INTERCOMM

SNA LU6.2 SUPPORT GUIDE

ISOGON CORPORATION
330 Seventh Avenue · New York, New York 10001

LICENSE: INTERCOMM TELEPROCESSING MONITOR

SNA LU6.2 Support Guide

Publishing History

Publication	Date	Remarks
First Edition	September 1986	Documenting initial CICS to Intercomm support under Intercomm Release 9.0.
Second Edition	February 1990	Updates and corrections corresponding to Intercomm Releases 9 and 10.
Third Edition	February 1993	Documenting the new extended support allowing Intercomm pass through, and back end subsystem invocation, of transactions on an ISC link for Release 10 only.

NOTE: current users of (basic) LU6.2 support should continue to also refer to the Second Edition of this manual (particularly Appendix B) until executing under Release 10 with all relevant SMs and/or EXs applied.

PREFACE

Intercomm is a state-of-the-art teleprocessing monitor system executing on the IBM System/370 and System/390 family of computers and operating under the control of IBM Operating Systems (MVS/370, XA and ESA). Intercomm monitors the transmission of messages to and from terminals, concurrent message processing, centralized access to I/O files, and the routine utility operations of editing input messages and formatting output messages, as required.

This manual describes Intercomm support of IBM System Network Architecture (SNA) Logical Unit Type 6.2, which is a means of having VTAM application programs in the same or different nodes of a network communicate with each other. This feature is external to Intercomm, that is, it is not available with the bundled system, but must be purchased separately and is available in two modes:

- The passive mode (the original basic support) allows Intercomm to accept input from other applications (CICS) and process the transactions, returning the responses to the originator.

- The active mode is the new extended support and includes the passive mode; active mode allows Intercomm to initiate transactions in another application over an LU6.2 link as well as receive them, that is, both front end pass through of entered transactions and the initiation of transactions over an LU6.2 link by Intercomm subsystems. The other application on a specific LU6.2 link may be another Intercomm, CICS, or any system which supports the LU6.2 protocol (APPC).

This feature requires the preinstallation of standard Intercomm SNA (VTAM) support and does not include that special feature if it is not already owned by a site wishing this extension to SNA support.

The following Intercomm publications are prerequisite to this manual:

- Operating Reference Manual

- Basic System Macros

- SNA Terminal Support Guide

It is assumed throughout this manual that the reader is familiar with the appropiate IBM reference manuals describing the LU6.2 protocol and APPC (Advanced Program-to-Program Communication).

INTERCOMM PUBLICATIONS

GENERAL INFORMATION MANUALS

- Concepts and Facilities
- Planning Guide

APPLICATION PROGRAMMERS MANUALS

- Assembler Language Programmers Guide
- COBOL Programmers Guide
- PL/1 Programmers Guide

SYSTEM PROGRAMMERS MANUALS

- Basic System Macros
- BTAM Terminal Support Guide
- Installation Guide
- Messages and Codes
- Operating Reference Manual
- System Control Commands

CUSTOMER INFORMATION MANUALS

- Customer Education Course Catalog
- Technical Information Bulletins
- User Contributed Program Description

FEATURE IMPLEMENTATION MANUALS

- Autogen Facility
- ASMF Users Guide
- DBMS Users Guide
- Data Entry Installation Guide
- Data Entry Terminal Operators Guide
- Dynamic Data Queuing Facility
- Dynamic File Allocation
- Extended Security System
- File Recovery Users Guide
- Generalized Front End Facility
- Message Mapping Utilities
- Model System Generator
- Multiregion Support Facility
- Page Facility
- Store/Fetch Facility
- SNA Terminal Support Guide
- TCAM Support Users Guide
- Utilities Users Guide

EXTERNAL FEATURES MANUALS

- SNA LU6.2 Support Guide

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Chapter 1

INTRODUCTION

The IBM SNA LU6.2 support is designed to provide a standardized means of having VTAM application programs (Intercomm, CICS, etc.) in the same or different nodes of an SNA network communicate with each other. The VTAM applications may execute in the same or different CPU's connected on the same network. Communication is via LU6.2 sessions created by one VTAM application to "talk" to another VTAM application which can accept and respond to data transmitted via the session, rather than a terminal. LU6.2 data transfer concepts are a design detail only, and until APPC/MVS, each VTAM application that wishes to communicate with another VTAM application must provide an implementation of this design. CICS, for example, has implemented the LU6.2 design for user transaction processing programs through the EXEC CICS interface, wherein the CICS program can pass a transaction via an LU6.2 session to another application. Internal VTAM interface modules in CICS issue standard VTAM macros to establish LU6.2 sessions and then to pass transactions on them as a result of the EXEC CICS commands. VTAM V3 provides a universal macro (APPCCMD) and APPC/MVS, which is available only under MVS/ESA V4, provides 'callable services' to replace the standard VTAM macros for LU6.2 communication. Neither Intercomm nor CICS (currently) use the APPCCMD macro or the high-level APPC services, instead, standard VTAM macros are being used so that overlap of processing may be achieved. Nevertheless, both systems may be referred to as APPC systems since they support LU6.2 transaction processing.

In Intercomm basic LU6.2 support, or passive mode, Intercomm subsystems can receive passed transactions (from another APPC system) in the same form as standard Intercomm terminal input and respond to them as though the output is returned to an Intercomm terminal. That is, the origin of the transaction is transparent to the Intercomm subsystem. In the basic LU6.2 support, internal code has been provided to accept external session creation requests, control these sessions, and allow back-end Intercomm subsystems to be invoked by input on these sessions. The output created by an Intercomm subsystem invocation will be directed back on the same session under strict control. Some of this control implies restrictions on generally available Intercomm facilities such as verb locking and use of the CONVERSE facility. A summary list of known restrictions is given at the end of this manual in Appendix C. The restrictions are due to the random allocation of a session in the sending system to pass each input message, which has no correspondance to the originating terminal. (See the description of LUTYPE6.2 mapped conversations in IBM's CICS Intercommunication Guide for the CICS implementation.)

For CICS to communicate with Intercomm, a CICS application program will be required to invoke "conversations" on the LU6.2 sessions upon receipt of terminal input. Therefore, the purpose of the initial design was to allow terminals logged on to CICS to be able to invoke Intercomm transactions and have the response created under Intercomm returned to them. Now, Intercomm can invoke conversations on an LU6.2 link with extended LU6.2 support installed.

In the extended (active mode) Intercomm LU6.2 support for Release 10, transactions can be defined to be processed in another system by specifying the identity of that system using the APPLID parameter of the BTVERB macro defining the transaction in the BTVRBTB transaction-id table. The internal session creation requests and control of these sessions will include the division on an LU6.2 link between transaction receiver sessions and transaction sender sessions (known as Secondary and Primary sessions respectively). The transfer of an entered transaction to another system and return of the response is transparent to the terminal operator (unless an LU6.2 error condition occurs). Further, Intercomm user application subsystems can invoke transactions in another system via an INITLU6 facility. The response from the LU6.2 invocation can be passed back to the subsystem via INITLU6, or can be passed back to the originating terminal, based on subsystem request.

In active mode, an Intercomm system can be used as a Front End for other systems (other Intercomms, CICS, etc.). A transaction-id defined to Intercomm (in the BTVRBTB) as assigned to one system can be dynamically reassigned to another APPC system or unassigned (for local processing).

To paraphrase:

a) Terminals in session with another system can invoke Intercomm transactions and have the responses returned to them.

b) Terminals in session with Intercomm can invoke transactions in other systems and have the responses returned to them.

c) Intercomm application subsystems can invoke transactions in other systems and have the responses returned to them or to the originating terminal.

Chapters 2 and 3 describe the passive support in the CICS and Intercomm address spaces respectively, while Chapter 4 describes the Intercomm address space in active mode. Chapter 5 describes the Intercomm subsystem INITLU6 interface. Chapter 6 describes the necessary VTAM session parameters for both passive and active mode support. Chapter 7 describes Intercomm commands affected by LU6.2 support. Appendix A describes Intercomm macro definition changes and recommendations for implementing LU6.2 support. Appendix B lists Intercomm messages related to LU6.2 support. Appendix C lists restrictions related to using the LU6.2 support.

Definition of terms as used in this manual:

VTAM Application - a load module executing in an address space on an IBM mainframe that communicates with IBM's SNA via VTAM to support terminal and/or LU6.2 processing.

LU6.2 Processing - supports communication via VTAM with another VTAM application for passing transaction requests between the applications for processing.

APPC System - a VTAM application which supports LU6.2 processing on an ISC link.

ISC Link - an Inter-system Communication link between two VTAM applications.

LU6.2 Link - an ISC link between two VTAM applications where more than one session exists. Each session has one of the VTAM applications acting as the Primary end of the link and the other VTAM application as the Secondary end of the link. For each of the VTAM applications one of the Primary end link sessions is considered the master (or control) session (used for controlling conditions of the link) and the others are generally for sending transactions and receiving their responses. The Secondary end links are generally used for receiving transactions and returning their responses.

Parallel Session - one of the many sessions in an LU6.2 link. For each and every session, one end of the link is considered the primary logical unit (PLU) and the other is the secondary logical unit (SLU). Not all sessions in a link need have the same end of the link as the PLU.

The application will consider a session in which it is the PLU as one of its primary sessions. Likewise, it will consider a session in which it is the SLU as one of its secondary sessions.

Under normal conditions the session remains in effect until terminated by either end of the link.

Primary Session - a parallel session on an LU6.2 link in which the VTAM application is the PLU. The VTAM application will use primary sessions to pass transactions to the partner in the LU6.2 link and receive the resultant responses. One Primary Session is reserved to be the master (control) session which is used for controlling the link and can be used for establishing the link if this VTAM application is initiating the link. No user data transactions are sent on the master session, only link control transactions. If the VTAM application (such as INTERCOMM or CICS) does not use the APPCCMD macro or high level APPC/MVS interface, then it needs to manage the use of its primary sessions itself. It does this by having control blocks defined to represent each session and having them chained to each other, and to the master control block. These control blocks will exist on different chains based upon the status of the session. When a session is being used to process a transaction it is said to be in an LU6.2 conversation state. In Intercomm these control blocks are LUNITs. For short, these control blocks are known as LU's. If the control block (LU) is representing a primary session, it will also be known as a PLU.

These sessions are also known as contention winner sessions.

Secondary Session - a parallel session on an LU6.2 link in which the VTAM application is the SLU. For each secondary session in this application, the other application in the LU6.2 link will have a primary session. One secondary session will correspond to the master session in the other application. Again, a VTAM application (such as INTERCOMM or CICS) which does not use the APPCCMD macro or high level APPC/MVS interface will need to have control blocks defined to represent each session and having them chained to the master control block. Transactions will be received from the partner application on a

secondary session and all responses are sent back over the same session to be received by the corresponding primary session in the partner application. In Intercomm, these control blocks are LUNITs (as for Primary Sessions), and will be known as an SLU when representing a secondary session.

These sessions are also known as non-contention winner sessions.

LU6.2 Conversation - the processing of a single transaction on a parallel session in an LU6.2 link. A single transaction and its response(s) can consist of many data transfers on the session in either direction, and is determined by VTAM bracket protocol. That is, a conversation is initiated with a begin-bracket (together with an FMH-5 for the transaction) and is terminated by an end-bracket (which may be conditional).

Session Counts - the initial number of parallel sessions on a LU6.2 link is set when the link is being established. This is accomplished by a control transaction called CNOS (Change Number Of Sessions) being sent on the master session of the application initiating the link. This transaction defines the total session count in the link and the number of those sessions which should have the initiator system as the PLU. The modename, and other control information is also passed in this transaction.

Should the application receiving the CNOS transaction not approve of the session counts, it will return a negotiated response, with the values specified being the lower of the requested, or its defined, session counts.

A CNOS exchange, initiated by either application in the link on its master session, can subsequently be used to modify these session counts, but this is not currently supported by Intercomm.

While it is usual that transactions are initiated by the sending application by starting an LU6.2 conversation on a session in which it is the PLU, it can request (should no primary sessions be available) that such a conversation be initiated on a secondary session. A secondary session that has no LU6.2 conversation in progress can be asked to carry the transaction by means of a BID. This can be accepted or rejected by the partner application. Intercomm currently does not support the sending of a BID, nor the receiving of a BID, on its 6.2 sessions.

Modename - during the establishment of sessions between the two VTAM applications, a modename must be used to identify the type of session being established. A minimum of two modenames are required, and must be defined in the VTAM region Logon Mode Table (LM Table) via MODEENT macros. The first modename is fixed as SNASVCMG (SNA Service Manager) and is used for the master sessions in the link (it will be used for the master sessions in all LU6.2 links). The other modename(s) is (are) of user's choice and allows for sessions to be grouped together by modename based on user defined criteria. Multiple user modenames on one LU6.2 link are not currently supported by Intercomm.

General Data Stream (GDS) - the SNA required format for transmitting data and responses on an LU6.2 link. The data stream (GDS) may or may not contain Function Management Headers (FMH's) to indicate certain conditions or functions in the state of, or initiation of, a transaction.

Function Management Header (FMH) - the part of a data stream containing control information. SNA has defined specific types of FMH's for LU6.2 protocol. Two are of particular importance: the FMH-5, which is used to request the initiation of a transaction in the partner system, and the FMH-7 which is used to reject an initiation request.

For Intercomm purchasers of LU6.2 support, the layout of the General Data Stream and the Function Management Header formats are provided in a COPY member called VT62DATA, which is for internal use only.

Chapter 2

THE CICS ADDRESS SPACE

2.1 INTRODUCTION

In order to allow CICS to initiate transactions in Intercomm (whether executing in passive or active mode), a CICS user application program needs to be written and Intercomm must be defined as an APPC application to CICS. The complexity of the CICS application program will depend on user requirements and Intercomm services that are to be supported. A sample of such a transaction invoking program is provided later in this chapter, together with some indications as to what may need to be added.

2.2 MESSAGE PATH TO INTERCOMM

The path of a message from a terminal defined to CICS, which is to be processed in Intercomm, is illustrated in Figure 1. Each step is briefly described below.

(1) An Intercomm transaction is entered at a terminal in session with (logged on to) CICS.

(2) CICS control logic routes this input to the special CICS application program.

(3) Through the use of LU6.2 EXEC CICS interface commands, the CICS program allocates the use of a 6.2 session and starts a conversation on that session by passing the transaction using CICS terminal control.

(4) The Intercomm Front End receives the transaction, and after session dependent processing, passes the message to the standard Intercomm input processing logic.

(5) The Intercomm Front End Verb Table (BTVRBTB) is searched for the transaction code, Edit Utility processing is noted if requested, and the message is placed on the pertinent Subsystem Queue.

(6) The Intercomm user subsystem is scheduled and the message is processed.

(7) The Intercomm user subsystem produces a response, and in a normal manner queues the output message for the session "terminal".

(8) The output message is placed on the session terminal's output queue.

(9) The Front End dequeues the output message and reformats it as necessary for LU6.2 protocol.

(10) The output message is sent to CICS across the same session that the input came in on.

(11) The CICS application program receives the response via CICS terminal control and processes it as necessary.

Steps 7 through 11 can be repeated until the entire terminal output is produced. (The determination is made at steps 7/8/9 when the message(s) are queued. Intercomm will deem the conversation over and will say so to CICS in step 10 if the message queued in step 7 does not say "release next" message).

(12) Once the entire output has been received, the CICS application program ends the conversation and frees up the use of the 6.2 session with Intercomm, and then passes the accumulated and processed message(s) back to CICS terminal control for sending to the terminal.

(Depending on the precise CICS application design, each message may be transmitted to the terminal separately, but the important fact is that the terminal is not ready to produce another input until all of the output is sent.)

(13) CICS will actually transmit the messages(s) back to the terminal. The terminal operator is now free to enter another input message which may be either another Intercommm transaction, or a local CICS transaction.

Note: in an Intercomm Multiregion environment, the processing subsystem may be in a satellite region. That is, steps 5 and 8 may include transfer of the input message to a satellite region and of the output response back to the control region.

