . . . 1 . 1 (X'0105')	Incorrect length: The available space in the segment was not large enough to contain all the statistical counters. Enough sense data was delivered to fill the specified area. *Action:* Increase the size of the area specified.	LCNTRL
. . . . 1 . . . . . 1 . . . (X'0108')	Incorrect length: The message read did not fit into the controller's internal buffer. This condition may have been caused by an error during the controller configuration or NCP generation. *Action:* Determine the cause of the condition and correct the controller configuration or the NCP generation. (Ignore the value in SMSIML.)	LREAD
	Incorrect length (Single buffer mode): The message read was longer than the specified size of the input buffer. (Dynamic buffer mode): Not enough buffers were available to read the message. *Action:* Change the size or increase the number of the related read buffers in the configuration.	LREAD LCNTRL
. . . . . . 1 . . . . (X'0200')	Unit check: A loss of contact caused the station to be placed in the not-ready state. Data transfer was unpredictable for an outstanding read or write operation. For a read, some data may have entered the controller application program's buffer. For a write, some of the data may not be entered into the network. The condition may have been caused by a problem in the communication network (for example, a line is down, or a VTAM deactivation occurred). *Action:* Operate offline until a ready command is received.	LREAD LWRITE LCHECK
	Unit check: Loss of contact on a line. The line is not currently operational, or the line for this network ID has changed from an operational to a nonoperational state. *Action:* Interrogate the statistical counters and system log to determine the cause. Correct the problem and restart the line, if required.	LREADAL LWRITEAL LCHECKAL

Figure D-1 (Part 2 of 10). SNA/SDLC Link Status Codes

Status Bits/Code:	Condition and Action:	Instruction:
. . . . . . 1 . . . . 1 (X'0201')	Unit check: A loss of contact caused the SLU to be placed in the Vary Online Pending state. Data transfer is unpredictable for an outstanding read or write operation. For a read, some data might have entered the PLU AP buffer. For a write, data may or may not have entered the network. The condition might have been caused by a problem on the line or in an SNA node. This status is reported only to stations that have received online status (X'0800'). *Action:* Do not use the SLU until Read status is received.	LREADAL LWRITEAL LCHECKAL
. . . . . . 1 . . . . 1 1 (X'0203')	Unit check: No response was received from the SDLC station for a Link Test message. *Action:* Repair the failing unit.	LCNTRL
. . . . . . 1 . . . . . . 1 . . (X'0204')	Unit check: The data sent in a SDLC Link Test message does not compare to the data received, or a response was received without data. *Action:* Send a shorter message or repair the failing unit.	LCNTRL
. . . . . 1 . . . . 1 (X'0401')	Command reject: Invalid parameter list (SMSIML identifies the entry being processed when the error was detected). *Action:* Correct the parameter list.	LCNTRL
. . . . . 1 . . . . 1 . (X'0402')	Command reject: The LCNTRL option was not specified or the required SNA-Primary optional modules were not specified during system initialization. *Action:* Correct the CPGEN, controller application program, or operator initialization procedure.	LWRITEAL LREADAL LCHECKAL LCNTRL
. . . . . 1 . . . . 1 1 (X'0403')	Command reject: Invalid network identifier or line number in AMSNID or parameter list, or the network entity specified is owned by another station, or the network identifier is 0 where 0 is not allowed, or the link network identifier was specified for other than a line function. *Action:* Correct the CPGEN or controller application program.	LWRITEAL LREADAL LCHECKAL LCNTRLAL
. . . . . 1 . . . . . . 1 . . (X'0404')	Command reject: Fast definite response required. The LWRITE completed without data transfer (only when options = PACE is specified on the COMLINK macro). *Action:* Send a fast definite response before issuing an LWRITE of data.	LWRITE

Figure D-1 (Part 3 of 10). SNA/SDLC Link Status Codes

Status Bits/Code:	Condition and Action:	Instruction:
. . . . . 1 . . . . 1 . . (X'0404') (CONTINUED)	Command reject: An LWRITE has been issued to a terminal identifier that was not included in the select list during CPGEN. *Action:* Correct the CPGEN or controller application program.	LWRITEAL
. . . . . 1 . . . . 1 . 1 (X'0405')	Command reject: Line not stopped. A wrap request was issued for a line that was currently active. *Action:* Stop the line and reissue the command.	LCNTRL
. . . . . 1 . . . . 1 1 . (X'0406')	Command reject: Control unit is in a non-SNA-Primary mode. A VONL or VOFF command was issued for a terminal associated with a control unit that had previously been placed in a non-SNA-Primary mode. *Action:* Correct the AMSNID field to permit the LCNTRL VONL or VOFF instruction to select a terminal rather than a control unit.	LCNTRL
. . . . . 1 . . . . 1 1 1 (X'0407')	Command reject: Control unit is in SNA-Primary message routing mode. A VONL command was issued for a control unit that had previously been placed in message routing mode. *Action:* Correct the controller application program by removing the LCNTRL VONL instruction referring to the control unit network ID.	LCNTRL LREADAL LWRITEAL LCHECKAL
. . . . . 1 . . . . 1 . . (X'0408')	Command reject: No asynchronous entry point for the central processor was defined in the controller application program by means of the BEGIN instruction. *Action:* Define an asynchronous entry for the central processor in the controller application program, if necessary.	LREAD LWRITE LCHECK
. . . . . 1 . . . . . . 1 . 1 . (X'040A')	Command reject: Network entity specified in AMSNID is offline. An operation was requested that requires the entity to be in an online state. *Action:* Correct the CPGEN or controller application.	LWRITEAL LREADAL LCHECKAL LCNTRL
. . . . . 1 . . . . . . 1 . 1 1 (X'040B')	Command reject: Network entity specified in AMSNID or parameter list is not offline. An operation was requested that requires that the entity be in an offline state. *Action:* Correct the CPGEN or controller application program.	LCNTRL