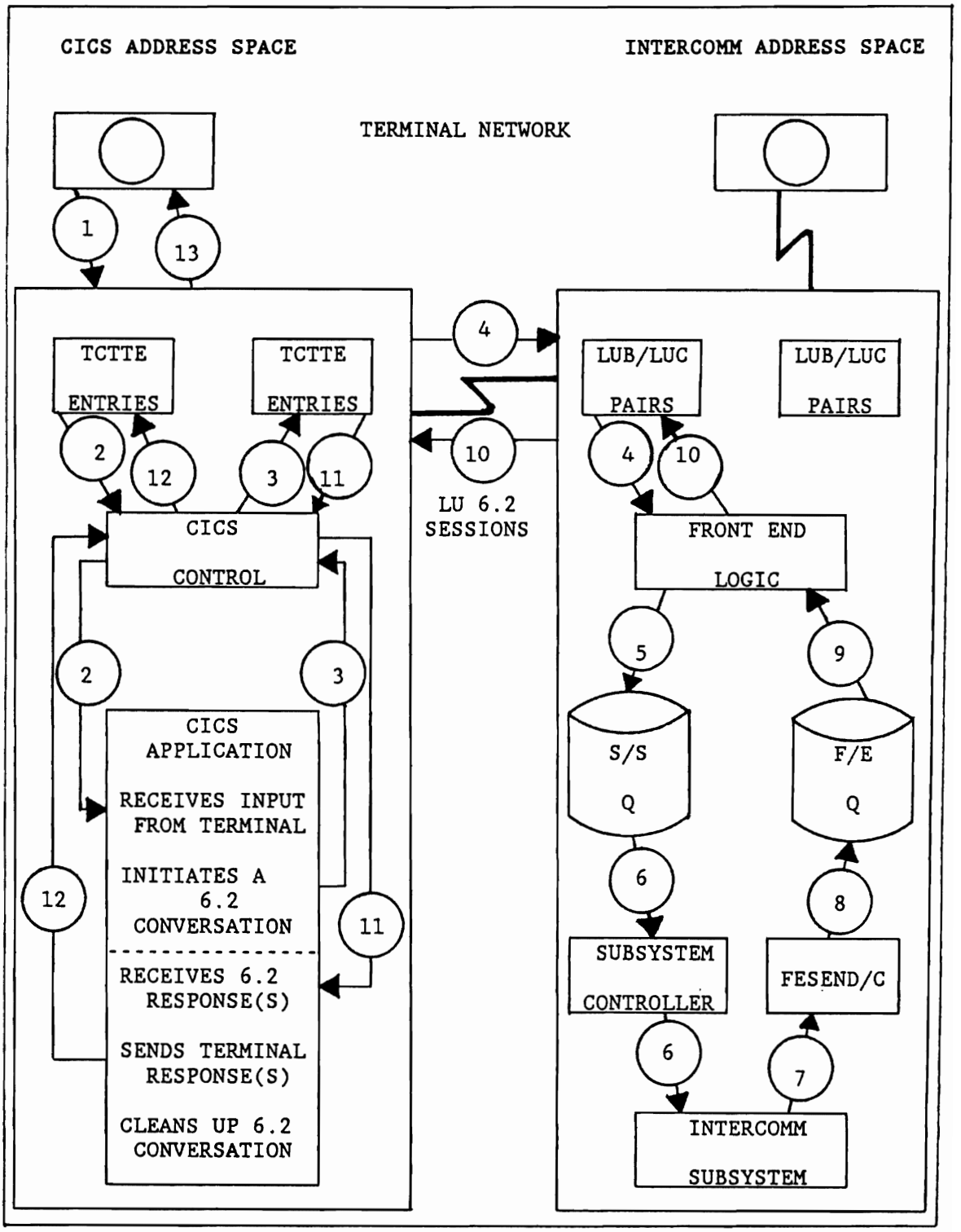

Figure 1. CICS to Intercomm Message Path

2.3 TABLE DEFINITIONS

Every Intercomm transaction code (verb) that may be invoked by a CICS terminal must be defined in the CICS Program Control Table. In the simplest case, each entry points to the same CICS application program. This is not a mandatory requirement, but depends on user requirements. See the next section for further implications for CICS application program coding.

The CICS Terminal Control Table (TCT) must have entries defining Intercomm as an ISC link. Sample table entries are as follows:

```
Icom      DFHTCT TYPE=SYSTEM,                    *
                 ACCMETH=VTAM,                   *
                 CONNECT=AUTO,                   *
                 FEATURE=PARALLEL,               *
                 NETNAME=Icomappl,               *
                 SYSIDNT=Icom,                   *
                 TRMTYPE=LUTYPE62

          DFHTCT TYPE=MODESET,                   *
                 CONNECT=AUTO,                   *
                 BUFFER=bufsize,                 *
                 RUSIZE=bufsize,                 *
                 MAXSESS=(nn,mm),                *
                 MODENAM=modename,               *
                 SYSIDNT=Icom
```

where:

- **Icom** is the internal identification of the Intercomm VTAM application, which is also used by the CICS application program EXEC CICS session commands
- **bufsize** is the maximum allowed Request Unit size between CICS and Intercomm (default is 256)
- **Icomappl** is the VTAM applid of the Intercomm VTAM APPC system (must be 8 characters under Release 9)
- **nn** is the maximum number of non-control parallel sessions to establish between CICS and Intercomm (this value can be altered by a CICS CEMT command).
- **mm** is the number of these sessions that are to be considered contention winner (or primary) sessions, that is, available to CICS to initiate user transactions in Intercomm. This value should be less than nn if Intercomm is also being used in an active mode with this CICS, or equal to nn when only passive support is required. If only the basic LU6.2 feature is available then this value should be equal to nn.
- **modename** is the name of a MODETAB entry in the VTAM region Logon Mode Table to be used for non-control sessions between CICS and Intercomm. CICSMODE is recommended and expected by Intercomm, unless changed as described in the next chapter.

Please note the following:

- The value coded for bufsize on the DFHTCT TYPE=MODESET should match that coded for the Intercomm VTLSB macro, SOUTSEG parameter, as described in Appendix A. The matching values may be higher, but should not be lower, than 256.

- CICS is being told that it is the contention winner on all sessions, if the original passive support is being used, otherwise a session count negotiation takes place.

- The Intercomm control session is not included in the MAXSESS count, and does not use the mode name defined on the DFHTCT TYPE=MODESET.

- If it is desired to connect to more than one Intercomm VTAM application, then additional pairs of DFHTCT macros must be defined in the TCT, with different NETNAME and SYSIDNT parameter values. In this case, different CICS application programs should be used to establish sessions with the different Intercomm VTAM applications.

- The CICS System Initialization Table (DFHSIT macro) ISC parameter must include the Intercomm LU6.2 link(s) in the total count of ISC links.

Refer to IBM's CICS Intercommunication Facilities Guide (1.7) or Intercommunication Guide (2.1 or higher) for further installation requirements.

2.4 A SAMPLE CICS APPLICATION PROGRAM

The CICS application program can be as simple or as complicated as the user desires. In the simplest case, the program should allocate an LU6.2 session for use, invoke the transaction in Intercomm, and receive the response. It should then deallocate the LU6.2 session, and return the received response to the originating terminal. In this case, it is assumed that the Intercomm region has fully formatted the output, and all that CICS needs to do is to remove the EOB/EOT (if any) at the end of the text. A system having only one terminal-type (that is, all 3270 terminals) is also assumed, so that the Intercomm region knows how to format the message. In the sample program illustrated in Figure 2, the transaction-id (Intercomm verb) is removed from the message text, and the input terminal-id is appended to the end of the message. The removal of the verb is a requirement as its value is passed by the program to Intercomm in control data. The adding of the true terminal-id (see ADD-EXTRA-DATA routine in the program) is for user purposes only, and in the Intercomm region, a user INQUEUE exit can be utilized to make use of this information. (See discussion in Chapter 3 on Intercomm Back End Station Table entries.)

Note also that the EIBFREE field setting is examined to see whether Intercomm will return further messages for the same transaction. This will happen if the Intercomm Front End is told that more messages are to follow, via the "release-next" queuing option of FESEND, and is equivalent to the Step 7 through Step 11 loop in the message flow illustrated in Figure 1 and described in the beginning of this chapter. Normally, it is expected that there will be one response message to an input transaction, unless the Intercomm subsystem constructs the screen image via multiple output message parts. Printer output should be queued directly to a printer defined in the Intercomm Front End Network Table. The printer may be shared between the Intercomm and CICS address spaces if defined to Intercomm as a shared LU, as described in the SNA Terminal Support Guide.

The sample COBOL CICS program listed in Figure 2 is provided in source form with the other LU6.2 support modules on SYMLU6 (Release 9) or SYMREL (Release 10) and is called CICSSAMP. Note that Intercomm does not support a SYNCLEVEL above 1 nor a SYNCPOINT command. PIP data is currently rejected by Intercomm, if used in CICS.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. TESTICOM.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  LENGTH-FIELDS.
    03  TERM-RECV-LENGTH      PIC S9(4) COMP  VALUE +3072.
    03  TERM-SEND-LENGTH      PIC S9(4) COMP.
    03  ICOM-RECV-LENGTH      PIC S9(4) COMP  VALUE +3072.
    03  ICOM-SEND-LENGTH      PIC S9(4) COMP.
    03  TRANSID-LENGTH        PIC S9(4) COMP  VALUE +4.
01  SWITCHES.
    03  ERROR-SWITCH          PIC S9(4) COMP-3  VALUE +0.
        88  ERROR-OCCURRED    VALUE +1.
    03  ERASE-SWITCH          PIC S9(4) COMP-3  VALUE +0.
        88  ERASE-OCCURRED    VALUE +1.
01  WORK-AREAS.
    03  SUB-1                 PIC S9(5) COMP-3.
    03  CONV-ID               PIC X(8).
    03  ICOM-VERB             PIC X(4).
    03  SBA-FLAG              PIC S9(4) COMP  VALUE +17.
    03  SBA-FLAG-R REDEFINES SBA-FLAG.
        05  FILLER            PIC X.
        05  HEX-11            PIC X.
    03  EXTRA-LL              PIC S9(4) COMP  VALUE +7.
    03  EXTRA-LL-R REDEFINES EXTRA-LL.
        05  HEX-00            PIC X.
        05  HEX-07            PIC X.
    03  ICOM-FORMAT-FLAG      PIC S9(4) COMP  VALUE +38.
    03  ICOM-FORMAT-R REDEFINES ICOM-FORMAT-FLAG.
        05  FILLER            PIC X.
        05  HEX-26            PIC X.
01  ICOM-MESSAGE.
    03  ICOM-MSG-BYTE         PIC X
            OCCURS 3072 TIMES.
01  OUTPUT-MESSAGE.
    03  OUTPUT-MSG-BYTE       PIC X
            OCCURS 3072 TIMES.
01  ERROR-TRANSID.
    03  FILLER                PIC X  VALUE 'Z'.
    03  ERROR-NUMBER          PIC X(3).
01  TERMINAL-ID-AREA.
    03  VTAM-TERMID.
        05  FILLER            PIC X  VALUE 'B'.
        05  CICS-TERMID       PIC X(4).
    03  FILLER REDEFINES VTAM-TERMID.
        05  ICOM-TERMID       PIC X
                OCCURS 5 TIMES.
```

Figure 2. Sample CICS Application Program (Part 1 of 7)

```
LINKAGE SECTION.
01  DFHCOMMAREA.
    03  DFHCA-TERMID          PIC X(4).
    03  DFHCA-TRANSID         PIC X(4).
    03  DFHCA-DATA            PIC X(8).
01  BLL-CELLS.
    03  FILLER                PIC S9(8) COMP.
    03  ADDR-TIOA-DATA        PIC S9(8) COMP.
01  TIOA-DATA.
    03  TIOA-FORMAT-1.
        05  TIOA-FORMAT-1-BYTE-1          PIC X.
        05  FILLER                        PIC X(2047).
    03  TIOA-FORMAT-2 REDEFINES TIOA-FORMAT-1.
        05  FILLER                        PIC X(8).
        05  TIOA-FORMAT-2-DATA            PIC X(2040).
    03  TIOA-FORMAT-3 REDEFINES TIOA-FORMAT-1.
        05  FILLER                        PIC X(5).
        05  TIOA-FORMAT-3-DATA            PIC X(2043).
PROCEDURE DIVISION.

    EXEC CICS IGNORE CONDITION
        ERROR
        END-EXEC.

    IF EIBCALEN EQUAL 0
        PERFORM L02-GET-DATA-FROM-TERM
            THRU L02-GET-DATA-FROM-TERM-EXIT
    ELSE
        PERFORM L02-GET-DATA-FROM-DFHCA
            THRU L02-GET-DATA-FROM-DFHCA-EXIT.
    PERFORM L02-ADD-EXTRA-DATA
        THRU L02-ADD-EXTRA-DATA-EXIT.
    IF ERROR-OCCURRED
        GO TO L01-PROGRAM-EXIT.

    PERFORM L02-GET-LU62-SESSION
        THRU L02-GET-LU62-SESSION-EXIT.
    IF ERROR-OCCURRED
        GO TO L01-PROGRAM-EXIT.

    PERFORM L02-START-REMOTE-TRANS
        THRU L02-START-REMOTE-TRANS-EXIT.
    IF ERROR-OCCURRED
        GO TO L01-PROGRAM-EXIT.

    PERFORM L02-LU62-CONVERSATION
        THRU L02-LU62-CONVERSATION-EXIT.
    IF ERROR-OCCURRED
        GO TO L01-PROGRAM-EXIT.
```

Figure 2. Sample CICS Application Program (Part 2 of 7)

```
           PERFORM L02-FREE-LU62-SESSION
               THRU L02-FREE-LU62-SESSION-EXIT.
           IF ERROR-OCCURRED
               GO TO L01-PROGRAM-EXIT.

           PERFORM L04-FORMAT-OUTPUT-MESSAGE
               THRU L04-FORMAT-OUTPUT-MESSAGE-EXIT.

           IF ERASE-OCCURRED
               GO TO L01-SEND-NOERASE.

           EXEC CICS SEND
               FROM(OUTPUT-MESSAGE)
               LENGTH(TERM-SEND-LENGTH)
               ERASE
               END-EXEC.
           MOVE +1 TO ERASE-SWITCH
           GO TO L01-PROGRAM-EXIT.

       L01-SEND-NOERASE.
           EXEC CICS SEND
               FROM(OUTPUT-MESSAGE)
               LENGTH(TERM-SEND-LENGTH)
               END-EXEC.

       L01-PROGRAM-EXIT.
           EXEC CICS RETURN
               END-EXEC.
           GOBACK.


       L02-GET-DATA-FROM-TERM.
           MOVE EIBTRMID TO CICS-TERMID.
           MOVE EIBTRNID TO ICOM-VERB.
           EXEC CICS RECEIVE
               LENGTH(TERM-RECV-LENGTH)
               SET(ADDR-TIOA-DATA)
               END-EXEC.
           PERFORM L03-STRIP-VERB
               THRU L03-STRIP-VERB-EXIT.
           MOVE TERM-RECV-LENGTH TO ICOM-SEND-LENGTH.
       L02-GET-DATA-FROM-TERM-EXIT.
           EXIT.
```

Figure 2. Sample CICS Application Program (Part 3 of 7)

```
       L02-GET-DATA-FROM-DFHCA.
           MOVE DFHCA-TERMID TO CICS-TERMID.
           MOVE DFHCA-TRANSID TO ICOM-VERB.
           MOVE DFHCA-DATA TO ICOM-MESSAGE.
           MOVE +1 TO ICOM-SEND-LENGTH.
       L02-GET-DATA-FROM-DFHCA-EXIT.
           EXIT.

       L02-ADD-EXTRA-DATA.
           PERFORM L03-MOVE-TERMID
               THRU L03-MOVE-TERMID-EXIT
                   VARYING SUB-1 FROM +1 BY +1
                   UNTIL SUB-1 GREATER THAN +5.
           ADD +1 TO ICOM-SEND-LENGTH.
           MOVE HEX-00 TO ICOM-MSG-BYTE (ICOM-SEND-LENGTH).
           ADD +1 TO ICOM-SEND-LENGTH.
           MOVE HEX-07 TO ICOM-MSG-BYTE (ICOM-SEND-LENGTH).
       L02-ADD-EXTRA-DATA-EXIT.
           EXIT.

       L02-GET-LU62-SESSION.
           EXEC CICS ALLOCATE
               SYSID('ICOM')
               NOQUEUE
               END-EXEC.
           IF EIBRCODE NOT EQUAL LOW-VALUES
               MOVE +1 TO ERROR-SWITCH
               MOVE '990' TO ERROR-NUMBER
               PERFORM L04-START-ERROR-TRANS
                   THRU L04-START-ERROR-TRANS-EXIT
               GO TO L02-GET-LU62-SESSION-EXIT.
           MOVE EIBRSRCE TO CONV-ID.
       L02-GET-LU62-SESSION-EXIT.
           EXIT.

       L02-START-REMOTE-TRANS.
           EXEC CICS CONNECT PROCESS
               CONVID(CONV-ID)
               PROCNAME(ICOM-VERB)
               PROCLENGTH(TRANSID-LENGTH)
               SYNCLEVEL(0)
               END-EXEC.
           IF EIBRCODE NOT EQUAL LOW-VALUES
               MOVE +1 TO ERROR-SWITCH
               MOVE '991' TO ERROR-NUMBER
               PERFORM L04-START-ERROR-TRANS
                   THRU L04-START-ERROR-TRANS-EXIT.
       L02-START-REMOTE-TRANS-EXIT.
           EXIT.
```

Figure 2. Sample CICS Application Program (Part 4 of 7)

```
L02-LU62-CONVERSATION.
    EXEC CICS CONVERSE
        CONVID(CONV-ID)
        FROM(ICOM-MESSAGE)
            FROMLENGTH(ICOM-SEND-LENGTH)
        INTO(OUTPUT-MESSAGE)
            TOLENGTH(ICOM-RECV-LENGTH)
        END-EXEC.
    IF EIBCONF EQUAL HIGH-VALUES
        EXEC CICS ISSUE CONFIRMATION
            CONVID(CONV-ID)
            END-EXEC.
    PERFORM L03-RECEIVE-EXTRA-MSGS
        THRU L03-RECEIVE-EXTRA-MSGS-EXIT
            UNTIL EIBFREE EQUAL HIGH-VALUES.
    MOVE ICOM-RECV-LENGTH TO TERM-SEND-LENGTH.
L02-LU62-CONVERSATION-EXIT.
    EXIT.

L02-FREE-LU62-SESSION.
    EXEC CICS FREE
        CONVID(CONV-ID)
        END-EXEC.
    IF EIBRCODE NOT EQUAL LOW-VALUES
        MOVE +1 TO ERROR-SWITCH
        MOVE '992' TO ERROR-NUMBER
        PERFORM L04-START-ERROR-TRANS
            THRU L04-START-ERROR-TRANS-EXIT.
L02-FREE-LU62-SESSION-EXIT.
    EXIT.


L03-STRIP-VERB.
    IF TIOA-FORMAT-1-BYTE-1 EQUAL HEX-11
        IF TERM-RECV-LENGTH IS GREATER THAN 8
            MOVE TIOA-FORMAT-2-DATA TO ICOM-MESSAGE
            SUBTRACT 8 FROM TERM-RECV-LENGTH
        ELSE
            MOVE +0 TO TERM-RECV-LENGTH
    ELSE
        IF TERM-RECV-LENGTH IS GREATER THAN 5
            MOVE TIOA-FORMAT-3-DATA TO ICOM-MESSAGE
            SUBTRACT 5 FROM TERM-RECV-LENGTH
        ELSE
            MOVE +0 TO TERM-RECV-LENGTH.
L03-STRIP-VERB-EXIT.
    EXIT.
```

Figure 2. Sample CICS Application Program (Part 5 of 7)

```
L03-RECEIVE-EXTRA-MSGS.
    MOVE ICOM-RECV-LENGTH TO TERM-SEND-LENGTH
    PERFORM L04-FORMAT-OUTPUT-MESSAGE
        THRU L04-FORMAT-OUTPUT-MESSAGE-EXIT.
    IF ERASE-OCCURRED
        GO TO L03-SEND-NOERASE.
    EXEC CICS SEND
        FROM(OUTPUT-MESSAGE)
        LENGTH(TERM-SEND-LENGTH)
        ERASE
        END-EXEC.
    MOVE +1 TO ERASE-SWITCH.
    GO TO L03-GET-NEXT-MESSAGE.
L03-SEND-NOERASE.
    EXEC CICS SEND
        FROM(OUTPUT-MESSAGE)
        LENGTH(TERM-SEND-LENGTH)
        END-EXEC.
L03-GET-NEXT-MESSAGE.
    EXEC CICS SEND
        CONVID(CONV-ID)
        INVITE
        WAIT
        END-EXEC.
    MOVE +3072 TO ICOM-RECV-LENGTH.
    EXEC CICS RECEIVE
        CONVID(CONV-ID)
        INTO(OUTPUT-MESSAGE)
        LENGTH(ICOM-RECV-LENGTH)
        END-EXEC.
    IF EIBCONF EQUAL HIGH-VALUES
        EXEC CICS ISSUE CONFIRMATION
        CONVID(CONV-ID)
        END-EXEC.
L03-RECEIVE-EXTRA-MSGS-EXIT.
    EXIT.

L03-MOVE-TERMID.
        ADD +1 TO ICOM-SEND-LENGTH
        MOVE ICOM-TERMID (SUB-1)
            TO ICOM-MSG-BYTE (ICOM-SEND-LENGTH).
L03-MOVE-TERMID-EXIT.
    EXIT.
```

Figure 2. Sample CICS Application Program (Part 6 of 7)

```
L04-FORMAT-OUTPUT-MESSAGE.
    MOVE +1 TO SUB-1.
    PERFORM L05-GET-END-MESSAGE
        THRU L05-GET-END-MESSAGE-EXIT
            UNTIL SUB-1 GREATER THAN TERM-SEND-LENGTH
               OR OUTPUT-MSG-BYTE (SUB-1) EQUAL HEX-26.
    IF OUTPUT-MSG-BYTE (SUB-1) EQUAL HEX-26
        MOVE SPACE TO OUTPUT-MSG-BYTE (SUB-1)
        SUBTRACT +1 FROM SUB-1
            GIVING TERM-SEND-LENGTH.
L04-FORMAT-OUTPUT-MESSAGE-EXIT.
    EXIT.

L04-START-ERROR-TRANS.
    EXEC CICS START
        TRANSID(ERROR-TRANSID)
        TERMID(CICS-TERMID)
        END-EXEC.
L04-START-ERROR-TRANS-EXIT.
    EXIT.


L05-GET-END-MESSAGE.
        ADD +1 TO SUB-1.
L05-GET-END-MESSAGE-EXIT.
    EXIT.
```

Figure 2. Sample CICS Application Program (Part 7 of 7)

Chapter 3

INTERCOMM IN PASSIVE MODE

3.1 MESSAGE FLOW

The data comes over an LU6.2 session in the form of a General Data Stream, is accepted by Intercomm, formatted into a standard Intercomm message (header/verb/text), and queued for the Back End. If an invalid verb is passed from the other system (CICS), a suitable FMH-7 is sent as a response to the other system. Unless user exits change the situation, the LU6.2 session-id is used as the terminal-id in the MSGHTID field of the message header passed to the Back End. The BMN of this message (MSGHBMN field in input message header) is used as a key for passing a response back over the LU6.2 session. Unsolicited output will not be sent across LU6.2 sessions. Only the output message(s) with the same BMN will be returned. Therefore, all Intercomm subsystems which may process input received via LU6.2 sessions must copy the input message header to the output message area (automatic if MMU is used). If more than one output message is anticipated, then all but the last must be sent using the "release-next" queuing option of FESEND (FESENDC). The transaction is then terminated.

3.2 STATION TABLE DEFINITIONS

It is important to note that as the LU6.2 session (LU) name is used in the MSGHTID field, Station Table entries for these session names must be created in PMISTATB. They should also be defined with the same device characteristics (IBM 3270 CRT's), as it will never be certain, from one invocation to the next, on which session (LU) a particular terminal's input message will be presented to Intercomm. (An alternative is to have the CICS application program perform considerable reformatting of the returned data if the devices are not similar.)

It is also possible to have Station Table entries for all the real terminals defined to the other system (CICS), and use an INQUEUE user exit (see SNA Terminal Support Guide) in Intercomm to place the real terminal-id in the MSGHTID field. (See the sample CICS program in Chapter 2, which adds a terminal-id to the message data.) In this case, it will be the responsibility of a USROTEDT user exit called by FESEND (see Operating Reference Manual) to put the session name back in the MSGHTID field of the output message header, so that the response gets queued for, and sent on, the originating LU6.2 session. The two user exits would be responsible for maintaining a table of Station Table names with the corresponding session name for each input/output message pair. If the terminal-id passed by the (CICS) application program is the VTAM terminal-id, and is defined in the Intercomm VTIDTAB table, the corresponding Intercomm terminal-id can be retrieved via the VTIDCONV macro (see SNA Terminal Support Guide) and placed in the MSGHTID field. In this case, the Station Table would have entries for all VTAM terminals in the combined network, and the LU6.2 session-ids can be omitted. Only VTAM terminals which may log on to, or be acquired by, Intercomm need be defined in the Front End Network Table. See also Section 3.6 regarding user exit processing and error messages.

3.3 DEFINING THE INTERCOMM NETWORK

The following sections describe Intercomm Front End Network Table definition requirements. Sample table coding is provided in Figure 3. Further details on applicable macros and their parameters are provided in Appendix A, and the SNA Terminal Support Guide.

3.3.1 The VCT Macro

No special considerations apply other than to ensure that SEQNO=BTAM is used. This is because the Front End BMN message number is used as a key to a legitimate response on the session and is stored in the LU6.2 session LUB (LUNIT). Should SEQNO=VTAM be used, the number would not be sufficiently unique to ensure discarding unsolicited output. Under Intercomm Release 9, the APPLID parameter (defining the Intercomm APPL name in the VTAM region and on the CICS DFHTCT TYPE=SYSTEM) must be coded on the VCT macro and must be 8 characters long. These requirements are waived under Release 10. Note that the SNMAX value (as well as the EAS value on the Intercomm APPL statement in the VTAM region) may have to be adjusted to include the LU6.2 sessions. Under Release 10, the VCT current session count is increased and compared to the SNMAX value as each 6.2 session is established, however for Release 9, the session count only gets increased when the master session is established.

3.3.2 LUNIT Macro to Define the other (CICS) VTAM Application

One LUNIT macro is coded to represent each VTAM application (CICS) with which Intercomm may establish an LU6.2 link, and is called the master or control LUB/LUC (LUNIT/LCOMP) pair. A SESSION parameter must be specified to define the maximum number of parallel sessions that can be initiated at one time. This value must include the master session, and the secondary session established with modename SNASVCMG to represent the partners master session. The SESSION parameter is used to negotiate (via a CNOS exchange) the number of allowable sessions with Intercomm's partner. (For a CICS link this value should be two higher than that coded on the DFHTCT TYPE=MODESET entry, MAXSESS parameter, when defining Intercomm to CICS.) The LSB and CSB parameters should point to suitably defined VTLSB and VTCSB specification block entries respectively, the definition of which is discussed below. If the (CICS) VTAM application name is longer than five characters, a corresponding Intercomm LU name (terminal-id) is coded on the LUNIT macro, and both names must be in the VTIDTAB table. (Do not define disk queuing for this master LUNIT, because subsystem output messages are never queued to it.) Under Release 10, each LUNIT defining an LU6.2 link master session will also generate a LUB extension or LBX. The prime purpose of this control block is for the extended (active mode) support, but will exist in the passive mode even though it will not be used (see the next Chapter for details on the use of the LBX).

3.3.3 Defining a Pool of Dynamic Sessions

A pool of control blocks (dynamic LUB/LUC pairs or SLUs) is generated by using the DYN parameter on a separate LUNIT macro. This parameter is used to define the size of the pool, a two character name prefix, and the fact that the dynamic pool is for LU6.2 use (optional in Release 10). CSB and LSB parameters are not used for the pool LUNIT definition. If more than one master LUNIT is defined, the size of the pool must be large enough to handle the maximum possible number of sessions across all the master LUNITs for the different VTAM applications.

It is important to note that parameters defining queues will apply to every one of the generated LUB/LUC pairs. For example, if a disk queue is specified in a DFLN parameter, then when the PCEN parameter is defined, it must be small enough to accomodate each and every generated LUB/LUC pair. That is, PCEN must equal 100 divided by the number coded for the DYN parameter (or smaller). The number of RBNs in the data set must be at least 8*(100/smallest PCEN), or some multiple, with a minimum BLKSIZE of 2048. NUMCL must be at least 2 if DFLN not coded (to allow for a timeout message plus the actual response message), or greater if multiple responses may be returned.

For debugging, note that the pool of LUB/LUC pairs are initially chained to each other. When LU6.2 sessions are established, pairs are dechained from the pool and chained to the master LUB/LUC entry defining the (CICS) VTAM application which has created the LU6.2 sessions. The LSB and CSB pointers of the VTAM application master LUNIT are placed in the dynamic LUB/LUC pair when it is chained to the master LUB/LUC.

3.3.4 Defining the Specification Blocks

The respective LUTYPE and COMPTYP parameters of the VTLSB and VTCSB macros pointed to by a CICS control LUNIT must both specify a type of CICS. If the partner VTAM application is not CICS, code ICOM instead. For other VTCSB parameters, see Appendix A.

The only other special consideration is that the VTLSB points to a suitable user bindarea (BNDAREA parameter). For CICS, the user bindarea to be provided contains CICS release-dependent information. Because much of the data in the bindarea is CICS-unique, there is no known IBM macro to generate it. Intercomm provides an LU6.2 VTBIND62 macro to generate a suitable user bindarea which may also be used for Intercomm to Intercomm links. Macro parameters provide the Intercomm and partner system (CICS) applids to the generated area as illustrated in Figure 3, and defined in Appendix A.

The SOUTSEG parameter on the VTLSB macro may be coded to define the maximum Request Unit size that can be used on any of the LU6.2 sessions except the Intercomm and partner system (CICS) master sessions, for which a value of 256 is forced. On all other sessions (for message transmission), the SOUTSEG value is used, if defined, unless the partner system (CICS) initiates the bindarea at session establishment with a smaller value, which is then used instead of SOUTSEG. If the SOUTSEG parameter is not coded, then the Request Unit size is taken from the value provided in the bindarea, or if less than 256, then the length coded for the Intercomm VCT macro, RCANYLN parameter, is used. However, if the determined value is less than 256, then a default value of 256 is substituted. Whichever value is used, it gets stored in the LUBOTSEG field of the LUNIT for the session when it is established. (Refer also to Chapter 2 and Appendix A.)

3.3.5 Modifying the Modename

As described earlier in Chapter 1, two VTAM region Logon Mode Table entries are required for an LU6.2 link. The modename for all non-control sessions is assumed in Intercomm to be CICSMODE. When only passive mode is installed, if a different value is used by the other system(s) (coded for the CICS DFHTCT TYPE=MODESET), and for the MODEENT macro in VTAM, then the DC CL8'CICSMODE' at sequence number 01380000 (in the LUD6CNOS table) in VTLUDM6 must also be changed to the new name. If active mode is installed under Release 10, a different modename can be coded as the third subparameter of the SESSION parameter on the LUNIT defining the master session for each specific LU6.2 link (see Appendix A). The same modename must be used for the non-control sessions by the other partner in the link.

3.4 DEFINING TRANSACTION-IDS (BTVERB MACRO)

Every transaction-id (verb) that may be received from another VTAM application on an LU6.2 session must be defined via a BTVERB macro in the Intercomm Front End Verb Table (BTVRBTB). The codes of the Intercomm user application program (subsystem) to process the transaction must also be given, along with a conversational timeout value (CONV parameter) which should be greater than the TCTV value for the associated subsystem (SYCTTBL macro). If CONV=YES is coded on the VTCSB of the master LUNIT (or coded on the LUNIT defining the dynamic sessions pool), then the CONV timeout value coded on the BTVERB macro is used to ensure a response (NO PROGRAM RESPONSE message) on the LU6.2 session if none is received from the processing subsystem. If Edit Utility processing is used for the transaction, also code the EDIT parameter. Other parameters do not apply to LU6.2 transactions.

3.5 SUBSYSTEM SYCTTBL MACRO CONSIDERATIONS

For subsystems which may process LU6.2 transactions, ensure that the value coded for the MNCL parameter is high enough to provide adequate response time on the link. As traffic increases, it may also be necessary to increase the CONV value on the associated BTVERB macro (see above) to allow for queue time. Under Release 10, on the SYCTTBL macro, also code REJECT=YES.

3.6 USER EXIT PROCESSING AND MSGCOL ERROR MESSAGES

A response in the form of a queuing error message from a receiving Intercomm system depends on the Intercomm release in use and whether an INQUEUE user exit is used (see Section 3.2). On receipt of a transaction on an LU6.2 session, if the receiving Intercomm has the requested verb defined in its BTVRBTB for local processing, then it will queue the transaction via MSGCOL for the associated SYCTTBL. If the SYCTTBL does not exist, or its queue is full, or some other problem occurs (subsystem not available, etc.), then the appropriate Intercomm error message will be generated. However, if the receiving Intercomm system is a Multiregion Intercomm and the processing SYCTTBL is in a satellite region and the message is successfully queued for that satellite region, then an Intercomm error message is generated only if there is a queuing problem in the satellite region. Note that if the Message Collection information message MM103I or MM104I (see Messages and Codes) is generated (input message queued but processing deferred), then if the input message is actually processed at a later time, the output response from the subsystem will be discarded by the responding Intercomm Front End (unexpected response).

Under Release 9, whether or not an INQUEUE exit is used, a USROTEDT exit (see Section 3.2) may be coded (or modified) to trap a queuing error message from Message Collection. This message will not have a proper BMN number in the header, so it will be discarded by the Front End. To force the message to be sent back as a response, every message to be queued for an LU6.2 session terminal-id must be examined. The text of a Message Collection response will always start with the nine-character message identifier (see Messages and Codes, Chapter 2) within which the fourth and fifth letters will contain the component identifier MM (for MSGCOL). If the error message is to be sent back to the originating system, it will then be necessary to obtain (via the EXTERM macro) the address of the LUC component of the LUNIT for the LU6.2 session defined in MSGHTID of the message header. The fullword field LUCCBMN in the LUC contains the expected response BMN number. Move the last 2 bytes to the MSGHBMN field in the message header. This technique can be used for any Intercomm or user-generated error message where the original input message header is not copied for the output message (as in a message generated via a PMIWTO macro).

Under Release 10, if an error message is generated via a PMIWTO with the TERM parameter (as in MSGCOL processing in the module BLMSGCOL), and an INQUEUE exit is not used to change the LU6.2 session-id in the input message header MSGHTID field, then the correct BMN number will be put in the message header by WTOMOD formatting of the error message as a response message. This does not apply if the session-id is changed (TERM parameter does not indicate an LU6.2 secondary session-name) or if the message is generated in a satellite region. In the latter cases, user exit processing as in Release 9 (see above) is needed if the error message is to be sent back to the originating system. Note that under Release 10, MSGHBMN is a 3-byte number (move low-order 3 bytes of LUCCBMN) unless the Release 9 2-byte version is used in the system.

In any event, a conversational timeout message will eventually prevent a 'hung' condition if neither an error message nor a subsystem response is correctly queued for the inputting LU6.2 session.

The above discussion does not apply to subsystem processing error messages generated via the Intercomm released USRCANC error exit (PMICANC - see Operating Reference Manual), which uses a copy of the subsystem's input message header.

In a Multiregion environment, where the processing subsystem is defined in a satellite region, but a problem occurs in message transfer from the control region, then an applicable error message is generated (by MRQMNGR) by copying the input message header (with BMN number), not via a PMIWTO (no error message-id in the text). The text of such error messages is one of the following:

REGION xxxxxxxx IS INACTIVE. YOUR MESSAGE WAS FLUSHED

PROGRAM TEMPORARILY STOPPED. YOUR MESSAGE WAS FLUSHED

REGION xxxxxxxx IS INACTIVE. YOUR MESSAGE WAS QUEUED.

The last error message can occur only if subsystem hold queues (on DDQ disk data sets) are defined. Should such a queued message eventually be passed to a reactivated satellite region, then the response from the subsystem will be discarded by the Intercomm VTAM Front End (unexpected response) which considers the error message to have been the only valid response on the LU6.2 session. Note that MRQMNGR is called by Message Collection (MSGCOL) and returns a non-zero code if a queuing error (queue full or subsystem code not found) occurs. MSGCOL then issues one of its standard error messages (see above), unless the message can be queued for a subsystem in the control region.