Figure D-1 (Part 4 of 10). SNA/SDLC Link Status Codes

Status Bits/Code:	Condition and Action:	Instruction:
. . . . . 1 . . . . . . 1 1 . 1 (X'040D')	Command reject: Maximum number of status entries pending. The station has attained the maximum number of write operations specified during CPGEN without issuing an LCHECK to clear the pending status. *Action:* Correct the CPGEN or controller application program.	LWRITEAL
. . . . . 1 . . . . . . 1 1 1 . (X'040E')	Command reject: Invalid instruction format. An instruction has an invalid format, probably because of a branch error. *Action:* Correct the controller application program.	LREADAL LWRITEAL LCHECKAL LCNTRL
. . . . . 1 . . . . . . 1 1 1 1 (X'040F')	Command reject: Invalid AMS specification. Segment 14 or an undefined segment has been indicated in SMSAMS1 as the location of AMS; or the displacement contained in SMSAMS2 is not divisible by two or would cause the AMS fields to extend beyond the designated segment. *Action:* Correct the application program or CPGEN to initialize SMSAMS1 and SMSAMS2 before executing any secondary link instructions.	LREADAL LWRITEAL LCHECKAL LCNTRL
. . . . . 1 . . . . . 1 . . . . (X'0410')	Command reject: The controller application program tried to write a nondata message to the central processor before reading the response to a previous nondata message. *Action:* Change the controller application program so that the response is read before the write is attempted.	LWRITE
. . . . . 1 . . . . 1 . . . . . (X'0420')	Command reject: The station was not allowed by the STATION configuration macro to use the communication link, or the station does not currently have an LU address assigned. *Action:* Reconfigure the station for link use, or assign an LU address to the station.	LREAD LWRITE LCHECK
. . . . . 1 . . . 1 . . . . . . (X'0440')	Command reject: While the station was in data flow reset state, the controller application program tried to write some type of data other than session control. *Action:* Change the controller application program so that only session control data is written during data flow reset state.	LWRITE
. . . . . 1 . . . . 1 (X'0441')	Command reject: The PLU tried to write a command to the SLU before reading to the response to the previous command. *Action:* Change the PLU AP so that the response is read before the write is attempted.	LWRITEAL

Figure D-1 (Part 5 of 10). SNA/SDLC Link Status Codes

Status Bits/Code:	Condition and Action:	Instruction:
. . . . . 1 . . . . 1 . (X'0442')	Command reject: The PLU has issued an LWRITE and does not currently have the required pre-requisite sessions establised or has requested a function that is not valid when the LU-LU session has already been established; for example, BIND. *Action:* Change the PLU AP to establish the required sessions before the write is attempted.	LWRITEAL
. . . . . 1 . . . . 1 1 (X'0443')	Command reject: The PLU tried to write to an SLU with an invalid write type in the AMSAWCT field. The write type is unknown or is not valid on the SSCP-PU flow. *Action:* Change the PLU AP to initialize the AMSAWCT to a correct value before issuing the write request.	LWRITEAL
. . . . . 1 . . . . 1 . . (X'0444')	Command reject: While the PLU was in data flow reset state, the PLU tried to write some type of data other than session control. *Action:* Change the PLU AP so that only session control data is written during data flow reset state.	LWRITEAL
. . . . . 1 . . . . 1 . 1 (X'0445')	Command reject: The PLU has requested a function that is not supported by either the TS or FM profiles which were set up at BIND time. *Action:* Issue another BIND command that will change the TS and/or FM profiles to support the function. Otherwise, correct the PLU AP so the invalid function is not used.	LWRITEAL
. . . . . 1 . . . . 1 1 . (X'0446')	Command reject: The PLU requested a specific function that requires a minimum amount of data and provided an incorrect amount. *Action:* Change the PLU AP to provide the required amount of data.	LWRITEAL
. . . . . 1 . . . . 1 1 1 (X'0447')	Command reject: The PLU has issued a BIND, and the TS or FM profile parameters are invalid. *Action:* Change the PLU AP to provide only the supported TS and FM profile parameters.	LWRITEAL
. . . . . 1 . . . . . . 1 . . . (X'0448')	Command reject: While the PLU was in quiesce state, it tried to write some type of data other than session control or asynchronous data flow control. *Action:* Change the PLU application program so that only session control or asynchronous data flow control data is written during quiesce state.	LWRITEAL