```
VTAM     TITLE 'VTSAMPC-SAMPLE VTAM NETWORK TABLE WITH CICS'
VTSAMPC  CSECT
*
*   DEFINE THE VTAM CONTROL TABLE
*
         VCT   SECT=CSECT,APPLID=INTERCOM,PASSWD=ICOMPASS,START=YES,    *
               SNMAX=100,RCVNO=3,RCVRSP=5,RCANYLN=256,                  *
               SHUTDTL=50,MXSDTHD=50,TRACE=(EWTO,ETRC),SEQNO=BTAM
*
*   DEFINE THE LOGICAL UNITS AND COMPONENTS  (LUB/LUC PAIRS)
*
*        3270 SDLC (SNA)  CRTS AND PRINTERS
*
         LUNIT NAME=LU040,ALT=LU041,LSB=LS3270S,CSB=CS3270SC,           *
               NUMCL=10                        CRT  #1
*
         LUNIT NAME=LU041,ALT=LU040,LSB=LS3270S,CSB=CS3270SC            *
               NUMCL=10                        CRT  #2
*
         LUNIT NAME=LU042,LSB=LS3270SP,CSB=CS3270SP,                    *
               DFLN=VTAMQU03,PCEN=20           HARDCOPY  #1
*
         LUNIT NAME=LU043,LSB=LS3270SS,CSB=CS3270SP,                    *
               DFNL=VTAMQU03,PCEN=20           HARDCOPY  #2(SHARED)
*
*        CICS LU 6.2 SYSTEM LINKS
*
         LUNIT NAME=CICS1,LSB=LSBCICS1,ACQ=YES,SESSION=10,              *
               CSB=CSBCICS,NUMCL=2
*
         LUNIT NAME=CICS2,LSB=LSBCICS2,ACQ=YES,SESSION=10,              *
               CSB=CSBCICS,NUMCL=2
*
*        A POOL OF DYNAMIC LUB/LUC PAIRS FOR 6.2 SESSIONS
*
         LUNIT DYN=(25,LU6,DL),DFLN=VTAMQU05,PCEN=2,NUMCL=4
*
*   END OF LUB/LUC DEFINITIONS -- PMISTOP AND PCENSCT MUST FOLLOW
*
         PMISTOP
         PCENSCT
*
*   DEFINE THE SPECIFICATION BLOCKS
*
*        3270 SDLC (SNA)  CRTS AND PRINTERS
*
LS3270S  VTLSB LUTYPE=3270S,TRSTBL=LOWTOUP,TIMEOUT=30
*
LS3270SP VTLSB LUTYPE=3270S,TIMEOUT=30,ASRESP=D,SRESP=(D,1)
*
LS3270SS VTLSB LUTYPE=3270S,TIMEOUT=30,ASRESP=D,SRESP=(D,1),            *
               RELREQ=RELEASE,OUTQ=ACQUIRE,ULVB=SHARE     SHARED LUNIT
```

Figure 3. Sample Intercomm Network Table LU6.2 Definitions - Passive Mode (Part 1)

```
CS3270SC VTCSB COMPTYP=3270S,CRT=YES,AIDGRP=1,CTCHAR=F5C3114040
*
CS3270SP VTCSB COMPTYP=3270S,CRT=NO,CTCHAR=F54C114040
*
SHARE    VTLVB OTQUEUE=VTUROTX1,SNDNRM=VTURSDX1
*
*        THE CICS 6.2 SESSION LINKS
*
LSBCICS1 VTLSB LUTYPE=CICS,BNDAREA=BIND621,SOUTSEG=1024
*
LSBCICS2 VTLSB LUTYPE=CICS,BNDAREA=BIND622,SOUTSEG=1024
*
CSBCICS  VTCSB COMPTYP=CICS,CONV=YES   (CSB SHARED BY BOTH CICS SYSTEMS)
*
BIND621  VTBIND62 APPLID=INTERCOM,CAPPLID=CICSREG1
*
BIND622  VTBIND62 APPLID=INTERCOM,CAPPLID=CICSREG2
*
*   END OF LOGICAL UNIT AND COMPONENT SPECIFICATION BLOCKS
*
*
*   MISCELLANEOUS TABLES TO COMPLETE DEFINITIONS
*
*
*   1)   TRANSLATION TABLES
*
LOWTOUP  EQU   *
         COPY  TRAN3270
         PMISTOP
*
*   2)   AID GROUP AND AIDDATA DEFINITIONS AND TABLES
*
         AIDGRP 1,CLEAR=1
*
         AIDDATA 1,'RLSE'
         AIDDATA END
*
*   3)   THE VTAM - INTERCOMM SYNONYM TABLE
*
         VTIDTAB VTAMIDS=(TSOCRT01,TSOCRT02,TSOPRT01,TSOPRT02),        *
               ICOMIDS=(LU040,LU041,LU042,LU043)
         VTIDTAB VTAMIDS=(CICSREG1,CICSREG2),                          *
               ICOMIDS=(CICS1,CICS2)
         VTIDTAB LAST=YES
*
VTSAMPC  CSECT
         LTORG
         END
```

Figure 3. Sample Intercomm Network Table LU6.2 Definitions - Passive Mode (Part 2)

CHAPTER 4

INTERCOMM IN ACTIVE MODE

4.1 INTRODUCTION

The extended (active mode) support for LU6.2 provides for Release 10 of Intercomm to initiate transactions in other VTAM applications. All the comments in Chapter 3 are still valid, but this chapter highlights the areas special to outgoing transactions. It is most likely that in active mode, Intercomm will also be acting in passive mode. Therefore, the dynamic LUNITs allocated to a VTAM application-defining LUNIT must be divided into more than one group. Instead of just having dynamic LUNITs handling the input of transactions, there must also be dynamic LUNITs available over which transactions can be initiated. Such dynamic LUNITs will be on two different chains; the available chain, and the chain for those with a transaction in progress. Further, Intercomm will be acting as the Primary LU on such sessions, implying that these sessions will be "LOGGING ON". They will be known as PLU's (Primary Logical Units).

For active mode support, the new modules VTCMD62 and VTLUDM62 will supercede the original (passive mode) VTCMD6 and VTLUDM6 modules, and two new interface routines will be provided. They are VTPASS62 to handle the queuing of transactions to be sent over an LU6.2 link, and INITLU6 which can be called by a user subsystem to initiate a transaction for sending over an LU6.2 link (see Chapter 5). Updated versions of standard Intercomm modules (particularly the VTAM Front End) allow for use of passive mode alone (with VTCDM6 and VTLUDM6) or for both modes (with VTCDM62, VTLUDM62, VTPASS62 and INITLU6) depending on the purchased LU6.2 support version.

4.2 MESSAGE FLOW

Whenever a transaction has to be "passed" to another application, either by subsystem request via INITLU6, or by input transaction routing, a FECMLU6 (which describes the transaction) will be created and queued to the master LU defining the link. Careful scrutiny is made before the queuing takes place (such as no link yet established) and should the transaction be rejected, message FC018I will be the generated response. (See Appendix B for details of the FC018I message reason codes.) Note that the origin of such a transaction can be from another LU6.2 link.

When the queuing has taken place, a special routine is dispatched, if not already active, to attempt to allocate an available primary session over which the transaction can be sent. When one is found, the FECMLU6 is dequeued from the Master and requeued onto the available PLU's output Q. VTSEND will then get dispatched to accomplish the actual transmission of the message. (The actual message transmitted will not be the FECMLU6, but a generated GDS data stream containing an FMH-5 to indicate the transaction to initiate, and the message text, if any.)

At this point in time, the primary session has been removed from the available chain of PLU's and placed on the in-use chain and is in an LU6.2 conversation state. When all responses to the invoked transaction have been received, the conversation will be ended and the PLU will be returned to the available chain (eligible for a new transaction).

The response(s) on the session, when received, will be routed to either the originating input terminal or to the requesting subsystem, based on information in the saved FECMLU6. The FECMLU6 is then freed before the PLU is rechained on the available queue.

4.3 EXTRA LOGGING POINTS

Extra logging points can occur during the message flow described above, and can be used to analyze the time taken to pass a message out on an LU6.2 link. These logging points can be suppressed by coding ELOG62=NO on the VCT macro for this Intercomm. The new logging points and codes are as follows:

LOG CODE	MSGHTID	TYPE	ORIGIN	SITUATION
21	Originating Input Terminal	HT	FESEND	Transaction has been designated for Passthrough (Terminal Input).
2E	Originating Input Terminal	HT	FESEND	Subsystem has called INITLU6 in order to invoke a transaction on an ISC link. INITLU6 has built a FECMLU6 and called FESEND.
22	Master LU6.2	HT	VTPASS62	FECMLU6 has been queued to the Master Session.
2F	Available PLU	HO	VTPASS62	FECMLU6 has been transferred to an available PLU.
23	Available PLU	HO	VTSEND	A 6.2 Data Stream has been successfully sent across the link using the PLU above.
24	Available PLU	HT	VTCDM62	A response has been received on the PLU as a result of the log 23 output.

Figure 4. LU6.2 Transaction Log Codes

Note that the indication that the transaction has completed is the next F2 log code entry for the originating input terminal, or 01 log code (C1 log code in a Multiregion environment) for the receiving subsystem (as designated in the FECMLU6).

4.4 DEFINING THE INTERCOMM NETWORK

The following sections are an addition to section 3.3 in the previous chapter, in order to define an active mode LU6.2 Intercomm address space. Sample table coding is provided in Figure 5.

4.4.1 The VCT Macro

The SNMAX value must include all the 6.2 sessions (master, primary and secondary) that could be established at any one time, as well as concurrent terminal sessions.

An additional parameter, ELOG62, is provided to specify whether extra logging points as described earlier are to be created on the Intercomm Log. Code ELOG62=NO if you do not want the extra log records. The default specification is YES.

4.4.2 LUNIT Macro to Define the LU6.2 Link

Additional subparameters of the SESSION parameter are provided to define necessary specifications about the sessions to be established during a successful link initiation.

The first subparameter, as in the passive mode, defines the total number of sessions to be established, including the master sessions.

The second subparameter specifies the number of these sessions that should be primary sessions. The default value is 1 (master session only), coinciding with the definition of a passive mode Intercomm. In order for Intercomm to invoke transactions in the defined application, this value must be greater than one.

The third subparameter provides an alternate to the modename for the user (primary/secondary) sessions. The two control sessions will utilize the modename SNASVCMG (as required by VTAM), but the other sessions will utilize this name if specified. The default name is CICSMODE. An entry for each user session modename must appear in the VTAM LM table (see Chapter 6). For two systems to communicate with each other, the same modename value must be defined in each.

The generation of the master LUNIT for the LU6.2 link will include the internal generation of an LUNIT extension (LBX). This becomes necessary as there are now too many fields required that can fit in the LUNIT control block. The LBX control block has the origins of various session chains, a flag byte to specify any special conditions, an ECB for internal use and the name of the logmode (modename) for the user sessions (third subparameter of the SESSION parameter).

A disk queue may be specified, but a reasonably sized core queue should be adequate to hold all the initial queuing requests of transactions to be sent across the LU6.2 link (see Section 4.2).

4.4.3 LUNIT Macro to Define a Pool of Dynamic Sessions

There are no special considerations relating to this pool except that any queuing specification must be sufficient to handle the requirements of an entry as an SLU. If an entry is used as a PLU, only one message will get queued to it at a time.

The requirement to specify that the pool is being defined for LU6.2 use is optional (for compatibility with the original passive support).

The entry count must be sufficient to handle all primary and secondary 6.2 session requests, that is, use the total of the values coded for the total session count (first SESSION subparameter) on each master LUNIT defining an LU6.2 link.

4.5 DEFINING TRANSACTIONS FOR TRANSFER ON AN LU6.2 LINK

A special parameter on the BTVERB macro provides for specification of the application name of the VTAM APPC system to which all input with that verb should be routed. The value of this parameter should be the same as the VTAMID of the LUNIT defining that LU6.2 link. If coded, an index to an applid table, together with an entry in the table, will be generated. The VCT will point to the origin of this table. The name of this parameter is APPLID.

Two special system control commands, LOKA and ULKA are available for the dynamic "locking" and "unlocking" of transactions (verbs) to and from specific applications. To make the LOKA command flexible, every possible VTAM application name for which a LUNIT is coded in the Network Table must be generated in the BTVRBTB table at assembly time, otherwise the command could fail by not finding a matching applid table entry (see Chapter 7). To allow for use of the ULKA command (see Chapter 7), subsystem codes for local processing should be given and should be the same in both systems on Intercomm to Intercomm links, as must be the conversational timeout value (CONV parameter).

If a verb defined to be processed in another system is unlocked from that system (ULKA command) and no SYCTTBL definition for the defined subsystem code exists in the originating system, then an INVALID SUB CODE message (MM101I) is returned to the entering terminal. If the SYCTTBL exists in the originating system, but the subsystem does not exist (or is not available), then an appropriate message is returned to the terminal. Otherwise, the transaction is processed locally. However, if the source of the transaction is a subsystem, and the response is to be returned to the subsystem, then the transaction is not processed locally. Instead, should the INITLU6 caller provide a valid APPLID (one that can be found in the BTVRBTB applid table) for a remote system, then an attempt to forward the message using the requested APPLID will take place. If a valid APPLID for a remote system was not provided by the INITLU6 caller, then an error code will be returned by INITLU6 (see Chapter 5 for further details).

If the other system defined for a verb is unknown or not available, a message is returned to the terminal (see message FC018I in Appendix B), or requesting subsystem (converted to an INITLU6 error return code) as appropriate.

If the transaction can be sent to the other system, then if the partner system cannot process the transaction, it is responsible for returning a reject response (FMH-7) or appropriate error message. If the receiving system is another Intercomm, then if that Intercomm cannot process the transaction, it may return an FC018I message, or may return an FMH-7 response (which is converted to an FC018I message on receipt), or may return an error message (see Chapter 3). As above, an error message response to a subsystem-initiated transaction will be converted by INITLU6 to an error return code, if the response was to be returned to the subsystem.

In any event, a conversational timeout message will eventually prevent a 'hung' condition if no response is generated (see below).

4.6 CONVERSATIONAL TIMEOUT PROCESSING

A CONV parameter should be coded on each user BTVERB which should be at least the value coded for the TCTV parameter on the associated subsystem SYCTTBL macro, so that the subsystem timeout message occurs first as a response.

For transactions received from another system, if CONV=YES is defined on the dynamic LUNIT (or its shared VTCSB macro), then the conversational timeout value coded for the BTVERB macro defining the transaction-id in the receiving Intercomm will be used. If the CONV parameter is not coded on the BTVERB macro, then the TIMEOUT value coded for the shared VTLSB will be used (see Section A.6 and SNA Terminal Support Guide).

For transactions sent to another system, the conversational timeout coded on the BTVERB macro (plus 60 seconds) in the sending Intercomm is used, or a default of 60 seconds is used if the CONV parameter is not coded on the BTVERB macro defining the transaction-id. The additional 60 seconds should provide more than enough time for a timeout response (or FMH-7 or error message as appropriate) to be returned from the receiving system.

Thus, the value coded for the CONV parameter on the BTVERB macro for LU6.2 transaction-ids should be:

- for Intercomm to Intercomm links: the same for both systems and greater than the associated subsystem TCTV value on its SYCTTBL macro
- for Intercomm to 'other system' (CICS) links: should be coordinated with the means of defining processing timeout in the other system.

If Intercomm is the receiving system responsible for a response

to an LU6.2 session transaction, then the standard NO PROGRAM RESPONSE message (FC009I) is returned (unless a subsystem timeout message is sent first) as the timeout response to the transaction. Should the processing application subsystem eventually time out, or complete, the tardy response will be discarded (not queued for the SLU). If Intercomm is the sending system of an LU6.2 transaction, then after a timeout while waiting for a response, message FC109I (see Appendix B) is returned to the originating terminal and/or subsystem (via an INITLU6 error return code), as appropriate, and the LU6.2 session (PLU) is internally disconnected (via SPLU) and a new PLU is acquired (via STLU) for new transaction use.

When a PLU is disconnected via an SPLU, it is first checked for being on the PLU available chain for the link. The timeout will ensure that this is so. The SPLU process will remove the PLU from the available chain and return it to the dynamic pool of LU's. A new LU will be retrieved from this dynamic pool and put on the PLU available chain as part of the STLU processing. This new LU could be the same as the one just returned there, but free LU's are retrieved from the head of the dynamic pool and returned to the end of that pool.

4.7 SUBSYSTEM SYCTTBL MACRO CONSIDERATIONS

For subsystems which may process input LU6.2 transactions, ensure that the value coded for the MNCL parameter is high enough to provide adequate response time on the link. As traffic increases, it may also be necessary to increase the CONV timeout value on the associated BTVERB macro so that a conversational time out does not occur prematurely in the VTAM Front End. For subsystems which invoke LU6.2 transactions, ensure the TCTV time is adequate to allow for error response processing by the INITLU6 interface, to prevent a premature subsystem timeout. For both subsystem types, ensure queuing parameters are large enough to handle increased traffic. Also code REJECT=YES on the SYCTTBL macro.