Figure D-1 (Part 6 of 10). SNA/SDLC Link Status Codes

Status Bits/Code:	**Condition and Action:**	**Instruction:**
. . . . . 1 . . 1 . . . . (X'0480')	Command reject: While the station was in quiesce state, the controller application program tried to write some type of data other than session control or asynchronous data flow control. *Action:* Change the controller application program so that only session control or asynchronous data flow control data is written during quiesce state.	LWRITE
. . . . 1 . . . . (X'0800')	Attention: The operator signaled attention by pressing the reset key twice in succession. The operation was in a wait state with an indeterminate end point (an attention does not affect a wait state with a determinate end point). The wait state may have resulted from such conditions as: a read from 3614 terminal or the central processor; intervention required for a printer; failure to encode a magnetic stripe after the magnetic stripe encoder was enabled. *Action:* Prompt the operator to carry out the appropriate action (such as replacing the forms on a printer). Reset the magnetic stripe encoder if it was enabled.	LWRITE
	Attention: Online reported. If a Vary Online command is issued for a secondary link device that is not ready, online status is reported on a subsequent secondary link command. The online status reported can be either from the device or control unit. Online status reported as a result of a Vary Online request for a terminal always refers to that terminal's control unit. It may or may not indicate establishment of contact with the terminal. To find the status of the terminal, issue an LWRITE command and test the returned status.	LREADAL LWRITEAL LCHECKAL LCNTRL
. . . 1 . . . . (X'1000')	Prior operation: The status presented refers to a previous LWRITE operation, and is presented with the ORed status of previous LWRITE operations. AMSAFN holds the event number for the write operation when it began. *Action:* If an LWRITEAL instruction fails because a previous instruction did not end correctly, the current LWRITEAL instruction returns "prior operation" status. The AP should issue LCHECKAL to determine if prior instructions are successful. The event number for an outstanding LWRITEAL instruction is in AMSAFN; the status applies to that instruction.	LREADAL LWRITEAL LCHECKAL
. . 1 . . . . (X'2000')	Data check: A network error was detected, and an error response was generated (if appropriate). The message was lost, if possible, sense data was sent to the originator of the message. The error was recorded in the system log. The condition was probably caused by an error in the communication network. *Action:* Send a request recovery message or terminate the session.	LREAD

Figure D-1 (Part 7 of 10). SNA/SDLC Link Status Codes

Status Bits/Code:	**Condition and Action:**	**Instruction:**
. . 1. . . . . 1 (X'2001')	Data check: The system detected a network error and, if appropriate, issued an error response. The message was lost. If possible, sense data was sent to the one who sent the message. The error, which was probably caused by an SLU error, was sent to the system log. *Action:* End the session and correct the problem.	LREAD LWRITE LCHECK
. 1. . . . . (X'4000')	Unit exception: No data or status is pending from the device specified in AMSNID, or no data or status is pending for the station if a general read was performed. If you specify a NID = X'FFFF', no data or status for the secondary device is pending. *Action:* If SMSAAP is on, issue a general read (AMSNID = X'0000' or X'FFFF') to read or locate the status or input data.	LREAD
. 1. . . . . 1 (X'4001')	Unit exception: Vary Online pending. Online when contact with the device is established. *Action:* Either issue an LEXIT instruction (if an ALA asynchronous entry point was defined) or test SMSAAP periodically for Online status. If you issue LEXIT, the station is dispatched at its asynchronous ALA entry point when Online status is available.	LREADAL LWRITEAL LCHECKAL LCNTRL
. 1. . . . . 1 . (X'4002')	Unit exception: Vary Online pending, line not started, line start not previously requested. Online is reported when the line is started. *Action* (See Note): Issue a STRL instruction for the unit's line, followed by an LEXIT instruction (if an ALA asynchronous entry point was defined) or a periodic test of SMSAAP to determine when Online status is available. If you issue an LEXIT, the station is dispatched at its asynchronous ALA entry point when Online Status is available.	LREADAL LWRITEAL LCHECKAL LCNTRL
. 1. . . . . 1 1 (X'4003')	Unit exception: Vary Online pending. "Line state" has been requested but the line did not start due to equipment or modem malfunction or a missing optional module. Online status occurs if the line starts. *Action:* Either correct the problem and reissue STRL, or correct the operator initialization procedure. If you reissue STRL and an ALA asynchronous entry point was defined, you should also either issue LEXIT or make a periodic test of SMSAAP for online status. If you issue LEXIT, the controller dispatches the station at the ALA= entry point when online status becomes available.	LREADAL LWRITEAL LCHECKAL LCNTRL