```
VTAM     TITLE 'VTSAMPC-SAMPLE VTAM NETWORK TABLE WITH CICS'
VTSAMPC  CSECT
*
*   DEFINE THE VTAM CONTROL TABLE
*
         VCT   SECT=CSECT,APPLID=INTERCM1,PASSWD=ICOMPAS1,START=YES,    *
               SNMAX=100,RCVNO=3,RCVRSP=5,RCANYLN=256,ELOG62=YES,       *
               SHUTDTL=50,MXSDTHD=50,TRACE=(EWTO,ETRC),SEQNO=BTAM
*
*   DEFINE THE LOGICAL UNITS AND COMPONENTS  (LUB/LUC PAIRS)
*
*        3270 SDLC (SNA)  CRTS AND PRINTERS
*
         LUNIT NAME=LU040,ALT=LU041,LSB=LS3270S,CSB=CS3270SC,           *
               NUMCL=10,VTID=TSOCRT01                CRT  #1
*
         LUNIT NAME=LU041,ALT=LU040,LSB=LS3270S,CSB=CS3270SC            *
               NUMCL=10,VTID=TSOCRT02                CRT  #2
*
         LUNIT NAME=LU042,LSB=LS3270SP,CSB=CS3270SP,                    *
               DFLN=VTAMQU03,PCEN=20,VTID=TSOPRT01   HARDCOPY #1
*
         LUNIT NAME=LU043,LSB=LS3270SS,CSB=CS3270SP,                    *
               DFNL=VTAMQU03,PCEN=20,VTID=TSOPRT02   HARDCOPY #2(SHARED)
*
*        LU 6.2 SYSTEM LINKS
*
         LUNIT NAME=CICS1,LSB=LSBCICS1,ACQ=YES,SESSION=10,              *
               CSB=CSBCICS,NUMCL=2,VTID=CICSREG1
*
         LUNIT NAME=ICOM2,LSB=LSBICOM2,ACQ=YES,                         *
               SESSION=(10,5,ICOMMODE),                                 *
               CSB=CSBICOM,NUMCL=10,VTID=INTERCM2
*
*        A POOL OF DYNAMIC LUB/LUC PAIRS FOR 6.2 SESSIONS
*
         LUNIT DYN=(25,DL),DFLN=VTAMQU05,PCEN=2,NUMCL=4
*
*   END OF LUB/LUC DEFINITIONS -- PMISTOP AND PCENSCT MUST FOLLOW
*
         PMISTOP
         PCENSCT
*
*   DEFINE THE SPECIFICATION BLOCKS
*
*        3270 SDLC (SNA)  CRTS AND PRINTERS
*
LS3270S  VTLSB LUTYPE=3270S,TRSTBL=LOWTOUP,TIMEOUT=30
*
LS3270SP VTLSB LUTYPE=3270S,TIMEOUT=30,ASRESP=D,SRESP=(D,1)
```

Figure 5. Sample Intercomm Network Table LU6.2 Definitions - Active Mode (Part 1)