Figure D-1 (Part 8 of 10). SNA/SDLC Link Status Codes

Status Bits/Code:	**Condition and Action:**	**Instruction:**
. 1 . . . . 1 . . (X'4004')	Unit exception: Vary Online pending, terminal and/or control unit not in active poll list. Online status is reported when the polling list is changed. *Action:* Correct the CPGEN or application program.	LREADAL LWRITEAL LCHECKAL LCNTRL
. 1 . . . . 1 . . . . (X'4010')	Unit exception: An end bracket was read; data transfer took place. A begin-bracket, end-bracket, first of chain, or last-of-chain indicator was read. These conditions can occur individually or in the following combinations: begin bracket and first of chain, begin bracket and last of chain, end bracket and first of chain, or end bracket and last of chain.	LREAD
. 1 . . . . 1 . . . . . (X'4020')	Unit exception: A begin bracket was read; data transfer took place (LREADAL). A begin-bracket, end-bracket, first-of-chain, or last-of-chain indicator was read. These conditions can occur individually or in the following combinations: begin bracket and first of chain, begin bracket and last of chain, end bracket and first of chain, or end bracket and last of chain. *Action:* Defined by the controller application program.	LREAD
	The station was requested to read a nondata message that is waiting to be read; the message, response, or data was sent to the station either before or during the write operation; or OPTIONS = PACE but no pace count. *Action:* Read the pending message, response, or data or complete processing and exit. (Otherwise, the controller's internal buffers may become full and messages may stop flowing into the controller.)	LREADCP
. 1 . . . . 1 . . . . . . (X'4040')	Unit exception: Last of chain was read; data transfer took place (LREADAL). A begin-bracket, end-bracket, first-of-chain, or last-of-chain indicator was read. These conditions can occur individually or in the following combinations: begin bracket and first of chain, begin bracket and last of chain; end bracket and first of chain, or end bracket and last of chain. *Action:* Defined by the controller application program.	LREAD
	The station was requested to read a response that is waiting to be read. The message, response, or data was sent to the station either before or during the write operation. *Action:* Read the pending message, response, or data or complete processing and exit. (Otherwise, the controller's internal buffers may become full and messages may stop flowing into the controller.)	LWRITE

Figure D-1 (Part 9 of 10). SNA/SDLC Link Status Codes

Note: Prior Operation status is returned when a read, write, or a check request for failing writes shows that a previously requested operation has not completed. To ensure that all outstanding write operations have been completed successfully, the application program (AP) must issue an LCHECK instruction. When status is returned, the event number of any outstanding write operation is found in AMSAFN. The status shown in SMSDST applies to that write operation.

Status Bits/Code:	Condition and Action:	Instructions:
. 1 . . . . 1 . . . . . . (X'4080)	Unit exception: First of chain was read; data transfer took place (LREADAL). A begin-bracket, end-bracket, first-of-chain, or last-of-chain indicator was read. These conditions can occur individually or in the following combinations: begin bracket and first of chain, begin bracket and last of chain, end bracket and first of chain, or end bracket and last of chain. *Action:* Defined by the controller application program.	LREAD
	Unit exception: The station was requested to read data that is waiting to be read. The message, response, or data was sent to the station either before or during the write operation. *Action:* Read the pending message, response, or data or complete processing and exit. (Otherwise, the controller's internal buffers may become full and messages may stop flowing into the controller.)	LWRITE
. 1 . . 1 . . . . . 1 . . . . . (X'4820')	Unit exception, attention: OPTIONS = PACE was specified and a nondata message is pending from the host; no data is transferred. *Action:* Read the pending message or complete processing and exit. (Otherwise, the controller's internal buffers may become full, and messages may stop flowing into the controller.)	LWRITE
. 1 . . 1 . . . . 1 . . . . . . (X'4840')	Unit exception, attention: OPTIONS = PACE was specified and data is pending from the host; no data is transferred. *Action:* Read the pending response or complete processing and exit. (Otherwise, the controller's internal buffers may become full, and messages may stop flowing into the controller.)	LWRITE
. 1 . . 1 . . . . . 1 . . . . . (X'4880')	Unit exception, attention: OPTIONS = PACE was specified and data is pending from the host; no data is transferred. *Action:* Read the pending data or complete processing and exit. (Otherwise, the controller's internal buffers may become full, and messages may stop flowing into the controller.)	LWRITE
1 . . . . 1 . . . . . (X'8020')	Intervention required: The controller application program was not permitted to write a Network Services (NS) command (SSCP-PU session command) because the SSCP-PU session was not yet established. *Action:* Revise application program.	LWRITE
1 . . . . 1 . . . . . . (X'8040')	Intervention required: The work station was not in session because it had not responded positively to a Bind message. *Action:* Wait for a Bind message and respond positively.	LWRITE
1 . . . . 1 . . . . (X'8080')	Intervention required: The controller application program was not allowed to initiate or terminate a session because the station had not read a Ready message or the part of VTAM that controls initiation and termination was lost. *Action:* Wait for a Ready message and initiate or terminate a session.	LWRITE

Figure D-1 (Part 10 of 10). SNA/SDLC Link Status Codes

BSC3 Status Codes

The following status codes will be set when a condition code of 2 is indicated.

Status Code: X'0104'

Condition: Incorrect length. The application program buffer was too small for the message. As much data as possible is placed in the buffer and the rest discarded.

Instruction: LREAD

Explanation: The message did not fit in the buffer in the controller application program's segment. The message sent by the host application program was too large, or the controller application program's buffer was too small.

Action Required: Determine the cause of the condition and correct the host application program, or the controller application program as necessary.

Status Code: X'0108'

Condition: Incorrect length. The controller's buffer was too small. As much data as possible was placed in the application program's buffer and an incorrect ACK was returned to the host.

Instruction: LREAD

Explanation: The message read did not fit into the controller's internal buffer. This condition may have been caused by an error during controller configuration.

Action Required: Determine the cause of the condition and correct the controller configuration. (Ignore the value in SMSIML).

Status Code: X'0200'

Condition: Loss of contact. An outage was detected; no data transfer took place.

Instructions: LREAD, LWRITE, LCHECK

Explanation: A loss of contact caused the station to be placed in the not-ready state. Data transfer was unpredictable for a write or check operation. Some data may have entered the controller application program's buffer for a read operation. The condition may have been caused by a problem in the communication line.

Action Required: Re-establish contact.

Status Code: X'0401'

Condition: Command reject. The batch mode option was being used, and the write control field was not X'10'. No data transfer took place.

Instructions: LWRITE

Explanation: The write control field (SMSBWC) was something other than X'10', and the controller was being run in batch mode.

Action Required: Either correct the write control field or restart the link in single-message mode.

Status Code: X'0408'

Condition: The command was rejected. The program issuing the command had no asynchronous central processor entry point. No data was transferred.

Instructions: LREAD, LWRITE, LCHECK

Explanation: No asynchronous entry point for the central processor was defined in the controller application program by means of the BEGIN instruction.

Action Required: Define an asynchronous entry point for the central processor in the controller program.

Figure D-2 (Part 1 of 2). BSC3 Link Status Codes

Status Code: X'0420'

Condition: The command was rejected. The station was not specified to use the link. No data was transferred.

Instructions: LREAD, LWRITE, LCHECK

Explanation: The station was not specified as a station that can use the communication link.

Action Required: If necessary, change the controller configuration so that the station can use the link.

Status Code: X'0800'

Condition: Attention. The operator signaled attention. No data was transferred.

Instruction: LREAD

Explanation: The operator signaled attention by pressing the reset key twice. The operation was in a wait state with an indeterminate end point.

Action Required: Prompt the operator for additional information.

Status Code: X'2001'

Condition: Bad data.

Instruction: LWRITE, LCHECK

Explanation: A control character was included in the data for a nontransparent transmission. For an LWRITE operation, if the status refers to the previous operation, prior operation is set in the status (X'1201').

Action Required: Either correct the application so that control characters do not appear in the data or send the data in transparent mode.

Status Code: X'4001'

Condition: The link is up. The link is running and contact is established with the host.

Instruction: LREAD

Explanation: The controller has established contact with the host.

Action Required: The logical work station may now transmit data to the host.

Status Code: X'4080'

Condition: Unit exception: there is data pending for this logical work station.

Instruction: LWRITE

Explanation: A message was received at the controller for this logical work station either before or during the write operation.

Action Required: Read the pending message to clear the controller's buffer.

Status Code: X'8040'

Condition: Intervention is required. The link was not running. No data was transferred.

Instruction: LWRITE, LREAD, LCHECK

Explanation: The link was not running.

Action Required: Start the link.

Status Code: X'8080'

Condition: Contact has not been established with the host.

Instruction: LWRITE

Explanation: The link has been started, but contact has not been established with the host.

Action Required: Wait for READY status and then reissue the instruction.

Figure D-2 (Part 2 of 2). BSC3 Link Status Codes

Appendix E. Communication Link Statistical Counters

This appendix lists brief descriptions of the communication statistical counters. These counters record certain preselected events that occur on the host and alternate/secondary communication links.

The events recorded in the counters include, but are not only for, errors. For example, host link counter 1 records each Set Normal Response Mode (SNRM) and Set Disconnect Response Mode (SDRM) command. For complete descriptions of the counters, how to read them with the system monitor, and their meanings refer to the *4700 Subsystem Operating Procedures*.