```
LS3270SS VTLSB LUTYPE=3270S,TIMEOUT=30,ASRESP=D,SRESP=(D,1),          *
               RELREQ=RELEASE,OUTQ=ACQUIRE,ULVB=SHARE     SHARED LUNIT
*
CS3270SC VTCSB COMPTYP=3270S,CRT=YES,AIDGRP=1,CTCHAR=F5C3114040
*
CS3270SP VTCSB COMPTYP=3270S,CRT=NO,CTCHAR=F54C114040
*
SHARE    VTLVB OTQUEUE=VTUROTX1,SNDNRM=VTURSDX1
*
*        THE 6.2 SESSION LINKS
*
LSBCICS1 VTLSB LUTYPE=CICS,BNDAREA=BIND621,SOUTSEG=1024
*
LSBICOM2 VTLSB LUTYPE=ICOM,BNDAREA=BIND622,SOUTSEG=1024
*
CSBCICS  VTCSB COMPTYP=CICS,CONV=YES
CSBICOM  VTCSB COMPTYP=ICOM,CONV=YES
*
BIND621  VTBIND62 APPLID=INTERCM1,CAPPLID=CICSREG1
*
BIND622  VTBIND62 APPLID=INTERCM1,CAPPLID=INTERCM2
*
*   END OF LOGICAL UNIT AND COMPONENT SPECIFICATION BLOCKS
*
*
*   MISCELLANEOUS TABLES TO COMPLETE DEFINITIONS
*
*
*   1)   TRANSLATION TABLES
*
LOWTOUP  EQU   *
         COPY  TRAN3270
         PMISTOP
*
*   2)   AID GROUP AND AIDDATA DEFINITIONS AND TABLES
*
         AIDGRP 1,CLEAR=1
*
         AIDDATA 1,'RLSE'
         AIDDATA END
*
*   3)   THE VTAM - INTERCOMM SYNONYM TABLE
*
*
*        NOT NEEDED WHEN VTID PARAMETER CODED ON LUNIT MACROS
*
*
VTSAMPC  CSECT
         LTORG
         END
```

Figure 5. Sample Intercomm Network Table LU6.2 Definitions - Active Mode (Part 2)

CHAPTER 5

INITLU6 SUBSYSTEM INTERFACE

5.1 INTRODUCTION

The routine INITLU6 is provided as a callable subroutine for user application subsystems so that they may invoke a transaction across an ISC link. For passing a transaction to another system, facilities are provided by INITLU6 to let the subsystem elect to return the response on the link to:

- the original input terminal, or
- a specified terminal (probably a hardcopy device), or
- the calling subsystem in a caller provided area, or
- the original input terminal and the calling subsystem in a caller provided area.

Note that the original input terminal may be an LU6.2 link. Error codes are returned to the application should there be a problem in invoking the transaction across the ISC link. For COBOL and PL/1 users, entries are provided in ICOMSBS, PENTRY, PLIENTRY and REENTSBS for INITLU6. Refer to the relevant Programmer's Guide for details on use of these facilities. Assembler programs call INITLU6 directly. For dynamically loaded Assembler and PL/1 programs, an entry for INITLU6 is provided in INTLOAD.

In contrast to CICS (see sample program in Chapter 2) only one call (to INITLU6) is made to invoke the transaction. Only one response to the caller is permitted, but multiple responses may be returned to a terminal.

INITLU6 upon a valid request will format a FECMLU6 (see Chapter 4) and call FESEND to pass it to the Front End of Intercomm. The Intercomm Front End LU6.2 interface handles all session allocation and freeing, and the routing of responses. The separation of transaction-id and text is also handled by the Front End of Intercomm. As in CICS, the application program may append the input terminal-id to the text for use by the receiving system. (See Section 2.4).

5.2 CREATING A CONVERSATION

As illustrated in Figure 6, in a single region Intercomm (or a multi-region control region Intercomm) the call to FESEND by INITLU6 will get a direct return code from the Front End relating to the ability to invoke the transaction on the ISC link. Only in cases where a response is requested by the caller in a user provided area, must INITLU6 then wait for that response. In a satellite region however, a

successful call to FESEND simply means that the FECMLU6 has been forwarded to the control region. In such cases, a problem in establishing the transaction on the ISC link is indicated by the control region queuing the FECMLU6 back to the satellite region, with the FECMLU6 containing a return code indicating the reason for failure. If there is no problem, and the caller has not requested that the response be returned to it, then only a FECMLU6 with a successful return code is returned. Thus for satellite regions, INITLU6 must always wait for a response.

When a response (or FECMLU6 with return code) is anticipated, INITLU6 will place the caller subsystem thread in an LU6WAIT condition, similar to the CONVERSE WAIT condition created by a call to CONVERSE (see relevant Programmer's Guide and Operating Reference Manual for details on the use of CONVERSE).

The FECMLU6 and/or anticipated response message is always returned to INITLU6. INITLU6 then returns to the caller with a return code indicating the result of the request or call to FESEND. Thus, a call to INITLU6 is similar to a call to any other Intercomm facility.

Given the difference between satellite region implementation and control/single region implementation, programming design should always assume a conversation wait was required to complete the call to INITLU6.

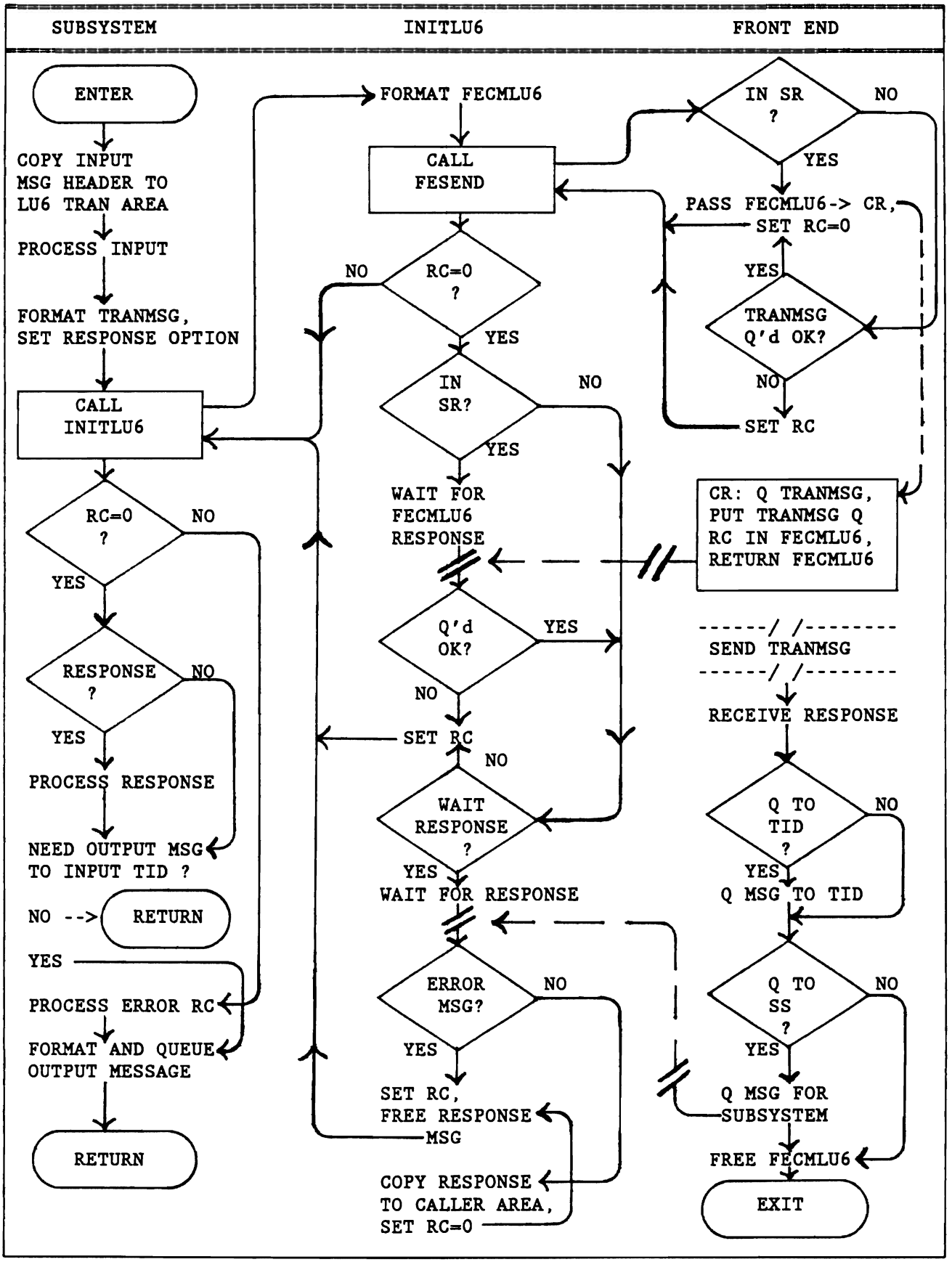

Figure 6. INITLU6 Processing Overview

5.3 PROVIDING A BTVERB ENTRY FOR INVOKED VERBS

It is not a requirement to have a BTVERB entry defined for each transaction-id that may be passed to another system using INITLU6. If an entry is not defined, the eight character area containing an application identifier passed as the third parameter to INITLU6 (see section 5.4) must supply a valid and known application name (APPLID). That is, the supplied name exists in the applid table in BTVRBTB, by having at least one BTVERB entry specifying APPLID=supplied-name. In any event, should an application name be specified in the third parameter, it will override any specification for the defined transaction-id in the BTVRBTB, if any.

Similarly, the conversation timeout value will be taken from the passed second parameter to INITLU6 (see section 5.4) if provided, otherwise a value will be sought from an existing BTVRBTB entry for the transaction. In using the latter value, if found, sixty seconds will be added to it in order to ensure that the remote system, if another Intercomm, can produce an error message first. If no value is defined in the BTVRBTB entry, or no BTVRBTB entry can be found, then a default value of sixty seconds will be used if none provided by the INITLU6 caller in the second parameter (see also section 4.6).

For security reasons, it may be that an installation does not want the APPLID to be passed by the calling subsystem, and that BTVERB entries for each and every transaction across an ISC link must be defined. A user exit, USRLU6I, is provided for this purpose. The exit routine will be passed the same parameters that were passed to INITLU6, and by examining these parameters may modify them based on installation requirements. In particular, the exit may want to force the blanking out of the supplied APPLID, to ensure that only the Front-end defined system, if specified, will be passed the transaction. Options such as routing the response to a specified terminal may be disallowed or forced, etc. Neither the size, nor the location of the transaction message area may be changed. See section 5.8 for a description of USRLU6I. See section 4.5 for further details on providing BTVERB definitions for transaction-ids to be sent across an ISC link.

5.4 PARAMETERS PASSED TO INITLU6

There are three required and one optional parameter to be passed to INITLU6 on each invocation as follows:

- The 'To Be Invoked' Transaction (required) - is the address (label) of an Intercomm type message area containing an Intercomm message header, a four-byte transaction-id and possibly user data (the text). The header should be created by copying the 42-byte Intercomm header of the input message to the sybsystem and then modifying the MSGHLEN field to reflect the size of this transaction (see Programmer's Guides). The text should consist of the transaction-id (verb) followed by any user data that is to be passed for the transaction. The verb and data must be separated by the system separator character (the installation defined field separator character which is used to separate input fields entered on a blank screen). If there is no data to pass, then either specify a MSGHLEN of 46 bytes, or place an EOB (X'26') or an ETX (X'03') following the verb. The verb must be four bytes long to conform to Intercomm standards. This area is not used for the FECMLU6, nor is it freed (may be later reused).

- The LU6CW Control Word (required) - is the address of a one word (four byte) aligned area to pass the processing request to INITLU6, and also to receive a return code.
  The first byte is used to receive a character return code, which for Assembler programs is also returned in register 15 as a hex-value times 4 (unless the return code was alphabetic). See section 5.6.
  The second byte must contain a request code to inform INITLU6 what processing is required, such as whether to return the ISC link response or error message (if received) directly back to a terminal, or to the caller in a user provided area in dynamic working storage (DWS). See section 5.5.
  The third and fourth bytes must contain a value to be used as a time-out value, so that a timeout error message is forced after the defined amount of time elapses. This value should be specified as binary zero (low-values) if a default time is to be used (see section 4.6).

  Upon return from the INITLU6 call, these third and fourth bytes may contain an FESEND return code, should there have been a problem in forwarding the request to, or within, the Front End of Intercomm or it may contain the return code from the user exit USRLU6X (called by INITLU6 to validate the response text, if anticipated and received - see section 5.8). These bytes are only significant if the first byte of the LU6CW contains a 7 or an 8 (see section 5.6). COBOL and PL/1 programs will receive this additional return code as a two byte character field in the third and fourth bytes, but Assembler programs will receive a single byte binary return code in the third byte.

- The Application Name (required) - is the address (label) of an eight character area containing the application identifier (APPLID) of the system to which it is desired to transmit the transaction. It is not necessary to provide the application identifier (see section 5.3), but the eight character area must be supplied and initialized to blanks if there is no APPLID to pass.

- The Response Area or Terminal Name (optional based on request) - the fourth parameter may be required, based on the option request specified in the LU6CW second byte. Should the request be to route any response(s) to a third party terminal, then this parameter must point to a five character field identifying that terminal. Should the request be to return the response back to the caller, then this parameter must point to an area where the response message can be placed (in the program's dynamic working storage area (DWS)). The area should be large enough to hold the anticipated response, otherwise the response will be truncated. The first halfword of this area must be initialized to the length of the area, including itself (as in an Intercomm message header), before the call to INITLU6. The response will be returned following this halfword, and will start with a standard Intercomm message header from which the actual response length can be determined, even if the response was truncated (see description of return code 5 in section 5.6). The response message header will be similar to that passed in the first parameter on the call to INITLU6 (and used as the basis for formatting the FECMLU6 - see section 5.7). The first two bytes (MSGHLEN) will contain the length of the response text plus 42 (header length). The terminal-id (in MSGHTID) and BMN number (MSGHBMN) will be the same as in the original input message to the subsystem (set by INITLU6).

5.5 OPTIONS FOR INVOKING INITLU6

The LU6CW processing option byte (byte 2) must contain one of the following request codes on a CALL to INITLU6, and based on that request code the fourth parameter may be needed on the CALL.

Value of Request Code	Meaning	Fourth Parameter Requirements
0 (binary zero) or blank	Return the response to the caller only	Point to an area in DWS which is large enough to hold any anticipated response.
Q	Return the response only to the terminal originating the input message to the calling subsystem	(not required)
T	Return the response only to a specified terminal	Point to a five character field containing the terminal-id.
B	Combine options 0 (zero) and Q. That is, return the response to both the originating input terminal and to the caller.	Point to an area in DWS which is large enough to hold any anticipated response.

5.6 RETURN CODES FROM CALLS TO INITLU6

The following is a list of return codes given as a one byte character field in the first byte of the LU6CW following a call to INITLU6. Return codes are also available in register 15 for Assembler Language programs.

Return Code	R15	Meaning
0	00	The request was successfully processed
1	04	The request was processed, however there was no response to the invoked transaction. A local FC109I message, or a remote Intercomm FC009I message was trapped by INITLU6 as the response to the transaction.
2	08	An error condition occurred on processing the invoked transaction in a remote Intercomm. A TCTV timeout or a program check occurred. Message MS009I or MP001I or an error message generated by PMICANC was trapped by INITLU6 as the response to the transaction.
3	12	The transaction was sent to a remote Intercomm, but the remote subsystem to be invoked had been delayed or a queuing error occurred. Message MM098I, MM099I, MM101I, MM103I or MM104I was trapped by INITLU6 as the response to the transaction.
4	16	No core was available to create a FECMLU6 or an LU6WAIT conversation control block.
5	20	The response received from the remote system was larger than the area provided for the fourth parameter of this call. The response was truncated. For follow up action the actual length of the response is returned in the message header length field (MSGHLEN) for analysis.
6	24	An invalid combination of parameters and/or options were specified on the call to INITLU6. Or, user exit USRLU6I rejected the parameters.

Return Code	R15	Meaning
7	28	A non-zero return code was given to INITLU6 upon its CALL to FESEND to forward the built FECMLU6 for processing. FESEND's return code will be placed in byte 3 (Assembler) or bytes 3 and 4 of the LU6CW. The FESEND return codes are as documented for FESEND together with a new code of 44. This new code is returned if the APPLID parameter (third parameter passed to INITLU6) was not valid. Either none was given and a BTVERB entry for the transaction was not found, or the entry did not point to an APPLID. Or, the supplied APPLID was not defined as specifying a remote ISC link or it defined this Intercomm. FESEND return code 44 may also be produced if the user exit USRLU6I was invoked to override any provided APPLID. (See sections 5.3 and 5.8).
8	32	The user exit USRLU6X, which is invoked by INITLU6 before passing a response back to the subsystem, indicated that the response was invalid by a non-zero return code to INITLU6. This return code will be placed in byte 3 (Assembler) or bytes 3 and 4 (COBOL or PL/1) of the LU6CW. See also section 5.8.
A thru N	A thru N	There was a problem initiating the transaction. The alphabetic return codes have the same meaning as that documented in the FC018I message (see Appendix B). If the value is K then the subsystem may wish to CALL MSGCOL or COBPUT, passing the message given in the first parameter of the CALL to INITLU6, so that the transaction can be processed locally. (No response to the subsystem will be available however).

5.7 PROGRAMMING AND LANGUAGE DEPENDENT CONSIDERATIONS

Each programming language requires a subroutine CALL in a particular format in order to maintain subsystem reentrancy. Languages COBOL, PL/1 and Assembler are individually discussed in the next sections.

NOTE:

- All passed parameters must be in the caller's dynamic working storage area (24-Amode area) if the calling subsystem may be dynamically loaded (above or below the 16M line). See the appropriate Programmer's Guide for dynamic loaded and XA/ESA programming considerations.

- The original input message to the subsystem is not freed by INITLU6. It is still available to (addressable by) the subsystem on return from INITLU6 unless the subsystem (or MMU) freed the message before the INITLU6 call.

- Unexpected responses received by INITLU6 (or the Subsystem Controller) are cancelled (log code FD).

- The message header in the transaction message area is copied by INITLU6 to use as the basis for the header of the FECMLU6 queued to FESEND, however, the header is not passed across the ISC link. Therefore, any user data in the header (such as MSGHTID, MSGHUSR or MSGHVMI) must be copied to an area in the transaction message text for subsequent retrieval by the receiving system or program. The data may be inserted in the text either by the application program, or if common to all transactions then by the USROTEDT Intercomm FESEND user exit (see Chapter 3). If the receiving system is another Intercomm, then either the Intercomm INQUEUE exit (see Chapter 3) or the receiving application program can restore the values to the input transaction message header, or process the passed values, as appropriate.

5.7.1 COBOL Programs

The CALL to INITLU6 must take place from within the main routine of the subsystem and not a called subroutine. The INITLU6 field may be in Working Storage. The format of the CALL is as follows:

```
CALL 'COBREENT' USING INITLU6, TRAN-MSG, LU6CW, APPLID[, (TERMID)].
                                                        (RAREA )
```

where INITLU6 is a field defined in COPY member ICOMSBS as:

```
    PIC 999 COMP VALUE 103.
```

LU6CW should be defined in the DWS as follows:

```
02  FORCE-ALIGN                PIC 9(8) COMP SYNC.
02  LU6CW REDEFINES FORCE-ALIGN.
    04  LU6RC                  PIC X.
    04  LU6OPT                 PIC X.
    04  LU6TIM                 PIC S9(4) COMP.
    04  LU6FERC REDEFINES LU6TIM.
        06  FERC               PIC XX.
    04  LU6USRC REDEFINES LU6TIM.
        06  USRC               PIC XX.
```

The APPLID field should be defined as an eight character field, and set to blanks should an APPLID value not want to be passed.

The TERMID field, if required, should be defined as a five character field and specify the terminal-id to which the response should be routed in cases where option T is used.

The RAREA field should specify an area to hold the response message, if expected. The first 2 bytes of the RAREA field must contain a halfword value equal to the length of the area provided. The RAREA field must be defined in dynamic working storage (DWS) as follows:

```
02  RAREA                      PIC X(desired size).
02  RLENX REDEFINES RAREA.
    04  RLEN                   PIC S9(4) COMP.
    04  RESPONSE               PIC X(desired size-2)
```

In Working Storage define a field:

```
01  XLEN                       PIC S9(4) COMP VALUE desired size.
```

and in the program's PROCEDURE DIVISION:

```
    MOVE XLEN TO RLEN.
```

before the CALL to INITLU6.

The RESPONSE field in the DWS may be further subdefined as it will always consist of an Intercomm message header followed by text. The MSGHLEN field of the message header may need to be examined by the program to determine the text length.

5.7.2 PL/1 Programs

As with COBOL, the CALL to INITLU6 must take place from within the MAIN procedure of the subsystem and not from a called subroutine. The format of the CALL is as follows:

```
CALL  INITLU6(TRANMSG,LU6CW,APPLID[,{TERMID}]);
                                    {RAREA }

                          or

CALL  PMIPLI(INITLU6,TRANMSG,LU6CW,APPLID[,{TERMID}]);
                                           {RAREA }
```

depending on whether PLIENTRY or PENTRY, respectively, is being used by the PL/1 program. In either case, INITLU6 is suitably defined in the respective %INCLUDE COPY member.

The LU6CW should be defined in the DSA as a word-aligned structure as follows:

```
DCL     1 LU6CW ALIGNED,
          2 LU6RC     CHAR(1),
          2 LU6OPT    CHAR(1),
          2 LU6TIM    FIXED BIN(15),
        1 LU6FERC DEF LU6CW.LU6TIM,
            4 FERC    CHAR(2),
        1 LU6USRC DEF LU6CW.LU6TIM,
            4 USRC    CHAR(2);
```

The RAREA field in the DSA must be initialized, if used, before a CALL to INITLU6 by putting the length of the field in the first halfword. As RAREA will contain an Intercomm message, the first 42 bytes beyond the initial length field will contain a message header and can be defined accordingly. For example:

```
DCL     1 RAREA       CHAR(desired size),
          3 LEN       FIXED BIN(15),
          3 HDR,
            5 MSGHLEN FIXED BIN(15),
            .
            .
              other header fields
            .
            .
          3 DATA      CHAR(desired size-44);
```

Before the CALL to INITLU6 initialize LEN to the desired size, and upon return the MSGHLEN field can be examined for the amount of data actually returned.

5.7.3 Assembler Programs

The format of the CALL to INITLU6 is as follows:

```
CALL    INITLU6,(TRANMSG,LU6CW,APPLID[,(TERMID)]),VL,MF=(E,list)
                                        (RAREA )
```

Register 15 may be preloaded with the address of INITLU6, by taking the address out of the SPA using field SPAILU6 if the program is not loaded above the 16M line, or by using a VCON of INITLU6. Using the INITLU6 VCON will cause a branch into INTLOAD if a dynamically loaded subsystem is linked with it (required if loaded above the 16M line); see Assembler Language Programmer's Guide.

Upon return, as well as the return code in byte one of the LU6CW, register 15 will contain a return code (see section 5.6). The first halfword of RAREA must be initialized to the length of RAREA before any CALL to INITLU6 is made passing RAREA as the fourth parameter.

The subsystem is responsible for freeing the original input message.

5.8 INITLU6 USER EXITS

Two user exits are conditionally called (by use of CALLIF) within INITLU6 in order for an installation to enforce certain standards in the ISC transaction invocation environment. These user exits, while possibly modifying the data passed to them, must not change the addresses of the parameters, or the parameter list, that they have been passed. These user exits may not give up control to the Intercomm Dispatcher, directly or indirectly (via a File Handler call for example). The exits must use standard linkage conventions with a local save area (must be serially reusable).

The two user exits are:

- USRLU6I which is called upon every INITLU6 call where the parameter list has at least three parameters.
- USRLU6X which is called when a response has been received for placing in the response area provided by the caller.

These user exits will be discussed individually in the subsequent sections.

5.8.1 The USRLU6I User Exit

This user exit gets called, if provided, once INITLU6 determines that a possible valid parameter list has been passed to it. The parameters passed are the same as those passed to INITLU6. That is, register 1 upon entry to USRLU6I will be identical to the register 1 upon entry to INITLU6 (will point to passed parameter list). The parameter list addresses, and register 1, must not be modified. The contents of the passed parameters may be modified in order to maintain some company standard. For example, the supplied APPLID parameter value may be interogated and modified based upon user installation criteria. If routing of a response to a third party terminal is requested, then the ID of this terminal can be inspected, and checked for certain values, etc. If after inspecting and possibly modifying the values of the passed parameters, and it is deemed correct for the invocation of the transaction to continue, then the exit must return to INITLU6 with a return code of 0 in register 15. If the return code (contents of register 15) is not zero, then INITLU6 will return to its caller with an LU6CW return code of 6 (Assembler R15 return code of 24) indicating an invalid parameter list was passed to INITLU6.

5.8.2 The USRLU6X User Exit

This user exit gets called, if provided, once a response for an outstanding request has been received. At entry, register 1 points to a parameter list containing one address. The parameter passed to this exit is the address of the response message (header and text). This address must not be altered, but upon review, the contents of the message text may be modified. It should be noted that a response may be an Intercomm error message if the ISC link was to another Intercomm. Neither a FECMLU6 response text, nor an error message generated via PMICANC in a remote Intercomm is passed to USRLU6X.

Some of the Intercomm error messages are trapped by INITLU6 and produce return codes of 1, 2 or 3 to the INITLU6 calling program (Assembler R15 return codes of 4, 8, 12) and are not passed to the user exit. The Intercomm error message ids and associated return codes that are checked for by INITLU6 can be found at the rear of the listing of INITLU6, at label MSGTBL. The user may wish to add other message ids to this table if desired, however return codes other than 4, 8 or 12 (0C) have other meanings, as defined in section 5.6. If USRLU6X returns a non-zero return code to INITLU6 in register 15, the response message is discarded, and return is made to the INITLU6 caller with a return code 8 (Assembler R15 return code 32), plus the USRLU6X return code in the low-order bytes of the passed LU6CW (see section 5.6). If the response (or error) message is to be returned to the INITLU6 caller, then a return code of 0 in register 15 must be used.

Chapter 6

VTAM SESSION PARAMETERS

Until LU6.2 support existed, Intercomm as a VTAM application has always been considered the primary logical unit (PLU) in any session established between itself and a terminal. (The terminal being the secondary logical unit or SLU.) In such an environment, any "Logoff" of the terminal from Intercomm was processed through an SPLU command request or by VTAM notifying the LOSTERM exit in Intercomm.

In the LU6.2 environment, two VTAM applications (or logical units) can have more than one session between them. Some of these sessions may have one VTAM application as the PLU and some sessions have the other VTAM application as the PLU. (Up until LU6 protocol introduction, the only way to achieve more than one session between two VTAM applications was to open more than one ACB.)

Under LU6.2 protocol, in order to support multiple parallel VTAM application sessions, a requirement, PARSESS=YES must be coded on each APPL statement defining the applications in the link to VTAM. SONSCIP=YES is not required by Intercomm, but is required by CICS. Also, the EAS session count will need to be increased for both systems in a link to add the LU6.2 sessions.

Each VTAM application in a link also requires the authority to ACQUIRE sessions. If this authority is not presently allowed, it should be granted by coding AUTH=ACQ on each APPL statement. Intercomm will want to acquire all 6.2 sessions in which it will act as the PLU. Also, APPC=NO should be coded on the APPL statements for any Intercomm or CICS in the link as they do not currently use the new ACF/VTAM 3.2 (or higher) LU6.2 APPCCMD macro or APPC/MVS services for LU6.2 communication.

At least two Logon Mode Table (LM Table) entries defined via MODEENT macros in the VTAM region are required for session establishment. The first is used for the master sessions in any LU6.2 link and requires a mode name of SNASVCMG. A default version of this entry, which is useable, is provided in ACF/VTAM Version 3.2 and higher (Intercomm's master session uses this modename as it is the Services Manager Session.) The others are used for all other sessions with a modename of the user's choice but defaults to CICSMODE, which may be changed by the user as described in the previous chapters. In any one link, both partners must use the same modename for establishing their user sessions. See also IBM's ACF/VTAM Programming for LU6.2 and Customization.

Intercomm messages relating to session problems are described in Appendix B.

Chapter 7

SYSTEM COMMANDS

Intercomm Back End control and processing commands such as LOAD, MMUC, FILE, PAGE and SAVE which output a single response may be entered across an LU6.2 session. Statistics commands such as FHST and TALY must be used with caution because multi-page output after the first message may be discarded in the Intercomm Front End (depending on how quickly the second and subsequent response messages are queued for the sending LU (SLU).

Intercomm Security (ESS and Basic) and Multiregion (LOKR, ULKR, COMM) commands may not be used.

Front End commands are meaningless and inoperative for terminals logged on to the other system. The following commands may not be used: COPY, FLSH, LOCK, LTRC, RLSE, SPLU, STLU, SWCH, UNLK, VTCN, VTST. Nor may WHOI or WHOU be used under Release 10.

A dynamic session may not be stopped or started, even from an Intercomm terminal. However, stopping or starting of the master (control) session may be requested from the Intercomm control terminal. Stopping the master session will disconnect communication with the other system, while starting the master session will reestablish communication with the associated system.

TALY$FE and VTST terminal status displays can be used from Intercomm terminals to display master, primary and secondary session status. Under Release 10, a VTST$ALU6 command can be used to display only the LU6.2 session LUs (see also Section A.4.2).

Two new commands are available with the extended LU6.2 interface, LOKA and ULKA. LOKA can be used to force input of a particular (application-program associated) transaction to be sent over a 6.2 link to another system. The ULKA command removes this status, whether established by the LOKA command, or by specification of the APPLID parameter on the associated BTVERB macro (see Section 4.5). The form of the two commands are as follows:

```
LOKA$vvvv$aaaaaaaa
ULKA$vvvv
```

where vvvv specifies a verb defined in the BTVRBTB which is <u>not</u> a Front End verb, and aaaaaaaa specifies the 1 to 8-character VTAM application name of the system to process all transactions entered with verb vvvv.

The LOKA command overrides any previous VTAM application name (APPLID) designation for the verb, while ULKA resets the verb for local processing (does not restore any previous designation).

The application system aaaaaaaa should already have a 6.2 link established for it, otherwise, any attempt at entering verb vvvv will be rejected. Further, to ensure that aaaaaaaa is predefined in the (internally generated) APPLNAME table (in the BTVRBTB), a verb, dummy or real, must be coded in the BTVRBTB (or COPY member USRBTVRB) via a BTVERB macro specifying APPLID=aaaaaaaa.

Note that processing of the LOKA command cannot detect whether a loop condition is being set up by having INTERCMA, for example, specify INTERCMB to process verb ABCD, while INTERCMB has this same transaction-id defined to be processed in INTERCMA (see message FC018I, reason code K, in Appendix B). If an INTERCMC was involved in the loop, then the situation cannot be detected at all.

Appendix A

INTERCOMM MACROS

A.1 ICOMGEN

A new option for the VTAM parameter has been provided, whereby the LUTYPE list can now include the type CICS or ICOM as appropriate. If included in the LUTYPE list, then JOB14 (Release 9) or JOB15 (Release 10) of the installation JCL stream will include assemblies of the two LU6.2 dependent modules VTLUDM6 and VTCDM6. Under Release 10, coding LU62=YES or PASSIVE will also generate JOB15. If LU62=ACTIVE is coded, then JOB15 will create assemblies of the extended LU6.2 dependent modules VTLUDM62 and VTCDM62, together with modules VTPASS62 and INITLU6. Note that VTLUDM62 and VTCDM62 can be used for passive mode processing.

A.2 ICOMLINK

As in the case of the ICOMGEN macro, the VTAM operand of this macro can now specify an LUTYPE of CICS or ICOM, and if coded, INCLUDE statements for the modules VTLUDM6 and VTCDM6, will be part of the generated Linkedit deck. Under Release 10, coding LU62=YES or PASSIVE will also generate the INCLUDE statements. For Release 10, coding LU62=ACTIVE will ensure that the extended LU6.2 dependent modules VTLUDM62 and VTCDM62, together with modules VTPASS62 and INITLU6 will be included in the generated linkedit deck.

A.3 VCT

SEQNO=BTAM is required. If no BTAM modules are included in the linkedit, ensure the BTAM global is set to 0 in SETGLOBE before assembling the Network Table. Under Release 10, for extended LU6.2 support only, a new parameter ELOG62 is added to specify whether or not logging of message activity for primary sessions is desired (see Chapter 4). The default is YES. Code ELOG62=NO to suppress the additional logging. If the SNMAX parameter is coded on the existing VCT, it may need to be increased to include all the dynamic sessions (primary and secondary) as well as the master session for each LU6.2 link (see Chapters 3 and 4).

A.4 LUNIT/LCOMP

As illustrated in Figure 3 (passive processing - Release 9) or Figure 4 (Release 10 - passive and/or active processing), special LUNIT definitions (which internally generate LCOMP definitions, that is, a LUB and LUC) for the master (control) LU for each LU6.2 link and for the pool of dynamic LUs (used as PLUs and SLUs for all the links) are required. Note that the number of dynamic LUs must include one to be used as an SLU for each partner system's master session.

A.4.1 Defining a Master LUNIT

A master session LUNIT macro must be coded for each LU6.2 link to be established with another system (VTAM application). Only one link may be established with a specific system from a specific system. Each specific link must also be defined in the partner's system. The master LUNIT must be given a unique 5-character Intercomm (local) NAME, which must be corresponded to the VTAM-id (APPLID) of the other system via the VTIDTAB macro (Release 9) in the VTIDTABL, or via the VTID parameter on the LUNIT macro (Release 10). If the other system's APPLID is exactly 5 characters and is unique, it may be used as the Intercomm NAME on the LUNIT, no VTAM-id is then needed. UPINTV may be coded on the master LUNIT for automatic reconnect if the link is lost. If ACQ=YES is coded, the link with the other system is established during VTAM startup. Otherwise, a STLU for the master LUNIT (include the ACQ option) must be entered to establish the sessions after Intercomm startup completes (unless the partner system is given responsibliity for always establishing the link). Messages (see Appendix B - FC134-141I) are issued as the sessions are established (or rejected).

Each link may have up to 255 parallel sessions (including the master LUs). Each dynamic PLU (corresponding to an SLU in the other system) and each dynamic SLU (corresponding to a PLU in the partner system) counts separately as a session in each system. To define the sessions, a SESSION parameter (for LU6.2 support only) must be coded on the master LUNIT. The SESSION parameter has 3 subparameters in the following form:

```
SESSION=(total[,{#plu's}[,{modename}]])
                {1      }  {CICSMODE}
```

where: total is the maximum total parallel sessions (from 3 to 255) that may be established on the link including the master, the partner's master, and all other primary and secondary sessions.

#plu's is the maximum of the total sessions that can be used by this system as primary sessions (including 1 for the master LU) if extended support is installed. Under extended (active mode) support, this number must be less than total. The default is 1 (master LU only) for basic support.

modename is the user-assigned VTAM modename to be used for establishing sessions on this link (extended support). The default is CICSMODE (basic support). (See Chapters 1 and 6.)

On each side of the link, the SESSION count parameters (or other system equivalent) should match, else the smaller value will be used. The modename must be the same for both partners in the link.

All other standard LUNIT parameters must be defined (LSB and CSB) along with NUMCL, which may be as small as 2. For active mode, disk queuing for overflow should also be defined, or increase NUMCL according to expected traffic (see also Chapter 4).

A.4.2 Defining the Dynamic LUNITs

One pool of dynamic LUs (for all PLUs and SLUs) is defined for all the links and the size of the pool must be at least as large as the sum of the 'total' subparameter for each SESSION parameter on each master LUNIT. As sessions are established across a link, an available (not in use) entry (LUB/LUC pair) from the dynamic pool is chained to the master LUB and its LSB and CSB pointers are set to those of the associated master LUNIT. The assignment of dynamic LUs to specific masters and their use (as SLU or PLU) is random depending on interaction during link establishment with the other system(s). Under Release 10, actual assignment can be determined from a VTST command display: the REGION column gives the VTAM-id (APPLID) of the partner system if the device TYPE is ICOM or CICS (as defined for the master LUNIT), and the LUNIT is currently chained to a master LUB.

Each dynamic session will utilize a LUB/LUC pair, and to create these, a special version of the LUNIT macro is coded at the end of all other LUNIT/LCOMP macros. The form of the macro is as follows:

```
LUNIT   DYN=(nnn,LU6,cc),...other LCOMP parameters (Release 9 or 10)

LUNIT   DYN=(nnn,cc),...other LCOMP parameters     (Release 10 only)
```

where:

nnn is the number of LUNIT/LCOMP pairs to generate (up to 999).

cc is the two character alphabetic prefix of the generated LUCNAME for each pair. The default is DL.

The names of the pairs will be of the form cc001 - ccnnn.

The LUNIT parameters defining queuing specifications are required, leaving all other parameters to default, or be defaulted to those on the master LUB/LUC pair's specifications blocks. The queuing specifications should ensure that NUMCL is at least 2, and possibly larger if multiple responses are likely to a transaction invoked within this Intercomm. Disk queuing may be used to handle the multiple responses from a transaction instead of extra core queue entries, but if disk queuing is specified, make sure the PCEN parameter is small enough to handle each and every generated LUB/LUC pair (see section 3.3.3). RESTART=NO can be coded on the shared VTCSB. AIDGRP, CRT, LOCK, NAME, PRYMSGS, and UPINTV are not applicable or not supported for LU6.2 sessions. The CONV parameter for conversational terminal processing of verbs defined with a conversational time-out can be coded on the dynamic LUNIT or the shared VTCSB (see also the VTLSB macro).

A.5 VTCSB

The COMPTYP parameter must specify CICS or ICOM on the CSB pointed to by the master LUNIT. RESTART=NO must be coded, even if message restart is used for Intercomm-owned terminals. The CTCHAR parameter should be coded if output messages are not fully formatted by MMU, and all input terminals are 3270 CRTs. The CONV=YES parameter may be coded (if not on the dynamic sessions LUNIT).

A.6 VTLSB

The LUTYPE parameter must specify CICS or ICOM on the LSB pointed to by the master LUNIT. The BNDAREA parameter must be coded (see below). Most other parameters are not applicable. Translation must be executed in the transaction-originating system, if needed. A conversational TIMEOUT value may be specified, and is used on secondary sessions if CONV=NO on the VTCSB (see above). The timeout message is sent as a response to the other system, and the transaction is terminated. (See Chapter 4, section 6 for further details.) Also, the SOUTSEG parm should be used to specify the maximum output message length that can be sent across the link in a single Request Unit (for example, 1024). A message of greater length will be split (chained) into more than one Request Unit. Should the SOUTSEG value be greater than the BUFFER and RUSIZE parameters on the CICS DFHTCT TYPE=MODESET definition (see Chapter 2), then the lower value from CICS will be used, possibly causing more Request Unit chaining (depending on average output message length) and potentially slowing response time. Refer to Section 3.3.4 for further details. For Intercomm to Intercomm links, the SOUTSEG value should be the same in each system.

A.7 VTBIND62

This macro is used to create a bindarea for Intercomm to use when establishing its master session with the partner system in the LU6.2 link. The form of the macro is as follows:

```
symbol  VTBIND62    APPLID=icom-name,CAPPLID=osys-name
```

where:

APPLID — is the VTAM name of Intercomm as defined to VTAM (and on the VCT).

CAPPLID — is the VTAM name of the other system as defined to VTAM and for the associated master LUNIT.

symbol — is the macro label pointed to by the BNDAREA parameter on the VTLSB macro pointed to by the LSB parameter on the master LUNIT defining this link.

A.8 VTLVB

If an INQUEUE user exit is coded (see Chapter 3), specify its name via the INQUEUE parameter coded on this macro, which must be pointed to on the shared VTLSB macro by the ULVB parameter. An additional user exit (SCIP) is available for LU6.2 sessions and is called whenever a new secondary session is about to be established. The exit name is defined on the SCIP parameter of the VTLVB macro. The SCIP user exit may reject the new session via a non-zero return code in register 15. The three-word parameter list (address passed via register 1) contains: 1) LUB address for new session, 2) NIB address, and 3) bindarea address. Note that the bindarea contains the modename used for the session establishment.

If a user exit routine is desired to be given control when a primary session is established (including the master session), then this routine should be pointed to by the LOGON parameter of this macro. Note that if the user has a VTUSLGNX exit routine present, it will get invoked for each 6.2 session established in primary mode (that is, when it is "logged on").

See SNA Terminal Support Guide for furthur details on VTAM user exits.

A.9 BTVERB

A BTVERB macro must be defined in the Intercomm Front End Verb Table (BTVRBTB, or its COPY member USRBTVRB) for every transaction-id (verb) that will be passed or received on a LU6.2 link. Required additional parameters are given in Chapters 3 and 4.

Appendix B

ERROR MESSAGES

FC018I PASS OF VERB vvvv DEFINED TO BE PROCESSED IN APPLID aaaaaaaa FAILED - REASON CODE r

FEMSG (VTRECVE/VTLUDM62/VTCDM62)

The transaction with verb vvvv was defined to be processed in application aaaaaaaa, either by coding the APPLID parameter on the BTVERB macro, or by use of the LOKA command, or by subsystem request via the INITLU6 interface. The pass of the associated transaction to application aaaaaaaa failed for the reason (r) defined below:

A The interface module VTPASS62 was not included in the Intercomm linkedit, making it impossible to do any ISC queuing.

B The APPLID aaaaaaaa did not define a valid VTAMID, for which there was a corresponding ICOMID, in the network definition tables.

C Although a valid ICOMID was found for VTAMID aaaaaaaa, no master LUB/LUC pair with that ICOMID was located (EXTERM error).

D The located master LUB/LUC pair did not have a CSB defined.

E The CSB did not define a valid LU6.2 link (COMPTYP not CICS or ICOM).

F The LU6.2 link to APPLID aaaaaaaa has not been, or is no longer, established.

G There is no LUB extension (LBX) defined for the master LU6.2 link LUNIT: possibly assembled the network table using old versions of the LUNIT/LCOMP macros.

H The LUB extension pointer in the master LUB does not point to a valid LUB extension: possible core clobber.

I There are no PLU's available, or likely to become available, in order to initiate this new conversation.

J A SHUTD or HALT of the LU6.2 link has been initiated. No new conversations (including this one) will be established.

K A loop condition has been detected by the Intercomm partner of the link. That is, Intercomm system aaaaaaaa has transaction vvvv defined to be processed in this originating system. Note: this reason code is unique in that it is the receiving Intercomm system in the link that is producing this message and not the originating system.

L The created FECMLU6 (describing the transaction) was not successfully queued to the master LUNIT of the LU6.2 link. FESEND came back with a non-zero return code, probably indicating that the queue was full. (Look for message MG600I to the control terminal.)

M The receiving system has initiated closedown, and has requested that no new conversations should be initiated.

N An FHM-7 response was received saying there was a problem initiating the transaction. (See message VT052I to the control terminal giving the FMH-7 sense codes.)

FC109I ********** NO RESPONSE TO TRANSACTION ON ISC LINK **********

VTPASS62

After a period of time equal to the conversational timeout value defined on the BTVERB macro for the transaction-id (plus sixty seconds), a response has not been received across the LU6.2 session from the processing system. The session over which the transaction was sent is terminated and a new one established to clear out any outstanding conditions.

For an Intercomm to Intercomm LU6.2 link, the extra sixty seconds (added to the timeout value coded on the BTVERB macro CONV parameter in the sending system) allows for either the NO PROGRAM RESPONSE message (FC009I), or the subsystem TCTV TIMEOUT message (MS009I), or the PROGRAM CANCELLED DUE TO TIME-OUT message (from PMICANC), to have a fair chance of being returned on the session if there is a problem in the receiving Intercomm system.

If the receiving system is not Intercomm, then this message (and timeout processing in the sending Intercomm system) prevents a permanent 'hung' condition for the LU6.2 session and the transaction originator.

FC134I SESSION REJECTED FOR APPLICATION aaaaaaaa,REASON CODE=r

FEMSG (VTEXITS)

A request to establish a session in which Intercomm will act as a Secondary Logical Unit (SLU) has been received, but could not be accepted from VTAM application aaaaaaaa due to the reason code (r) specified below:

0. Unknown VTAM application (aaaaaaaa). No LUNIT defined, or entry not placed in VTIDTABL correctly.

1. The LUNIT defining aaaaaaaa does not indicate a primary LU6.2 session (SESSION parameter omitted, or VTLSB type not CICS or ICOM).

2. The master LUNIT has been deactivated, implying no new sessions can be initiated.

3. There are no available LUNIT's in the dynamic pool. (Too many active sessions already, or the size of the pool needs to be increased.)

4. The VCT SNMAX count had already been attained, therefore no new sessions can be established.

5. User SCIP exit rejected a session establishment request.

6. An OPNSEC macro failed, or VTAM is inactive. (A VTERR with ID of 26 is also issued, see message VT025I.)

7. There is a temporary storage shortage.

A. The presented BIND parameters do not indicate a LU6.2 session.

B. The maximum session count for the master LU has already been attained.

C. No new sessions may be accepted, per control operator command request. (Not yet implemented.)

D. Sessions are being quiesced, and so no new ones can be accepted. (Intercomm in closedown, for example.)

E. The presented BIND parameters, despite indicating a LU6.2 session, are not acceptable.

FC135I SESSION BIND WITH APPLICATION aaaaaaaa FAILED, ERROR CODE=yy,zz

FEMSG (VTEXITS)

An attempt to accept a secondary session with application aaaaaaaa failed on the OPNSEC macro. yy,zz are the RPL RTNCD and FDBK2 field values respectively.

FC136I SESSION ESTABLISHED WITH APPLICATION aaaaaaaa USING yyyyy. SECONDARY SESSIONS NOW nn (mm)

FEMSG (VTEXITS)

An attempt to accept a session with application aaaaaaaa was successful. The LUNIT yyyyy was used from the dynamic pool to represent the secondary session, and the total number of secondary sessions with application aaaaaaaa is now nn, while the total number of sessions (secondary and primary) is now mm.

FC137I SESSION TERMINATED WTIH APPLICATION aaaaaaaa USING yyyyy. SECONDARY SESSIONS NOW nn (mm)

FEMSG (VTEXITS)

A session for which an FC136I message had already been issued has now terminated. The LUNIT yyyyy representing the session has been returned to the dynamic pool, and the total number of secondary sessions with application aaaaaaaa is now nn, while the total number of sessions (secondary and primary) is now mm.

FC138I LOGON ACCEPTED FOR APPLICATION aaaaaaaa, USING yyyyy. PRIMARY SESSIONS NOW nn (mm)

A successful LOGON of a 6.2 primary session has taken place with application aaaaaaaa. The logical unit yyyyy will be used to represent the session. Should yyyyy be the name of the master LUNIT defining the 6.2 link, this message would have been preceded by the standard LOGON ACCEPTED message (FC140I). (Message FC140I is suppressed for all dynamic logical units logging on.) The number of primary sessions with application aaaaaaaa is now nn, while the total number of sessions (primary and secondary) is now mm.

FC139I PRIMARY SESSION yyyyy FOR APPLICATION aaaaaaaa DISCONNECTED. PRIMARY SESSIONS NOW nn (mm)

A termination of a primary session for application aaaaaaaa has taken place. The LUNIT yyyyy is now available. Should yyyyy represent the name of the master LUNIT defining the 6.2 link this message will be followed by the standard LU DISCONNECTED message (FC121I). (Message FC121I is suppressed for all dynamic logical units logging off.) The number of primary sessions with application aaaaaaaa is now nn, while the total number of sessions (primary and secondary) is now mm.

FC141I LOGON REJECTED FOR LU aaaaaaaa(xxxxx),REASON CODE=r

FEMSG (VTEXITS)

aaaaaaaa is the VTAM name of the partner system and xxxxx is the logical unit (LU) name, r is the code indicating the reason for rejection. The following are LU6.2 reason codes:

F) VTLUDM6/VTLUDM62 logon exit invoked for a non-LU6.2 LUNIT.

G) LUNIT has been defined with more than one component. This is illegal for a LU6.2 LUNIT.

H) A secondary logical unit is trying to act as a primary session. This is not supported.

I) The modename for the Master Session was not given as 'SNASVCMG' or a dynamic LU is trying to LOGON using a modename of 'SNASVCMG'.

J) The maximum primary session count for the LU6.2 link has already been attained.

MG600I FESEND EP=eeeeeeee:MSG LOST,RC=dd,TID=ttttt,MMN=nnnnnnnn - SNAP 53 FOLLOWS

FESEND Snap 53

eeeeeeee is the entry point of FESEND; dd is the error code in decimal; ttttt is the destination terminal-id or LU6.2 session name; nnnnnnnn is the message monitor number (MSGHMMN). The message could not be queued; the following reasons (value of dd) are added for LU6.2 sessions:

32 The message being queued to a secondary LU6.2 session is not part of the current transaction on the session (it has a different MSGHBMN than the input message).

36 The message is being queued to a secondary LU6.2 session, which currently is not waiting for an output response message (can happen upon Intercomm restart, or after a transaction processing delay).

40 The message being queued to the master or a primary session (in order to create a LU6.2 conversation) was not a FECMLU6. This return code can also occur if a secondary session is terminated, while a conversation is taking place, and the SLU is returned to the dynamic pool.

VT025I VTAM ERROR CONDITION ID=ii AT LOC aaaaaa, LU=ttttt, SNAP SEQ NO=ssss(HEX)

VTERRMOD Snap 62 or 63

ii is the error-id (hexadecimal number); aaaaaa is the location of the VTERR macro issuance; ttttt is the name of the logical unit causing the error if known, ????? if not; ssss is the Snap sequence number (in register 2 of Snap 62).

The new error conditions encountered by LU6.2 logic are:

25 VTEXITS

An unexpected logical unit issued an UNBIND (was not in session, or not known to Intercomm).

26 VTEXITS

An OPNSEC macro failed on an attempt to BIND a dynamic session. For further information see messages FC134I and FC135I. One of these messages will also have been issued.

97 VTRECVE

A RECEIVE macro specifying the LU6.2 EXIT option failed after a previous RECEIVE had just completed.

98 VTEXITS

A RECEIVE macro specifying the LU6.2 EXIT option failed immediately after the BIND of the session.

99 VTEXITS

No storage available for a RECEIVE buffer on a LU6.2 session after a successful BIND of the session.

A0 VTSEND

No storage available for a RECEIVE buffer on a LU6.2 session after a successful SEND on the session.

A1 VTSEND

A RECEIVE macro with the LU6.2 EXIT option failed, after a SEND had successfully completed.

A2 VTPASS62

The routine to assign primary sessions (PLU's) for messages (FECMLU6's) queued for transmission to an application got an invalid return code from VTSTOUT.
The only valid return codes are:

0 nothing queued, or
4 message is queued.

As output is never actually sent on the master LU, message rescheduling or having a FECMRLSE queued should never occur.

A3 VTPASS62

Having had a return code of 4 from VTSTOUT, the subsequent call to VTDEQUE to retrieve the queued FECMLU6 produced a non-zero return code saying that no message was available. Should not occur.

A4 VTPASS62

Having successfully dequeued a message from the master LUNIT, an attempt to queue the message for the assigned PLU to handle the conversation failed. This should not occur as only one message should ever be on the PLU's queue at a time, and therefore there is no reason for a queuing failure.

B1 VTLUDM6/VTLUDM62

CNOS DRAIN for SHUTD of master LU6.2 session failed.

B2 VTLUDM6/VTLUDM62

CNOS DRAIN IMMEDIATE for HALT of master LU6.2 session failed.

B3 VTLUDM6/VTLUDM62

CNOS to set session limits while initiating a master LU6.2 session failed.

B4 VTLUDM62

A CNOS exchange to set the session limits received a negotiated response increasing the total number of sessions. This is not permitted by LU6.2 protocol.

B5 VTLUDM62

After successful negotiation of CNOS session limits, it was found that there were insufficient dynamic LUNITs defined to match the required session counts. Either lower the session counts, or increase the number of dynamic LUNITs defined.

B6 VTCDM62

The FECMLU6 describing where to route the response on a 6.2 primary session, did not describe any valid destination, (i.e. no terminal, and no subsystem).

B7 VTCDM62

Input on a 6.2 primary session is not a user data stream in response to an invoked transaction (or a reject of same). It is in fact attempting to invoke a new transaction. (This is invalid protocol - contact Intercomm SEOD.)

B8 VTCDM62
At the end of a conversation, when attempting to return the PLU to the available chain, it was found not to be on the in-use chain. Should not occur, contact the Intercomm SEOD.

B9 VTCDM62

A message other than a FECM type LU6.2 (FECMLU6) was queued to a 6.2 primary session. As VTLUDM62 is supposed to trap this situation and reject the queuing of such a message with RC=40 (see message MG600I above), this is an error condition.

BA VTPASS62

Similar to error condition B6, except that it was the timeout routine in VTPASS62 attempting to route message FC109I, and finding nowhere to route it.

BB VTPASS62

Similar to error condition B8, but being issued after a timeout. The PLU was not found on the in-use chain. This should not occur.

C1 VTRECVE

Conditional End Bracket (CEB) received on a LU6.2 session when session not in bracket mode.

VT050I CNOS EXCHANGE SUCCESSFUL ON aaaaaaaa FOR MODENAME yyyyyyyy SESSION COUNTS ARE nn,mm AS {DEFINED }
{NEGOTIATED}

VTLUDM62

Upon successful establishment of 6.2 link, a request to specify the number of sessions that can be used has taken place. This end of the link initiated the request, with application aaaaaaaa, and the sessions will have modename yyyyyyyy. The session counts will be nn total sessions, of which mm are to be primary sessions. (This implies that the number of secondary session will be nn-mm.) These counts exclude the service manager sessions (one SLU and the master LU with modenames of SNASVCMG). These counts will be as DEFINED (via the master LUNIT SESSION parameter) unless the other system wanted less sessions, in which case the lesser values will be used and indicated as NEGOTIATED (see the discussion of session counts in Chapter 1).

VT051I CNOS RECEIVED FROM APPLICATION aaaaaaaa FOR MODENAME yyyyyyyy, SESSION COUNTS WILL BE nn,mm AS {DEFINED }
{REQUESTED }
{NEGOTIATED}

VTCDM62

As part of session establishment on a 6.2 link, this system has been asked how many sessions can be established with this end of the link by application aaaaaaaa, with modename yyyyyyyy. Based on the requested values, this system has responded with session counts of nn total, and mm primary sessions at this end. These counts exclude the service manager sessions (see message VT050I above). These values will be as DEFINED if they coincide with the values this system has defined (master LUNIT SESSION parameter), or as REQUESTED if the other system requested fewer sessions than this system has defined. Should the other system request more sessions than defined in this system, then this system will change the values and indicate they are NEGOTIATED: if the requested total is greater than defined, it will be decremented; if the number of PLU's is greater than defined, the PLU count will be incremented in this system if possible.

VT052I *** FMH-7 *** TRANSACTION vvvv COULD NOT BE PROCESSED BY aaaaaaaa - FMH-7 SENSE CODES ARE xxxxxxxx

VTCDM62

vvvv is the transaction-id (BTVERB) of a LU6.2 transaction sent to VTAM application (APPC system) aaaaaaaa, which returned an FMH-7 response rejecting the transaction. The reason for rejection is provided by sense codes xxxxxxxx, which are documented in IBM VTAM LU6.2 programming manuals.

The invoker of the transaction will receive message FC018I with a reason code of N.

Appendix C

LU6.2 SESSION RESTRICTIONS AND RECOMMENDATIONS

Note: Some or all of these restrictions may be removed in future releases of the LU6.2 interface.

- Unsolicited messages (ones that are not created as a direct response to an input) will not be sent across any session to CICS or any other APPC system. The message will be logged and discarded by the Front End (FESEND), message MG600I is issued (see Appendix B) and the FESEND return code is not zero.

- Subsystems should not use conversational facilities by calling CONVERSE. However, if user exits (see Chapter 3) to substitute a real terminal-id for the session name are used, facilities such as CONVERSE and Store/Fetch, which may use the tid as a key, can then be accessed.

- Subsystems should not attempt to lock the input "terminal" to a verb; neither by the AUTOLOK facility (BTVERB macro), nor by issuing the LOCK command.

- If ESS is in use for terminals logged on to Intercomm, then the names of the LU6.2 session LUB/LUC pairs must be included in the terminal-exempt list to avoid ESS rejection of CICS or other system input transaction processing. ESS does not support LU6.2 session "terminals", because the session terminal-id may change for each new input transaction from the same physical terminal.

- RESTART=NO should be specified for the LU6.2 dynamic session LUs (via the shared VTCSB) and for associated subsystem processing. Subsystems may be restarted, but are likely to get non-zero return codes upon their attempts to queue an output response, and the output response will be discarded by the Front End.

- 3270 CRT AID values and cursor positioning for input messages on secondary sessions is not supported, that is, BTVERB macro coding of HDR3270=YES is ignored. However, code can be added to the CICS or other system application to pass the AID and CUR portion of the header at the beginning of the message text.

- In a Multiregion Intercomm, coding the MRPASSW parameter (to lock all LU6.2 secondary sessions to a specific satellite region) on the dynamic sessions LUNIT macro will be effective only if the user does not change the session name in MSGHTID in the message header of the input message. The test for region locking for the MSGHTID terminal (to determine message routing) is executed by Message Collection after INQUEUE user exit processing.

- In active mode, the accepting, or producing, of Program Initialization Parameters (PIPs) is not yet supported.

- The ability to BID on Contention Loser Sessions (secondary sessions or SLU's) to try to send a transaction is not yet supported, neither by Intercomm as the receiver of a BID, nor by Intercomm as an initiator of a BID.

- Dynamic changing of session totals (a CNOS exchange) after the LU6.2 link and its sessions are originally established is not yet supported.