SNA/SDLC Host Link Counters

Counter:	**Meaning:**
1	Beginning/ending (SNRM/SDRM) sequences (information)
2	Host-issued test message received (information)
3	Write retry by controller (recoverable error)
4	Timeout error by controller (unrecoverable error)
5	Overrun error at controller (recoverable error)
6	Underrun error at controller (recoverable error)
7	Connection problem (unrecoverable error)
8	Invalid adapter status (unrecoverable error)
9	Frame check sequence (FCS/CRC) error (recoverable error)
10	Abnormal ending (recoverable error)
11	Data communications equipment error (recoverable error)
12	Busy/no buffers available (information)
13	Command reject (protocol error)
14	Machine check (unrecoverable error)
15	Invalid SDLC data (protocol error)
16	Invalid SDLC command (recoverable error)
17	Multi-use loop/WTC line loss (unrecoverable error)
18	Clear-to-send (CTS) loss on multi-use loop/WTC (unrecoverable error)

BSC3 Host Link Counters

Counter:	**Meaning:**
1	Poll counter (information)
2	Test request counter (information)
3	Write retry NAK counter (recoverable error)
4	Timeout (unrecoverable error)
5	Overrun (recoverable error)
6	Underrun (recoverable error)
7	Retry error (unrecoverable error)
8	Invalid adapter status (unrecoverable error)
9	Block check test failure (recoverable error)
10	Host-requested abnormal ending (information)
11	Data communications equipment (DCE) error (unrecoverable error)
12	Busy/out-of-buffers (information)
13	Protocol sequence error (recoverable error)
14	Communications adapter error (unrecoverable error)
15	Controller select count (information)

ALA Link Counters

The ALA link has eight line counters, and 16 terminal and control unit counters, as described in this section. For detailed descriptions of the counter meanings, system actions in response to counter changes, and recommended actions that you should take, refer to the *4700 Subsystem Operating Guide*.

ALA Line Counters

Counter:	**Meaning:**
1	Overrun (recoverable error)
2	Underrun (recoverable error)
3	Secondary link adapter error (unrecoverable error)
4	Modem error (unrecoverable error)
5	Busy/Out-of-buffers (recoverable error)
6	Protocol error (unrecoverable error)
7	(reserved)
8	(reserved)

ALA Terminal/Control Unit Counters

Counter:	**Meaning:**
1	SDLC sequence error (unrecoverable error)
2	(reserved)
3	No response received (recoverable error)
4	(reserved)
5	Non-productive timeout (recoverable error)
6	Poll timeout (recoverable error)
7	FCS (frame check sequence) error (recoverable error)
8	(reserved)
9	(reserved)
10	Data count exceeded (unrecoverable error)
11	SNA protocol error (unrecoverable error)
12	Invalid command received (unrecoverable error)
13	ROL (request online) received (loss of contact posted) (unrecoverable error)
14	SDLC I-frame overflow (recoverable error)
15	(reserved)
16	(reserved)

Glossary

This glossary defines terms used in this manual. Some terms are unique to the 4700 system, and others are repeated for convenience because they apply to the 4700 system.

If you cannot find a term's definition here, refer either to the index or to the *IBM Vocabulary for Data Processing, Telecommunications, and Office Systems*, GC20-1699. This glossary includes definitions from that publication.

ALA link. The SNA/SDLC telecommunication line between the 4701 controller and a secondary logical unit (SLU) control unit or device.

alphameric characters. In 4700 assembler language, the characters A/a through Z/z, the digits 0 through 9, and the characters #,$,@.

application program. (1) In 4700, a program written for or by a user that processes a transaction or performs some other financially–related work. (2) In communications, a program used to connect and communicate with stations in a network, allowing users to perform application– oriented activities.

assembler. (1) A computer program that converts symbolic instructions into machine instructions. (2) In a 4700 system, a VS version of an assembler used to: convert a program written in assembler language into machine instructions that can be executed in the controller; to convert 4700 configuration macro instructions into configuration data; and to convert customization macro instructions into customization data.

asynchronous. Without regular time relationship; unexpected or unpredictable with respect to the execution of a program or its instructions.

asynchronous entry point. In an application program, the address to which control is transferred when data is pending for an idle work station. See also *host entry point, station entry point, and terminal/device entry point.*

block. (1) The smallest complete unit of data that can be transmitted between units in a communication network. The maximum size of a block depends on the characteristics of the sending or receiving device. (2) a group of contiguous characters recorded as a unit. (3) On a controller diskette, the subdivision of a track. Depending on the diskette type, each track contains a fixed number of blocks. One record can occupy one or more blocks, or smaller records can be packed, or "blocked", to fit in diskette blocks.

buffer. In the 4700 system, a storage area that is reserved for use by data transmission operations.

communication link. (1) In general, the physical means of connecting one location to another for the purpose of transmitting and receiving information. (2) In the 4700 system, the communication link consists of an external at each location and a telephone line that connects the locations.

communications network management/controller support (CNM/CS) facility. A facility of the expanded system monitor that permits a terminal operator at the host to control the 4700 subsystem, solicit performance data for analysis, and communicate with an operator at the controller. CNM/CS operates through a network management facility such as the Network Communications Control Facility (NCCF) and is compatible with performance analysis functions such as the Network Problem Determination Application (NPDA).

configuration. In the 4700 system, the group of terminals and controller storage areas and application programs that constitute a subsystem associated with a controller.

configuration image. A combination of formatted configuration data with selected modules of controller data that, when loaded into the controller storage, determines the operations of the controller.

controller log. In a controller, a temporary file on the diskette in which controller log messages are recorded and in which user data can also be recorded.

control operator. In a communication system, the person who performs special administrative, control, and testing functions.

cursor. (1) (SC1) In computer graphics, movable, visible mark used to indicate a position on a display space. (2) A movable spot of light on the screen of a display device, usually indicating where the next character will will be entered.

CPU. An abbreviation of "central processor unit", a deprecated term, that appears in macro parameters to refer to the host system.

data communication equipment (DCE). The common carriers lines, devices, and facilities that interconnect data terminal equipment.

data terminal equipment (DTE). (1)* A data source, a data sink, or both. (2) (SC1) The functional unit of a data station that serves as a data source or a data sink and provides for the data communication control function to be performed in accordance with a link protocol.

debug*. (ISO) To detect to trace, and to eliminate mistakes in computer programs or in other software. Synonymous with checkout.

digit. One of the numeric characters 0 through 9.

diskette. A thin, flexible magnetic disk and a semi-rigid protective jacket, in which the disk is permanently enclosed. See daily initialization diskette, diagnostic diskette, formatted diskette, installation diskette, unformatted diskette. Synonymous with flexible disk.

display. (1) a component that provides visual communication between the user and the controller. (2) a visual presentation of data.

dump. With reference to the controller, to copy a part of storage onto a diskette.

function key. (1) (SC1) In computer graphics, a button or switch that may be operated to send a signal to the computer program controlling the display. (2) A key on a terminal, such as the attention key, that causes the transmission of a signal not associated with a printable character. Detection of the signal usually causes the system to perform some predefined function for the user.

global storage. In a controller, programmable storage that is available to all work stations. Contrast with *private storage* and *shared storage*.

hexadecimal. A number system with a base of 16.

host or host system. (1) The primary or controlling computer in a multiple computer operation. (2) a computer used to prepare programs for use on another computer or on another data processing system; for example, a computer used to compile, link edit, or test programs to be used on another system. (3) The primary or controlling computer in a data communication system.

ID. Identification.

ID card. A card, similar in size to a credit card, that contains the users identification written on a magnetic stripe.

ID keys. Specially designed keys on shared terminals that identify the user to the controller.

inquiry. A request for information from storage.

installation diskette. A diskette used in a controller mainly to initiate communication with the host computer and to prepare the controller for reception and recording of the configuration image. Contrast with *operating diskette*.

institution. Any financial establishment, such as a commercial bank, mutual savings bank, savings and loan association, credit unit, and finance company.

LCF. Local Configuration Facility.

local control operator. A control operator at a local location. Contrast with *remote control operator*.

local location. A location that has a controller. Contrast with *remote location*.

local loop. A channel connecting the subscribers equipment to the line–terminating equipment in the central office exchange.

log. See *controller log*.

logical device address (LDA). A number used to designate a particular terminal component within a work station. Configuration data in the controller correlates the logical device address with the actual physical address. See *physical address*.

logoff. The steps by which a user signs off from the system.

logon. The steps by which a user signs on to the system.

loop. See *local loop* or *remote loop*.

loop number. In the 4700 system, a number that identifies a particular loop in a controller. See *physical address*.

magnetic stripe. The stripe on certain credit cards, ID cards, and passbooks on which data can be recorded magnetically.

magnetic stripe encoder/reader. A component available for the 4704/3604 that reads precoded information from, and writes coded information on, a magnetic stripe on a passbook, credit card, or ID card.

magnetic stripe reader. A component available for the 4704/3604 that reads precoded information from a magnetic stripe on a passbook, credit card, or ID card.

modem. (1) (SC1) A functional unit that modulates and demodulates signals. One of the functions of a modem is to enable digital data to be transmitted over analog transmissions facilities. (2)* (modulator-demodulator) A device that modulates and demodulates signals transmitted over data communication facilities. (3) See also data set (2), line adapter, modulation.

modem unit. A terminal (3604–2, –3, or –4 and any 3614 model) that has a modem or can be attached to an external modem.

numeric character. Same as digit.

operating diskette. A diskette used in a controller that contains the configuration image, and other data, relating to the operation of the controller. The operating diskette must be in the controller during its operations. A second diskette containing the same configuration image and data is referred to as a *backup operating diskette*. Contrast with *installation diskette*.

parameter. A variable that is given a fixed value for a specific application. See *parameter data byte* and *parameter flag byte*.

permanent file. In the 4700 system, a file on a diskette that can be used to store data to be retained from one controller startup to another. The permanent data might include such things as day–to–day totals or checkpoint/restart data. Contrast with *temporary file*.

physical address. In the 4700 system, an address that is used to reach a particular terminal or component. A physical address consists of a loop number, a terminal address, and a component address. In the configuration image in a controller, each physical address is correlated with a number (called a logical device address) that is used to identify a component in a work station. See *logical device address*.

private storage. In the controller, programmable storage that is associated with only one work station. Contrast with *global storage* and *shared storage*.

programmable storage. The portion of internal storage in the controller in which user–written programs are executed.

prompt. To help a terminal user by displaying messages that request information necessary to continue an operation.

record. Pertains to the classification of data stored on a diskette.

remote control operator. A control operator at a remote branch. Contrast with *local control operator*.

remote location. A location that is connected to the controller by a communication link. Contrast with *local location*.

remote loop. In the 4700 system, a closed circuit of telephones lines (not local cables) that starts at a controller and attaches remote locations one to another and back to the controller. Messages from the controller travel around the loop in one direction. Contrast with *local loop*.

remote subloop. In a remote loop, the closed circuit of cables that attach the remote terminals to each other at a remote location. See *remote loop*.

segment. In a controller, one of 16 portions into which the programmable storage related to a controller application program can be divided. The length of each segment is specified by the user.

shared storage. In the controller, programmable storage that is reserved for the application program and which may be shared between work stations. Contrast with *global storage* and *private storage*.

special character. Refers to any character other than the 26 letters and the 10 digits; for example, punctuation marks.

SR. service representative.

step. A fractional part of a print line on a passbook. There are 12 steps to a line.

storage. A part of the controller or host computer in which the program or data is kept.

subloop. See *remote loop*.

system monitor. The facility in a controller that handles communications with the control operator.

temporary file. In the 4700 system, a file on a diskette that can be used to store data that is not to be retained from one controller startup to another. This temporary data might include such things as a daily audit trail or a tellers cash position. Contract with *permanent file*.

transaction. (1) In the 4700 system, generally, an exchange between a terminal and another unit to effect a particular action or result. (2) More specifically, a single communication action involving an inquiry from a terminal that produces a response containing desired information (such as a request from a terminal for a customers account balance) or a more complex action in which data records must be changed (such as a request to update a customers balance with a new deposit).

work station. In the 4700 system, a collection of terminals and storage that is used by an application program executed by a controller to process transactions.

wrap test. At a local location, a test performed by the controller that checks the controller and its modem that connects to the remote loop. At a 1200–bps remote location or a location with an external modem, a test that checks the remote subloop and its modem. (**Note:** *Some external modem models cannot be wrap–tested; the wrap test is then valid for only the controller or terminal.*)

Index

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

4700 Finance Communication System
Controller Programming Library
Volume 3
Communication Programming
Order No. GC31-2068-1

READER'S COMMENT FORM

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. Comments may be written in your own language; English is not required.

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.*
Possible topics for comment are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name, company, mailing address, and date:

____

____

____

____

What is your occupation?____

Number of latest Newsletter associated with this publication:____________________

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.)

GC31-2068-1

Reader's Comment Form

Fold and Tape Please Do Not Staple Fold and Tape

NO
POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED
STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department 78C
1001 W.T. Harris Boulevard
Charlotte, NC, USA 28257

Fold and Tape Please Do Not Staple Fold and Tape

4700 Finance Communication System Controller Programming Library
Volume 3 Communication Programming (File No. S370/4300/8100/S34-30) Printed in U.S.A. GC31-2068-1

4700 Finance Communication System
Controller Programming Library
Volume 3
Communication Programming
Order No. GC31-2068-1

READER'S COMMENT FORM

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. Comments may be written in your own language; English is not required.

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.* Possible topics for comment are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name, company, mailing address, and date:

____

____

____

____

What is your occupation?______________________________

Number of latest Newsletter associated with this publication:____________________

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.

GC31-2068-1

Reader's Comment Form

Fold and Tape   **Please Do Not Staple**   Fold and Tape

NO
POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED
STATES

BUSINESS REPLY MAIL

FIRST CLASS   PERMIT NO. 40   ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department 78C
1001 W.T. Harris Boulevard
Charlotte, NC, USA 28257

Fold and Tape   **Please Do Not Staple**   Fold and Tape

4700 Finance Communication System Controller Programming Library
Volume 3 Communication Programming (File No. S370/4300/8100/S34-30) Printed in U.S.A. GC31-2068-1

Publication Number
GC31-2068-1

File Number
S370/4300/8100/S34-30

Printed in
USA